PHASE 3 SYSTEMS DESIGN

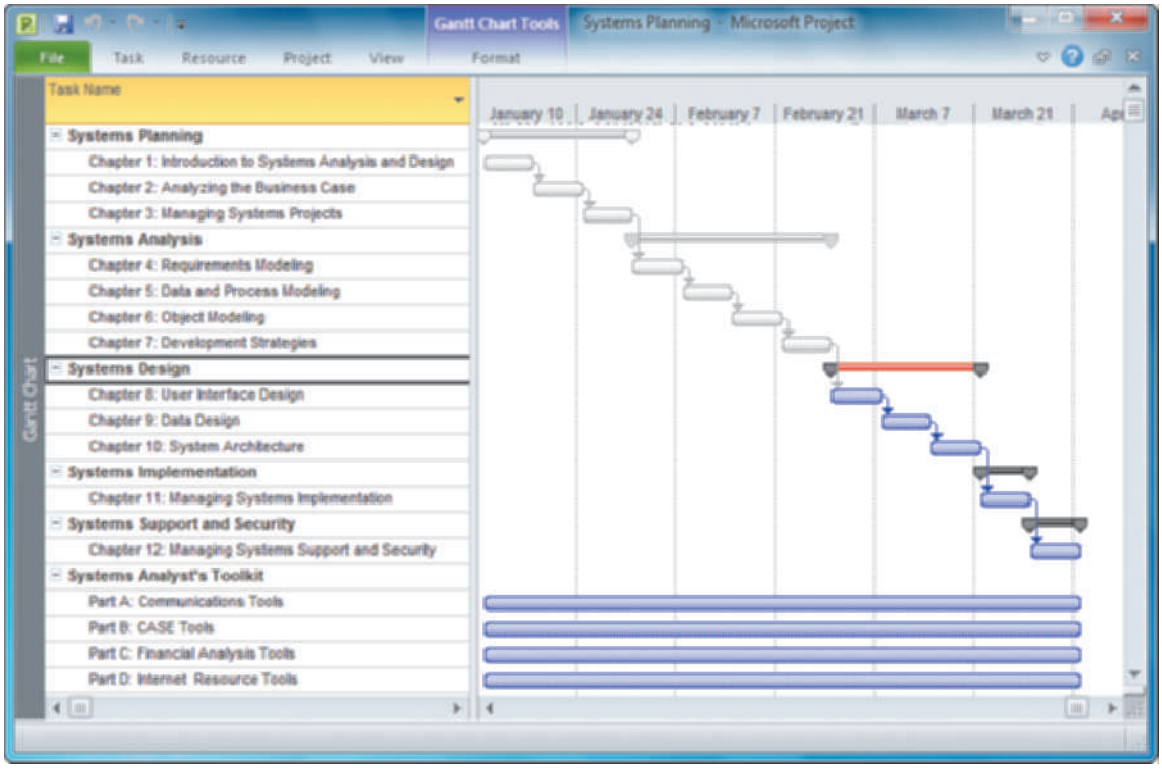
DELIVERABLE
System design specification

TOOLKIT SUPPORT
Primary tools: Communications and CASE tools

VIDEO LEARNING SESSIONS
Entity Relationship Diagrams, Data Normalization

As the Dilbert cartoon suggests, you should understand a problem before you decide that a database is the solution. You will learn more about systems design topics, including database design, in the systems design phase.

Systems design is the third of five phases in the systems development life cycle. In the previous phase, systems analysis, you developed a logical model of the new system. Now you will work on a physical design that will meet the specifications described in the system requirements document. Your tasks will include user interface design, data design, and system architecture. The deliverable for this phase is the system design specification.



CHAPTER 8 User Interface Design

Chapter 8 is the first of three chapters in the systems design phase of the SDLC. This chapter explains how to design an effective user interface, and how to handle data output, input, and security issues. The chapter stresses the importance of user feedback and involvement in all design decisions.

INTRODUCTION

OBJECTIVES

When you finish this chapter, you will be able to:

- Explain the concept of user interface design and human-computer interaction, including basic principles of user-centered design
- List user interface design guidelines
- Describe user interface components, including screen elements and controls
- Discuss output design and technology issues
- Design effective source documents
- Explain input design and technology issues
- Discuss guidelines for data entry screen design
- Use input masks and validation rules to reduce input errors
- Describe output and input controls and security

User interface design is the first task in the systems design phase of the SDLC. Designing the interface is extremely important, because everyone wants a system that is easy to learn and use.

After discussing user interface evolution, principles, and design techniques, the chapter describes output design, input design, and data security issues.

CHAPTER INTRODUCTION CASE: Mountain View College Bookstore

Background: Wendy Lee, manager of college services at Mountain View College, wants a new information system that will improve efficiency and customer service at the three college bookstores.

In this part of the case, Tina Allen (systems analyst) and David Conroe (student intern) are talking about user interface design issues.

Participants: Tina and David
Location: Mountain View College Cafeteria, Monday afternoon, November 28, 2011
Project status: Tina and David have examined development strategies for the new bookstore system. After performing cost-benefit analysis, they recommended in-house development of the new bookstore system. Now they are ready to begin the systems design phase by working on user interface design for the new system.
Discussion topics: User interface design concepts and principles

Tina: Hi, David. Ready to start work on user interface design?

David: *Sure. Will we start with output, because it's important to users?*

Tina: Output is very important, but the *most* important issue for users is the interface itself. For example, is it easy to learn? Is it easy to work with? We'll try to design everything — output, input, and all the other elements — from a user's point of view.

David: *How do we do that?*

Tina: Well, many sources of information about effective design concepts and principles are available. We'll study those, and then ask our own users for their input and suggestions.

David: *What about input and data entry?*

Tina: Good question. You've heard the old saying, "garbage in, garbage out." User interface principles apply to user input generally, but repetitive data entry deserves special attention. We need to create screen forms that are logical and easy to understand, as well as input masks and data entry validation rules. We also need to review any source documents that will be filled in manually.

David: *Anything else?*

Tina: Yes. The bookstore system probably will have some confidential data regarding budgets and markup policies, so we'll have to consider output and input control and security. If you're ready, here's a task list to get us started:

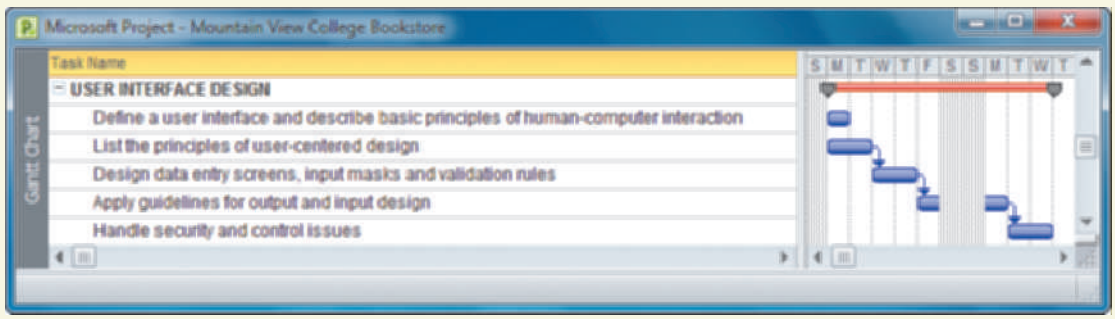


FIGURE 8-1 Typical user interface design tasks.

ON THE WEB

To learn more about user interface design, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the User Interface Design link.

What Is a User Interface?

A **user interface** (UI) describes how users interact with a computer system, and consists of all the hardware, software, screens, menus, functions, output, and features that affect two-way communications between the user and the computer.

Figure 8-2 suggests an interesting viewpoint that interface designers should keep in mind: Industry leader IBM believes that the best interfaces are the ones that users do not even notice — they make sense because they do what users expect them to do.

FIGURE 8-2 IBM says that a user interface is what you see, hear, touch, or talk to when you use a computer.

Evolution of the User Interface

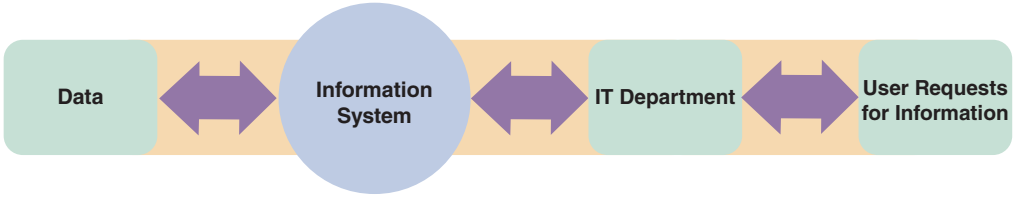
When developing older systems, analysts typically designed all the printed and screen output first, then worked on the inputs necessary to produce the results. Often, the user interface mainly consisted of **process-control** screens that allowed the user to send commands to the system. That approach worked well with traditional systems that simply transformed input data into structured output.

As information management evolved from centralized data processing to dynamic, enterprise-wide systems, the primary focus also shifted — from the IT department to the users themselves. The IT group became a supplier of information technology,

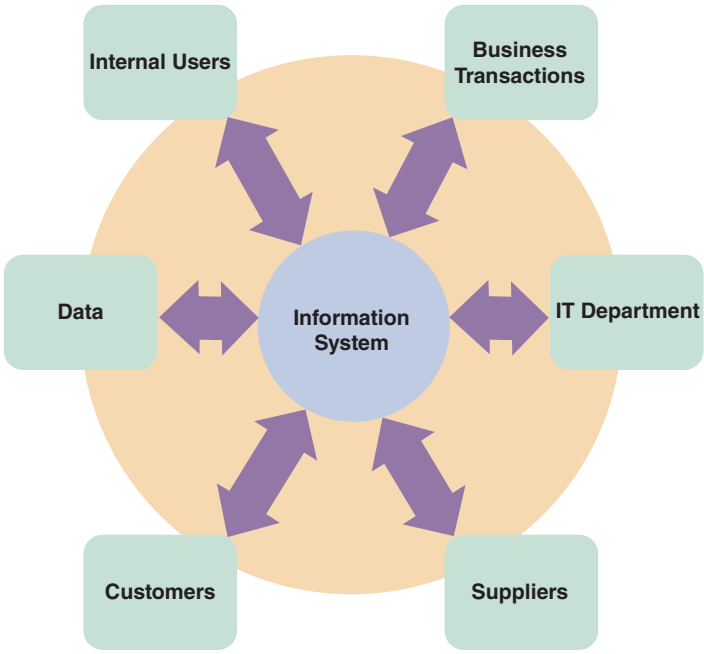
rather than a supplier of information. Today, the main focus is on users within and outside the company, how they communicate with the information system, and how the system supports the firm's business operations. Figure 8-3 compares a traditional, processing-centered information system with a modern, user-centered system. Notice that the IT department, which was the main interface for user information requests, has become a system facilitator that maintains and supports the system for its users.

In a **user-centered** system, the distinction blurs between input, output, and the interface itself. Most users work with a varied mix of input, screen output, and data queries as they perform their day-to-day job functions. Because all those tasks require interaction with the computer system, the user interface is a vital element in the systems design phase. Ergosoft laboratories is one of many firms that offer consulting services and software solutions to help companies develop successful user interfaces, as shown in Figure 8-4 on the next page.

User interface design requires an understanding of human-computer interaction and user-centered design principles, which are discussed in the next section. Input and output design topics are covered later in this chapter.



Traditional, Processing-Centered Information System Model



Modern, User-Centered Information System Model

FIGURE 8-3 Compare the traditional, processing-centered system at the top of the figure with the modern, user-centered information system at the bottom. Notice the change in the role of the IT department.

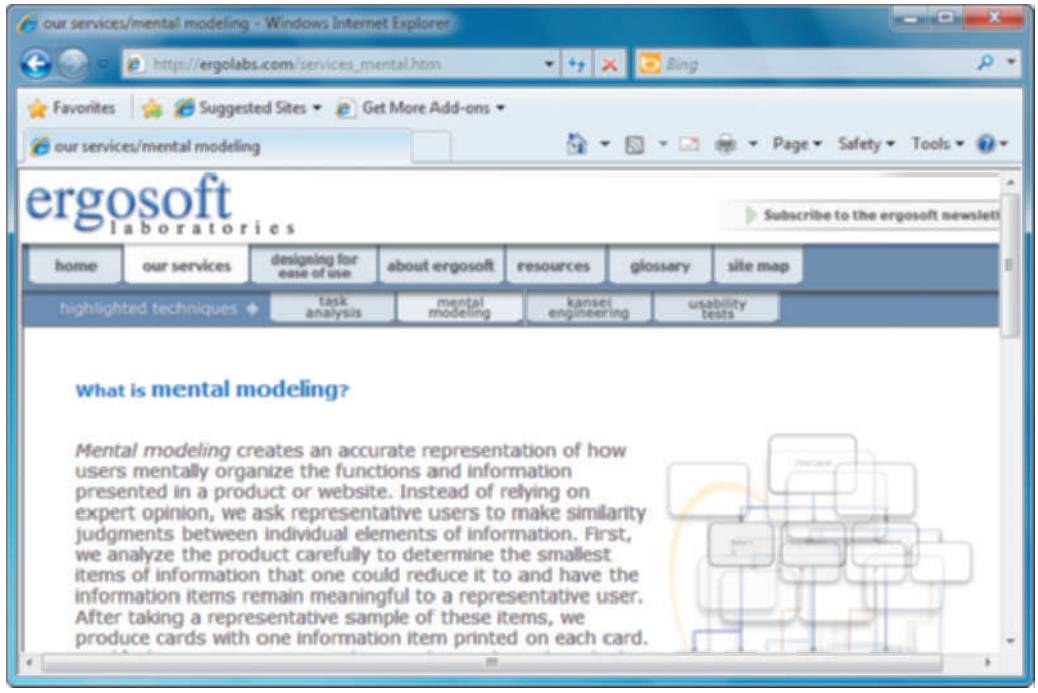


FIGURE 8-4 Ergosoft laboratories is an example of a firm that offers consulting services and software solutions to help companies develop successful user interfaces.

Human-Computer Interaction

A user interface is based on basic principles of human-computer interaction. **Human-computer interaction** (**HCI**) describes the relationship between computers and people who use them to perform their jobs, like the worker shown in Figure 8-5. HCI concepts apply to everything from PC desktops to global networks. In its broadest sense, a user interface includes all the communications and instructions necessary to enter input to the system and to obtain output in the form of screen displays or printed reports.

The human-computer interface started in the 1980s with users typing complex commands in green text on a black screen. Then came the **graphical user interface** (**GUI**), which was a huge improvement, because it used icons, graphical objects, and pointing devices. Today, designers strive to translate user behavior, needs, and desires into an interface that users don't really notice. As IBM points out in Figure 8-2 on page 336, the best user interfaces are "*almost transparent* — you can see right though the interface to your own work."

FIGURE 8-5 HCI is essential to employee productivity, whether the work is done in a traditional office setting, or on a construction site like the one shown here.

As a systems analyst, you will design user interfaces for in-house developed software and customize interfaces for various commercial packages and user productivity applications. Your main objective is to create a user-friendly design that is easy to learn and use.

Industry leaders Microsoft and IBM both devote considerable resources to user interface research. Figure 8-6 describes Microsoft's Redmond labs, where engineers observe volunteers who participate in software usability studies.

At its Almaden Research Center, IBM conducts usability testing and studies human-computer interaction, as shown in Figure 8-7. According to IBM, its User Sciences & Experience Research (USER) lab focuses on improving ease of use and exploring new ways of using computers.

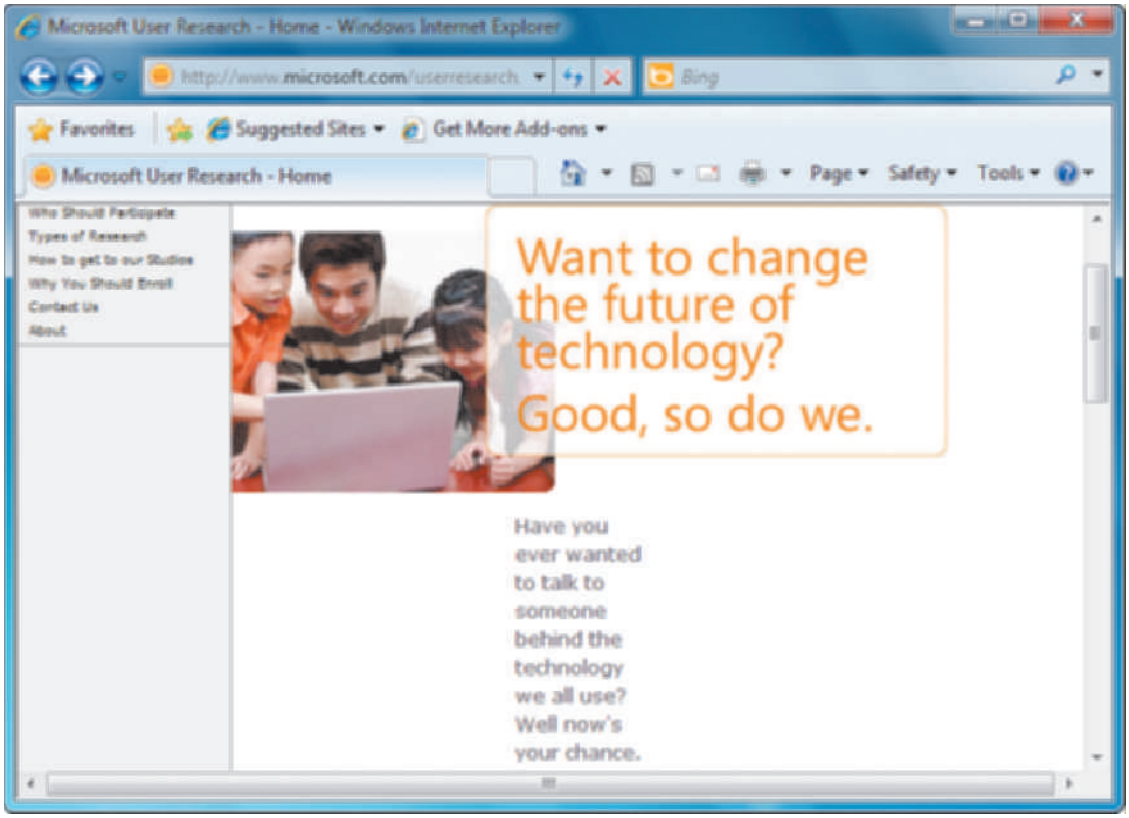


FIGURE 8-6 Microsoft invites people to participate in software usability studies at its labs.

FIGURE 8-7 According to IBM, human-computer interaction (HCI) is one of the most extensive research areas at the company.

CASE IN POINT 8.1: CASUAL OBSERVER SOFTWARE

Casual Observer Software's main product is a program that monitors and analyzes user keystrokes and mouse clicks to learn more about the way employees use their computer systems. The problem is that some users feel this is an unwarranted intrusion into their privacy, and they prefer not to be observed. Some even fear that the data would be used for other reasons, including performance appraisal. You are a consultant who has been hired by a client firm that is trying to decide whether or not to use this software.

Before you advise the client, go back and review the Microsoft usability lab shown in Figure 8-6 on the previous page, where the users being studied in the Redmond labs were willing participants. Then, refer to Chapter 4, Requirements Modeling, page 165, and consider the Hawthorne Effect, which suggests that employees might behave differently when they know they are being observed. Finally, think about the ethical issues that might be involved in this situation. What will you advise your client, and why?

ON THE WEB

To learn more about human-computer interaction, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Human-Computer Interaction link.

IBM believes that the user interface evolution will lead to computers that truly are *consumer products* that are simple and natural for the general population to use. This will occur, in IBM's view, because computers will function in a friendlier, more predictable way — much like a telephone or video player. Most important, the interface will be based on the perspective of a user rather than a computer engineer, programmer, or systems analyst. To understand the magnitude of this shift in thinking, consider the powerful statement shown in Figure 8-8, where IBM usability expert Dr. Clare-Marie Karat notes that "in this new computer age, the customer is not only right, the customer has rights." Those rights are listed in Figure 8-9.

FIGURE 8-8 IBM's logic is very simple: The user is the customer, and the customer is always right.

User Rights

1. **Perspective:** The user always is right. If there is a problem with the use of the system, the system is the problem, not the user.
2. **Installation:** The user has the right to install and uninstall software and hardware systems easily without negative consequences.
3. **Compliance:** The user has the right to a system that performs exactly as promised.
4. **Instruction:** The user has the right to easy-to-use instructions (user guides, online or contextual help, and error messages) for understanding and utilizing a system to achieve desired goals and recover efficiently and gracefully from problem situations.
5. **Control:** The user has the right to be in control of the system and to be able to get the system to respond to a request for attention.
6. **Feedback:** The user has the right to a systen that provides clear, understandable, and accurate information regarding the task it is performing and the progress toward completion.
7. **Dependencies:** The user has the right to be informed clearly about all systems requirements for successfully using software or hardware.
8. **Scope:** The user has the right to know the limits of the system's capabilities.
9. **Assistance:** The user has the right to communicate with the technology provider and receive a thoughtful and helpful response when raising concerns.
10. **Usability:** The user should be the master of software and hardware technology, not vice versa. Products should be natural and intuitive to use.

FIGURE 8-9 User rights suggested by IBM's Dr. Clare-Marie Karat.

Principles of User-Centered Design

Although IT professionals have different views about interface design, most would agree that good design depends on seven basic principles, which are described in the following sections.

Understand the Business

The interface designer must understand the underlying business functions and how the system supports individual, departmental, and enterprise goals. The overall objective is to design an interface that helps users to perform their jobs. A good starting point might be to analyze a functional decomposition diagram (FDD). As you learned in Chapter 4, an FDD is a graphical representation of business functions that starts with major functions, and then breaks them down into several levels of detail. An FDD can provide a checklist of user tasks that you must include in the interface design.

VIDEO LEARNING SESSIONS

To learn more about FDDs, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and navigate to the **Video Learning Sessions** for this book. These sessions can help you understand key concepts, practice your skills, and check your work.

Maximize Graphical Effectiveness

Studies show that people learn better visually. The immense popularity of Apple Mac OS and Microsoft Windows is largely the result of their graphical user interfaces that are easy to learn and use. A well-designed interface can help users learn a new system rapidly, and be more productive. Also, in a graphical environment, a user can display and work with multiple windows on a single screen and transfer data between programs. If the interface supports data entry, it must follow the guidelines for data entry screen design that are discussed later in this chapter.

Think Like a User

A systems analyst should understand user experience, knowledge, and skill levels. If a wide range of capability exists, the interface should be flexible enough to accommodate novices as well as experienced users.

To develop a user-centered interface, the designer must learn to think like a user and see the system through a user's eyes. The interface should use terms and metaphors that are familiar to users. Users are likely to have real-world experience with many other

machines and devices that provide feedback, such as automobiles, ATM machines, and microwave ovens. Based on that experience, users will expect useful, understandable feedback from a computer system.

Use Models and Prototypes

From a user's viewpoint, the interface is the most critical part of the system design because it is where he or she interacts with the system — perhaps for many hours each day. It is essential to construct models and prototypes for user approval. An interface designer should obtain as much feedback as possible, as early as possible. You can present initial screen designs to users in the form of a **storyboard,** which is a sketch that shows the general screen layout and design. The storyboard can be created with software, or drawn freehand. Users must test all aspects of the interface design and provide feedback to the designers. User input can be obtained in interviews, via questionnaires, and by observation. Interface designers also can obtain data, called **usability metrics,** by using software that can record and measure user interaction with the system.

Focus on Usability

TOOLKIT TIME

The Communication Tools in Part A of the Systems Analyst's Toolkit can help you communicate effectively with users. To learn more about these tools, turn to Part A of the four-part Toolkit that follows Chapter 12.

The user interface should include all tasks, commands, and communications between users and the information system. The screen in Figure 8-10 shows the main options for a student registration system. Each screen option leads to another screen, with more options. The objective is to offer a reasonable number of choices that a user easily can comprehend. Too many options on one screen can confuse a user — but too few options increase the number of submenu levels and complicate the navigation process. Often, an effective strategy is to present the most common choice as a default, but allow the user to select other options.

Invite Feedback

Even after the system is operational, it is important to monitor system usage and solicit user suggestions. You can determine if system features are being used as intended by observing and surveying users. Sometimes, full-scale operations highlight problems that were not apparent when the prototype was tested. Based on user feedback, Help screens might need revision and design changes to allow the system to reach its full potential.

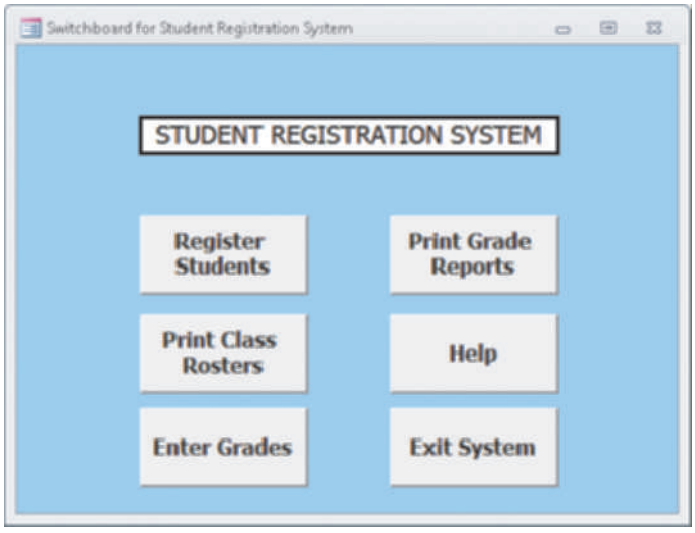


FIGURE 8-10 The opening screen displays the main options for a student registration system. A user can click an option to see lower-level actions and menu choices.

Document Everything

You should document all screen designs for later use by programmers. If you are using a CASE tool or screen generator, number the screen designs and save them in a hierarchy similar to a menu tree. User-approved sketches and storyboards also can be used to document the user interface.

By applying basic user-centered design principles, a systems analyst can plan, design, and deliver a successful user interface.

Designing the User Interface

It is important to design a user interface that is easy to use, attractive, and efficient. When you create a user interface, you should follow eight basic guidelines. These guidelines also apply to data entry screen design, which is discussed later in this chapter.

1. Design a transparent interface.
2. Create an interface that is easy to learn and use.
3. Enhance user productivity.
4. Make it easy for users to obtain help or correct errors.
5. Minimize input data problems.
6. Provide feedback to users.
7. Create an attractive layout and design.
8. Use familiar terms and images.

Good user interface design is based on a combination of ergonomics, aesthetics, and interface technology. **Ergonomics** describes how people work, learn, and interact with computers; **aesthetics** focuses on how an interface can be made attractive and easy to use; and **interface technology** provides the operational structure required to carry out the design objectives. As shown in Figure 8-11, Cognetics Corporation offers user interface design services. Cognetics stresses that an interface must be effective, efficient, engaging, error tolerant, and easy to learn.

The following sections provide examples of the basic user interface design guidelines. As mentioned earlier, many of the specific points also apply to data entry screen design, which is discussed later in this chapter.

Design a Transparent Interface

- Facilitate the system design objectives, rather than calling attention to the interface.
- Create a design that is easy to learn and remember.
- Design the interface to improve user efficiency and productivity.
- Write commands, actions, and system responses that are consistent and predictable.
- Minimize data entry problems.
- Allow users to correct errors easily.
- Create a logical and attractive layout.

FIGURE 8-11 Cognetics Corporation offers user interface design services and training.

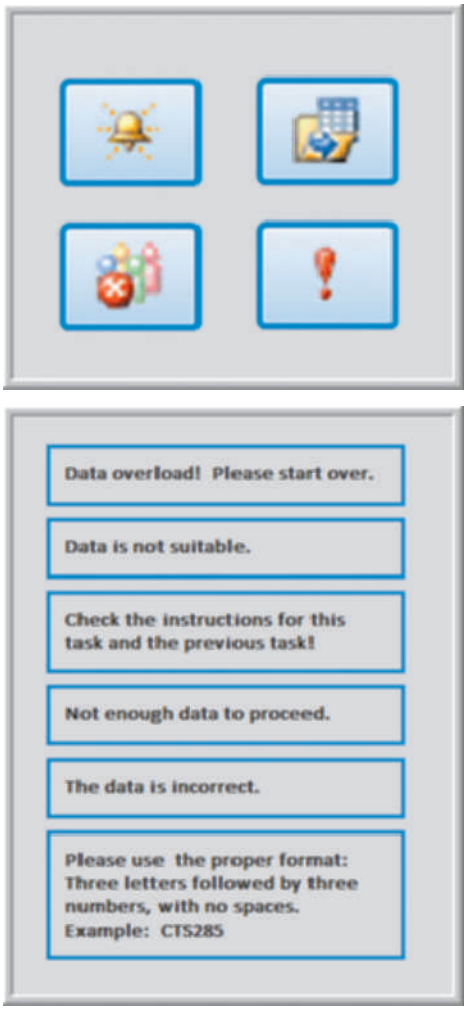


FIGURE 8-12 In the example at the top, the icons do not have a clear message. In the Help text examples at the bottom, only one message is understandable. The others would frustrate and annoy most users.

Create an Interface that Is Easy to Learn and Use

- Clearly label all controls, buttons, and icons.
- Select only those images that users can understand easily, and provide on-screen instructions that are logical, concise, and clear. Users become very frustrated when they see images or messages that are confusing or misleading. For example, the top screen in Figure 8-12 shows four control buttons, but many of them do not have an obvious meaning. In the bottom screen, notice the difference in the messages: The first five provide little or no information. The last message is the only one that is easy to understand.
- Show all commands in a list of menu items, but dim any commands that are not currently available.
- Make it easy to navigate or return to any level in the menu structure.

Enhance User Productivity

- Organize tasks, commands, and functions in groups that resemble actual business operations. You should group functions and submenu items in a multilevel menu hierarchy, or tree, that is logical and reflects how users typically perform the tasks. Figure 8-13 shows an example of a menu hierarchy for an order tracking system.
- Create alphabetical menu lists or place the selections used frequently at the top of the menu list. No universally accepted approach to menu item placement exists. The best strategy is to design a prototype and obtain feedback from users. Some applications even allow menus to show recently used commands first. Some users like that feature, but others might find it distracting. The best approach is to offer a choice, and let users decide.
- Provide shortcuts so experienced users can avoid multiple menu levels. You can create shortcuts using hot keys that allow a user to press the ALT key + the underlined letter of a command.
- Use default values if the majority of values in a field are the same. For example, if 90% of the firm's customers live in Albuquerque, use *Albuquerque* as the default value in the City field.
- Use a duplicate value function that enables users to insert the value from the same field in the previous record.
- Provide a fast-find feature that displays a list of possible values as soon as users enter the first few letters.
- Use a **natural language** feature that allows users to type commands or requests in normal English phrases. For example, many applications allow users to request Help by typing a question into a dialog box. The software then uses natural language technology to retrieve a list of topics that match the request. Most users like natural language features because they do not have to memorize a series of complex commands and syntax. According to the American Association for Artificial Intelligence (AAAI), the value of being able to communicate with computers in everyday "natural" language cannot be overstated. Natural language technology is used in speech recognition systems, text-to-speech synthesizers, automated voice response systems, Web search engines, text editors, and language instruction materials.

Make It Easy for Users to Obtain Help or Correct Errors

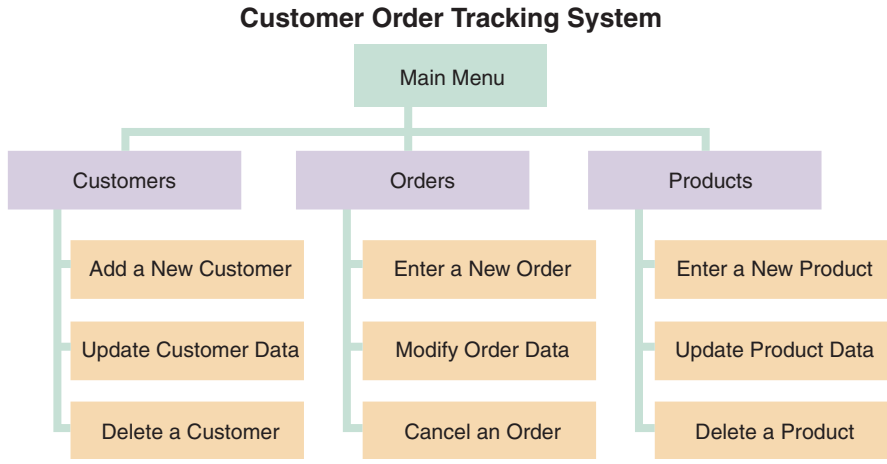


FIGURE 8-13 This menu hierarchy shows tasks, commands, and functions organized into logical groups and sequences. The structure resembles a functional decomposition diagram (FDD), which is a model of business functions and processes.

- Ensure that help is always available. Help screens should provide information about menu choices, procedures, shortcuts, and errors.
- Provide user-selected help and context-sensitive help. **User-selected** help displays information when the user requests it. By making appropriate choices through the menus and submenus, the user eventually reaches a screen with the desired information. Figure 8-14 shows the main Help screen for the student registration system. **Context-sensitive** help offers assistance for the task in progress. Figure 8-15 on the next page shows a Help dialog box that is displayed if a user requests help while entering data into the ADVISOR ASSIGNED field. Clicking the Close button returns the user to the current task.
- Provide a direct route for users to return to the point from where help was requested. Title every help screen to identify the topic, and keep help text simple and concise. Insert blank lines between paragraphs to make Help easier to read, and provide examples where appropriate.
- Include contact information, such as a telephone extension or e-mail address if a department or help desk is responsible for assisting users.
- Require user confirmation before data deletion (*Are you sure?*) and provide a method of recovering data that is deleted inadvertently. Build in safeguards that prevent critical data from being changed or erased.
- Provide an Undo key or a menu choice that allows the user to eradicate the results of the most recent command or action.
- When a user-entered command contains an error, highlight the erroneous part and allow the user to make the correction without retyping the entire command.
- Use hypertext links to assist users as they navigate through help topics.

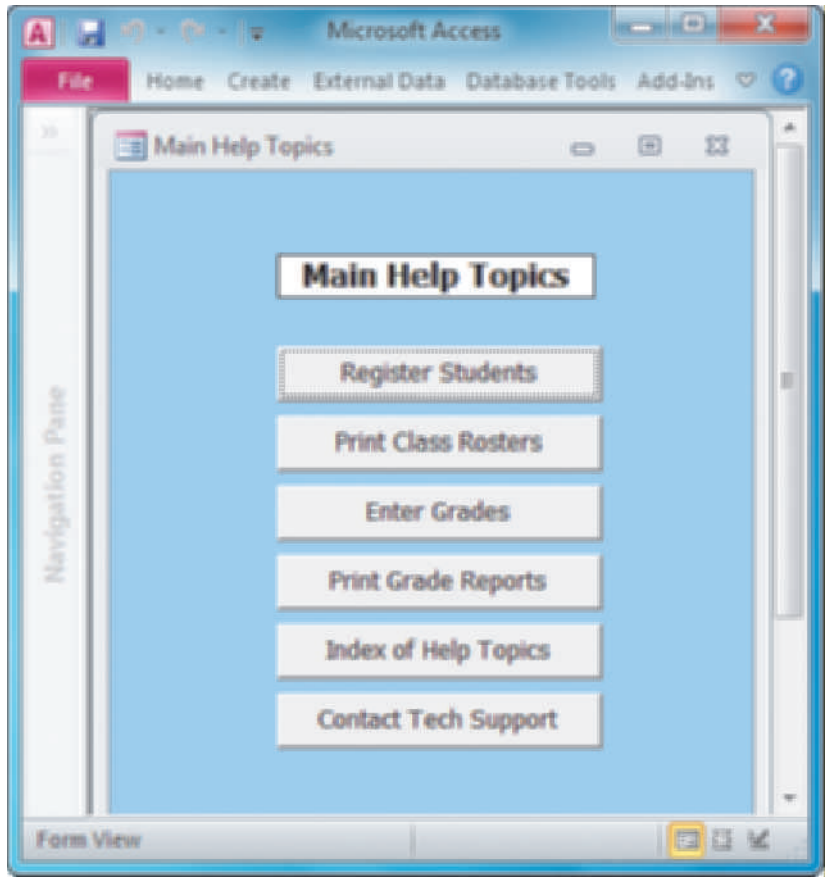


FIGURE 8-14 The main Help screen for a student registration system.

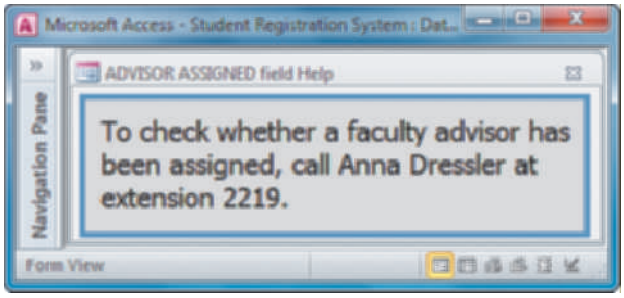


FIGURE 8-15 A context-sensitive dialog box displays if a user requests help while entering data into the ADVISOR ASSIGNED field. Clicking the Close button returns the user to the task.

Minimize Input Data Problems

- Create **input masks,** which are templates or patterns that make it easier for users to enter data. Also use data validation rules, which limit the acceptable values that users can enter. More information on input masks and data validation rules is provided in the section on input design later in the chapter.
- Display event-driven messages and reminders. Just as context-sensitive help is important to users, it is desirable to display an appropriate message when it is time for the user to perform a certain task. For example, when exiting the system, a message might ask users if they want a printed report of the data entered during the recent session.
- Establish a list of predefined values that users can click to select. Predefined values prevent spelling errors, avoid inappropriate data in a field, and make the user's job easier — the input screen displays a list of acceptable values and the user simply points and clicks the choice.
- Build in rules that enforce data integrity. For example, if the user tries to enter an order for a new customer, the customer must be added before the system will accept the order data.

Provide Feedback to Users

- Display messages at a logical place on the screen, and be consistent.
- Alert users to lengthy processing times or delays. Give users an on-screen progress report, especially if the delay is lengthy.
- Allow messages to remain on the screen long enough for users to read them. In some cases, the screen should display messages until the user takes some action.
- Let the user know whether the task or operation was successful or not. For example, use messages such as Update completed, All transactions have been posted, or ID Number not fo*und*.
- Provide a text explanation if you use an icon or image on a control button. This helps the user to identify the control button when moving the mouse pointer over the icon or image.
- Use messages that are specific, understandable, and professional. Avoid messages that are cute, cryptic, or vague, such as: *ERROR — You have entered an unacceptable value*, or *Error DE4-16*. Better examples are: *Enter a number between 1 and 5; Customer number must be numeric. Please re-enter a numeric value;* or *Call the Accounting Department, Ext. 239 for assistance.*

Create an Attractive Layout and Design

- Use appropriate colors to highlight different areas of the screen; avoid gaudy and bright colors.
- Use special effects sparingly. For example, animation and sound might be effective in some situations, but too many special effects can be distracting and annoying to a user, especially if he or she must view them repeatedly.
- Use hyperlinks that allow users to jump to related topics.
- Group related objects and information. Visualize the screen the way a user will see it, and simulate the tasks that the user will perform.
- Screen density is important. Keep screen displays uncluttered, with enough white space to create an attractive, readable design.
- Display titles, messages, and instructions in a consistent manner and in the same general locations on all screens.
- Use consistent terminology. For example, do not use the terms *delete*, *cancel*, and *erase* to indicate the same action. Similarly, the same sound always should signal the same event.
- Ensure that commands always will have the same effect. For example, if the *BACK* control button returns a user to the prior screen, the *BACK* command always should perform that function throughout the application.
- Ensure that similar mouse actions will produce the same results throughout the application. The results of pointing, clicking, and double-clicking should be consistent and predictable.
- When the user enters data that completely fills the field, do not move automatically to the next field. Instead, require the user to confirm the entry by pressing the ENTER key or TAB key at the end of every fill-in field.

Use Familiar Terms and Images

- Remember that users are accustomed to a pattern of *red* = *stop*, *yellow* = *caution*, and *green* = *go*. Stick to that pattern and use it when appropriate to reinforce on-screen instructions.
- Provide a keystroke alternative for each menu command, with easy-to-remember letters, such as File, Exit, and Help.
- Use familiar commands if possible, such as Cut, Copy, and Paste.
- Provide a Windows look and feel in your interface design if users are familiar with Windows-based applications.
- Avoid complex terms and technical jargon; instead, select terms that come from everyday business processes and the vocabulary of a typical user.

Add Control Features

The designer can include many features, such as menu bars, toolbars, dialog boxes, text boxes, toggle buttons, list boxes, scroll bars, drop-down list boxes, option buttons, check boxes, command buttons, and calendar controls, among others. Figure 8-16 on the next page shows a data entry screen for the student registration system. The screen design uses several features that are described in the following section.

The **menu bar** at the top of the screen displays the main menu options. Some software packages allow you to create customized menu bars and toolbars. You can add a shortcut feature that lets a user select a menu command either by clicking the desired choice or by pressing the ALT key + the underlined letter. Some forms also use a **toolbar** that contains icons or buttons that represent shortcuts for executing common commands.

A **command button** initiates an action such as printing a form or requesting help. For example, when a user clicks the Find Student command button in Figure 8-16, a dialog box opens with instructions, as shown in Figure 8-17.

Other design features include dialog boxes, text boxes, toggle buttons, list boxes, scroll bars, drop-down list boxes, option or radio buttons, check boxes, and calendar controls. These features are described as follows:

- A **dialog box** allows a user to enter information about a task that the system will perform.
- A **text box** can display messages or provide a place for a user to enter data.
- A **toggle button** is used to represent on or off status — clicking the toggle button switches to the other status.

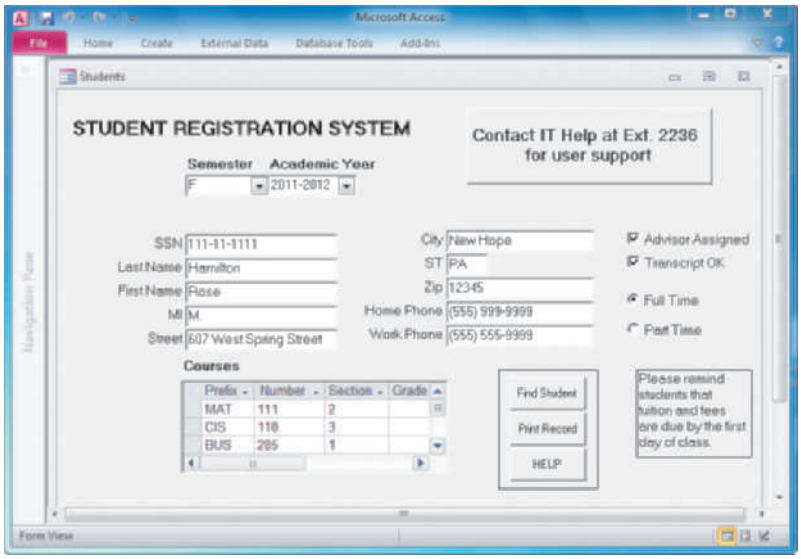


FIGURE 8-16 A data entry screen for the student registration system. This screen uses several design features that are described in the text.

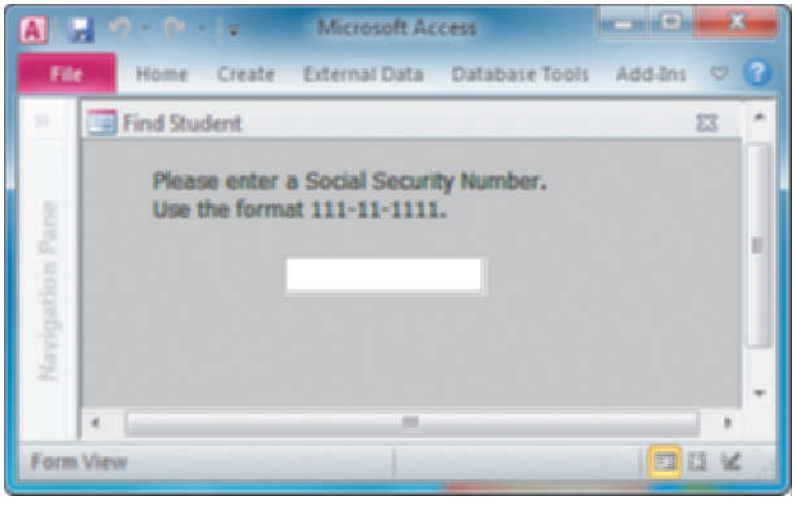


FIGURE 8-17 When a user clicks the Find Student command button, a dialog box is displayed with instructions.

- A **list box** displays a list of choices that the user can select. If the list does not fit in the box, a **scroll bar** allows the user to move through the available choices.
- A **drop-down list box** displays the current selection; when the user clicks the arrow, a list of the available choices displays.
- An **option button**, or **radio button**, represents one choice in a set of options. The user can select only one option at a time, and selected options show a black dot.
- A **check box** is used to select one or more choices from a group. Selected options are represented by a check-mark or an X.
- A **calendar control** allows the user to select a date that the system will use as a field value.

Screen design requires a sense of aesthetics as well as technical skills. You should design screens that are attractive, easy to use, and workable. You also should obtain user feedback early and often as the design process continues.

The opening screen is especially important because it introduces the application and allows users to view the main options. When designing an opening screen, you can use a main form that functions as a switchboard. A **switchboard** uses command buttons that enable users to navigate the system and select from groups of related

tasks. Figure 8-18 shows the switchboard and a data entry screen for a project management system. Notice the drop-down list box that allows users to enter a status code simply by clicking a selection.

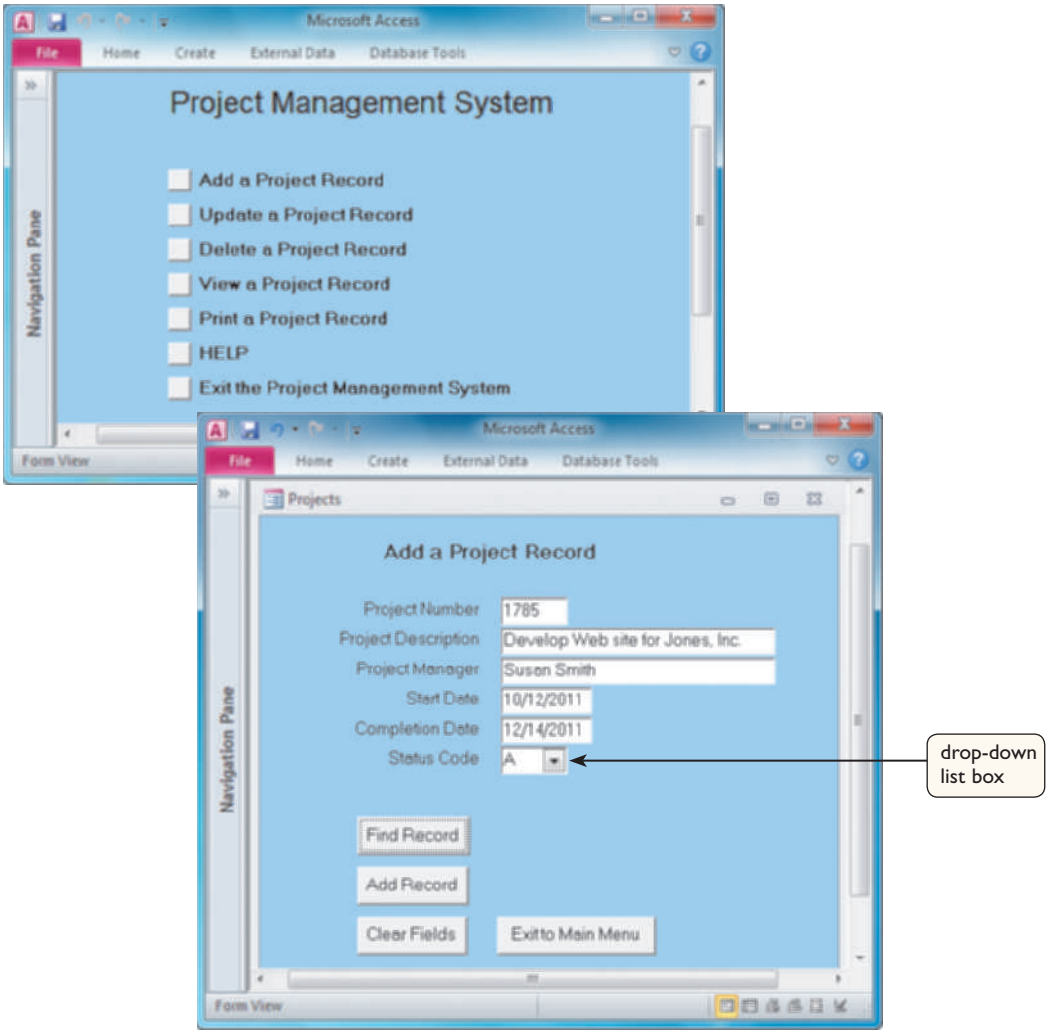


FIGURE 8-18 An example of a switchboard and data entry screen for a project management system.

CASE IN POINT 8.2: BOOLEAN TOYS

When should a systems analyst decide a design issue, and when should users be allowed to select what works best for them? The field of ergonomics is concerned with improving the work environment and studying how users interact with their environment.

Suppose you are a systems analyst studying the order processing system at Boolean Toys, a fast-growing developer of software for preschool children. You know that many data entry users have complained about the input screens. Some users would prefer to rearrange the order of the fields; others would like to change the background color on their screens; still others want shortcuts that would allow them to avoid a series of introductory screens.

What if Boolean's users could customize their own data entry screens without assistance from the IT staff by using a menu-driven utility program? What would be the pros and cons of such an approach?

Output Design

Before designing output, ask yourself several questions:

- What is the purpose of the output?
- Who wants the information, why is it needed, and how will it be used?
- What specific information will be included?
- Will the output be printed, viewed on-screen, or both? What type of device will the output go to?
- When will the information be provided, and how often must it be updated?
- Do security or confidentiality issues exist?

The design process should not begin until you have answered those questions. Some of the information probably was gathered during the systems analysis phase. To complete your understanding, you should meet with users to find out exactly what kind of output is needed. You can use prototypes and mock-ups to obtain feedback throughout the design process. Your answers will affect your output design strategies, as you will see in the next section.

In today's interconnected world, output from one system often becomes input for another system. For example, within a company, production data from the manufacturing system becomes input to the inventory system. The same company might transmit employee W-2 tax data to the IRS system electronically. A company employee might use tax preparation software to file a tax return online, receive a refund deposited directly into his or her bank account, and see the deposit reflected on the bank's information system.

Although digital technology has opened new horizons in business communications, printed material still is a common type of output, and specific considerations apply to it. For those reasons, printed and screen reports are discussed in a separate section, which follows.

Overview of Report Design

Although many organizations strive to reduce the flow of paper and printed reports, few firms have been able to eliminate printed output totally. Because they are portable, printed reports are convenient, and even necessary in some situations. Many users find it handy to view screen output, then print the information they need for a discussion or business meeting. Printed output also is used in **turnaround documents**, which are output documents that are later entered back into the same or another information system. In some areas, your telephone or utility bill, for example, might be a turnaround document printed by the company's billing system. When you return the required portion of the bill with your check, the bill is scanned into the company's accounts receivable system to record the payment accurately.

Designers use a variety of styles, fonts, and images to produce reports that are attractive and user friendly. Whether printed or viewed on-screen, reports must be easy to read and well organized. Rightly or wrongly, some managers judge an entire project by the quality of the reports they receive.

Database programs such as Microsoft Access include a variety of report design tools, including a Report Wizard, which is a menu-driven feature that designers can use to create reports quickly and easily. Microsoft Access also provides a comprehensive guide to designing reports, as shown in Figure 8-19.

In addition to built-in design tools, popular software packages such as Crystal Reports offer powerful features that help designers deal with professional-level design issues across the enterprise, as shown in Figure 8-20.

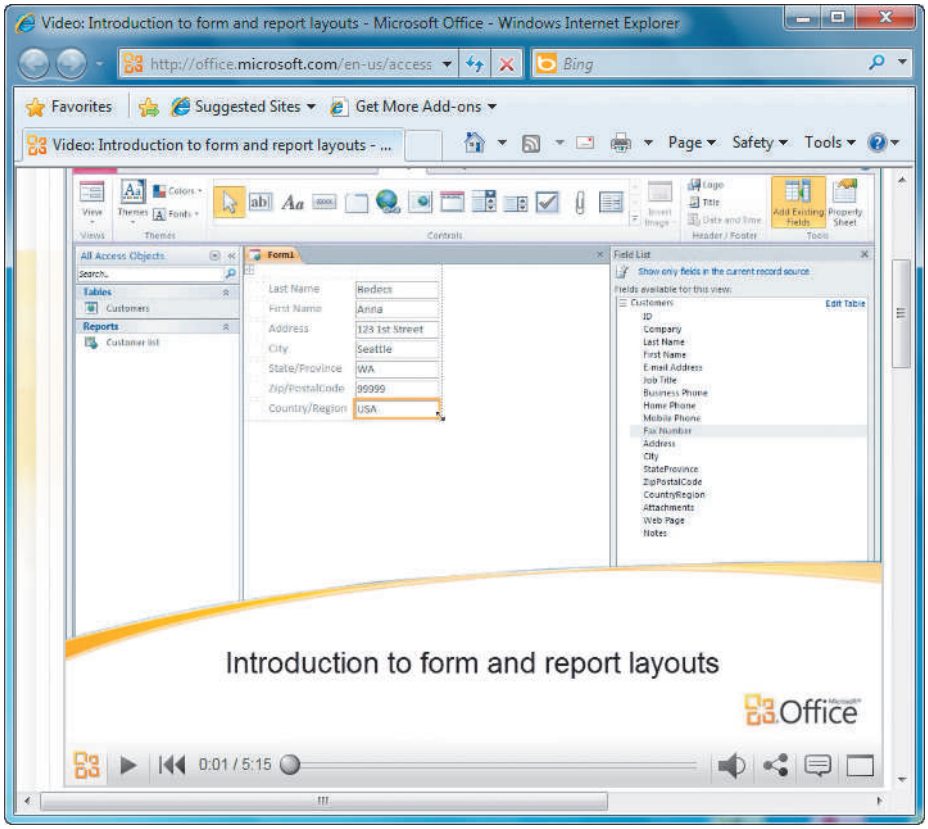


FIGURE 8-19 Microsoft offers suggestions, tips, and a video that can help you design better forms and reports.

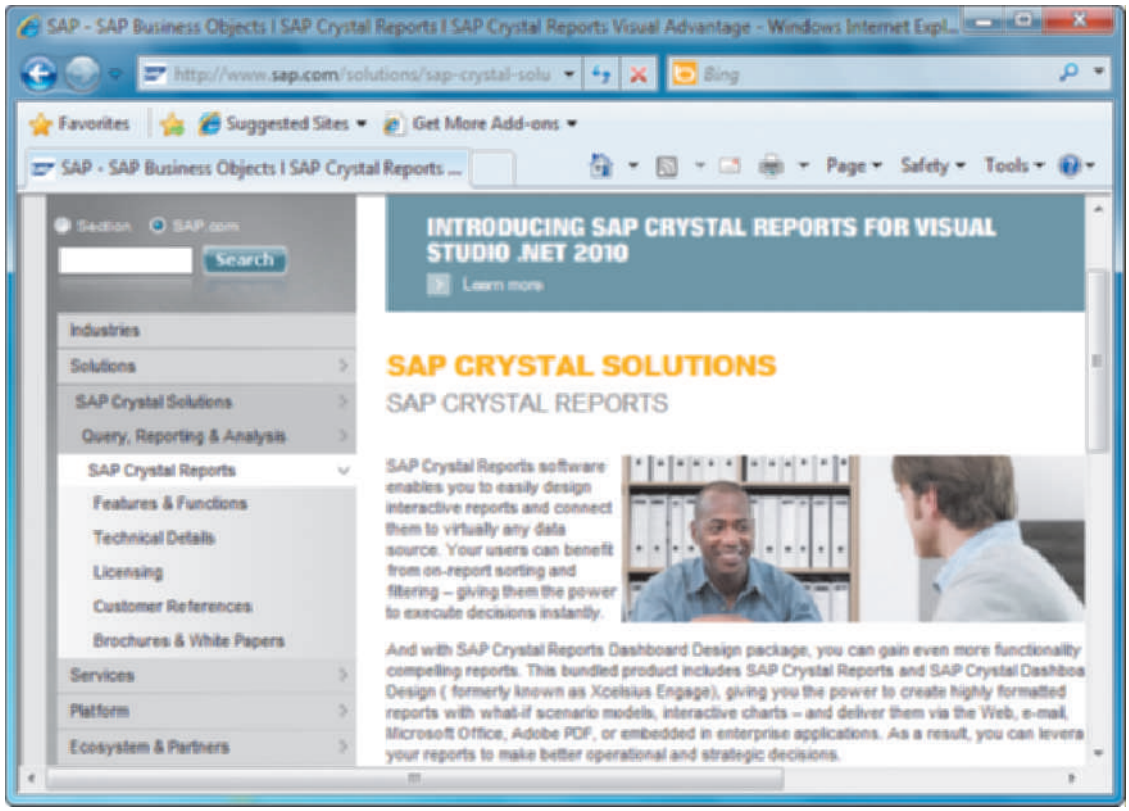


FIGURE 8-20 Crystal Reports is a popular, powerful, report design package.

Although the vast majority of reports are designed graphically, some systems still produce one or more **character-based reports** that use a character set with fixed spacing. Printing character-based reports on high-speed impact printers is a fast, inexpensive method for producing large-scale reports, such as payroll or inventory reports, or registration rosters at a school. This is especially true if multiple copies are required.

Types of Reports

To be useful, a report must include the information that a user needs. From a user's point of view, a report with too little information is of no value. Too much information, however, can make a report confusing and difficult to understand. When designing reports, the essential goal is to match the report to the user's specific information needs. Depending on their job functions, users might need one or more of the reports described in the following sections.

ON THE WEB

To learn more about printed output, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Printed Output link.

DETAIL REPORTS A **detail report** produces one or more lines of output for each record processed. Each line of output printed is called a **detail line**. Figure 8-21 shows a simple detail report of employee hours for a chain of retail stores. Notice that one detail line prints for each employee. All the fields in the record do not have to be printed, nor do the fields have to be printed in the sequence in which they appear in the record. An employee paycheck that has multiple output lines for a single record is another example of a detail report.

Because it contains one or more lines for each record, a detail report can be quite lengthy. Consider, for example, a large auto parts business. If the firm stocks 3,000 parts, then the detail report would include 3,000 detail lines on approximately 50 printed pages. A user who wants to locate any part in short supply has to examine 3,000 detail lines to find the critical items. A much better alternative is to produce an exception report.

Employee Hours
week ending date: 6/24/11

Page 1

Store Number	Employee Name	Position	Regular Hours	Overtime Hours	Total Hours
8	Andres, Marguerite	Clerk	20.0	0.0	20.0
8	Bogema, Michelle	Clerk	12.5	0.0	12.5
8	Davenport, Kim	Asst Mgr	40.0	5.0	45.0
8	Lemka, Susan	Clerk	32.7	0.0	32.7
8	Ramirez, Rudy	Manager	40.0	8.5	48.5
8	Ullery, Ruth	Clerk	20.0	0.0	20.0
17	De Martini, Jennifer	Clerk	40.0	8.4	48.4
17	Haff, Lisa	Manager	40.0	0.0	40.0
17	Rittenbery, Sandra	Clerk	40.0	11.0	51.0
17	Wyer, Elizabeth	Clerk	20.0	0.0	20.0
17	Zeigler, Cecille	Clerk	32.0	0.0	32.0

detail lines

FIGURE 8-21 A detail report with one printed line per employee.

EXCEPTION REPORTS An **exception report** displays only those records that meet a specific condition or conditions. Exception reports are useful when the user wants information only on records that might require action, but does not need to know the details. For example, a credit manager might use an exception report to identify only those customers with past due accounts, or a customer service manager might want a report on all packages that were not delivered within a specified time period. Figure 8-22 shows an exception report that includes information only for those employees who worked overtime, instead of listing information for all employees.

Overtime Report
week ending date: 6/24/11

Page 1

Store Number	Position	Employee Name	Overtime Hours
8	Asst Mgr	Davenport, Kim	5.0
	Manager	Ramirez, Rudy	8.5
		Store 8 totals:	**13.5**
17	Clerk	De Martini, Jennifer	8.4
	Clerk	Rittenbery, Sandra	11.0
		Store 17 totals:	**19.4**
		Grand total:	**32.9**

FIGURE 8-22 An exception report that shows information *only* for employees who worked overtime.

SUMMARY REPORTS Upper-level managers often want to see total figures and do not need supporting details. A sales manager, for example, might want to know total sales for each sales representative, but not want a detail report listing every sale made by them. In that case, a **summary report** is appropriate. Similarly, a personnel manager might need to know the total regular and overtime hours worked by employees in each store but might not be interested in the number of hours worked by each employee. For the personnel manager, a summary report such as the one shown in Figure 8-23 would be useful. Generally, reports used by individuals at higher levels in the organization include less detail than reports used by lower-level employees.

Employee Hours Summary
week ending date: 6/24/11

Page 1

Store Number	Regular Hours	Overtime Hours	Total Hours
8	181.2	13.5	194.7
17	172.0	19.4	191.4
Totals:	**337.2**	**32.9**	**370.1**

FIGURE 8-23 A summary report displays totals without showing details.

User Involvement in Report Design

Printed reports are an important way of delivering information, so users should approve all report designs in advance. The best approach is to submit each design for approval as you complete it, rather than waiting until you finish all report designs.

When designing a report, you should prepare a sample report, called a **mock-up**, or prototype, for users to review. The sample should include typical field values and contain enough records to show all the design features. Depending on the type of printed output, you can create a Word document or use a report generator to create mock-up reports. After a report design is approved, you should document the design by creating **a report analysis form**, which contains information about the layout, fields, frequency, distribution, data security considerations, and other issues.

Report Design Principles

Printed reports must be attractive, professional, and easy to read. For example, a well-designed detail report should provide totals for numeric fields. Notice that the report shown in Figure 8-21 on page 352 lacked subtotals and grand totals for regular hours, overtime hours, and total hours. Figure 8-24 shows the same report with subtotals and grand totals added. In the example, the Store Number field is called a **control field** because it controls the output.

When the value of a control field changes, a **control break** occurs. A control break usually causes specific actions, such as printing subtotals for a group of records. That type of detail report is called a **control break report**. To produce a control break report, the records must be arranged, or sorted, in **control field order**. The sorting can be done by the report program itself, or in a previous procedure.

Good report design requires effort and attention to detail. To produce a well-designed report, the analyst must consider design features such as report headers and footers, page headers and footers, column headings and alignment, column spacing, field order, and grouping of detail lines. The report shown in Figure 8-24 shows examples of these design features.

REPORT HEADERS AND FOOTERS Every report should have a report header and a report footer. The **report header**, which appears at the beginning of the report, identifies the report, and contains the report title, date, and other necessary information. The **report footer**, which appears at the end of the report, can include grand totals for numeric fields and other end-of-report information, as shown in Figure 8-24.

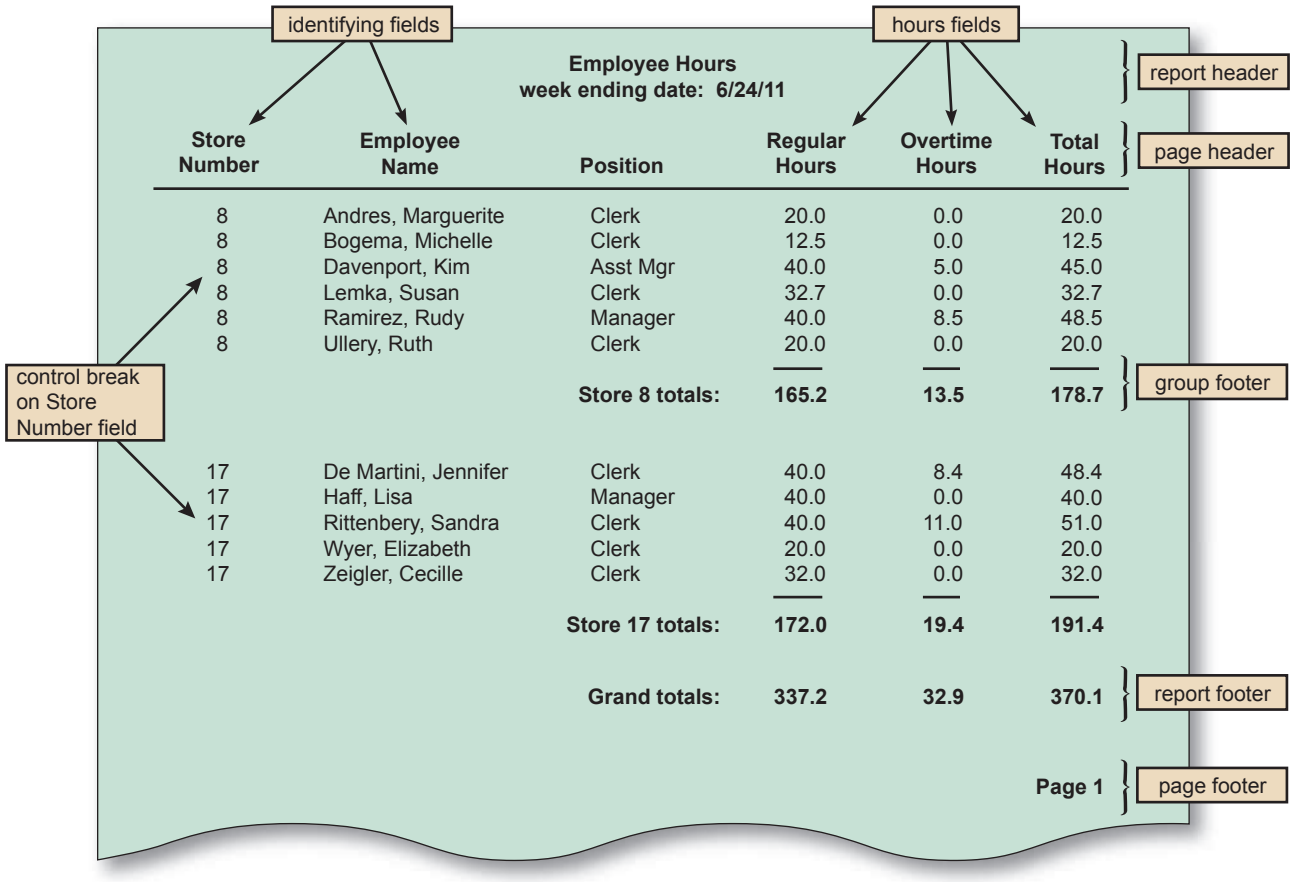


FIGURE 8-24 The Employee Hours report is a detail report with control breaks, subtotals, and grand totals. Notice that a report header identifies the report, a page header contains column headings, a group footer contains subtotals for each store, a report footer contains grand totals, and a page footer identifies the page number.

PAGE HEADERS AND FOOTERS Every page should include a **page header**, which appears at the top of the page and includes the column headings that identify the data. The headings should be short but descriptive. Avoid abbreviations unless you know that users will understand them clearly. Either a page header or a **page footer**, which appears at the bottom of the page, is used to display the report title and the page number.

COLUMN HEADING ALIGNMENT Figure 8-25 shows several column heading alignment options. In Example 1, the left-justified column headings do not work well with numeric fields because the amount 1.25 would print past the right edge of the AMOUNT heading. In Example 2, the right-justified headings cause a problem with alphanumeric fields, because none of the characters in a short name would print under any part of the NAME heading. Centering headings over *maximum* field widths, as shown in Example 3, is not ideal when many of the actual values are shorter than the maximum width. Many designers prefer Example 4, where headings are left-justified over alphanumeric fields and right-justified over numeric fields.

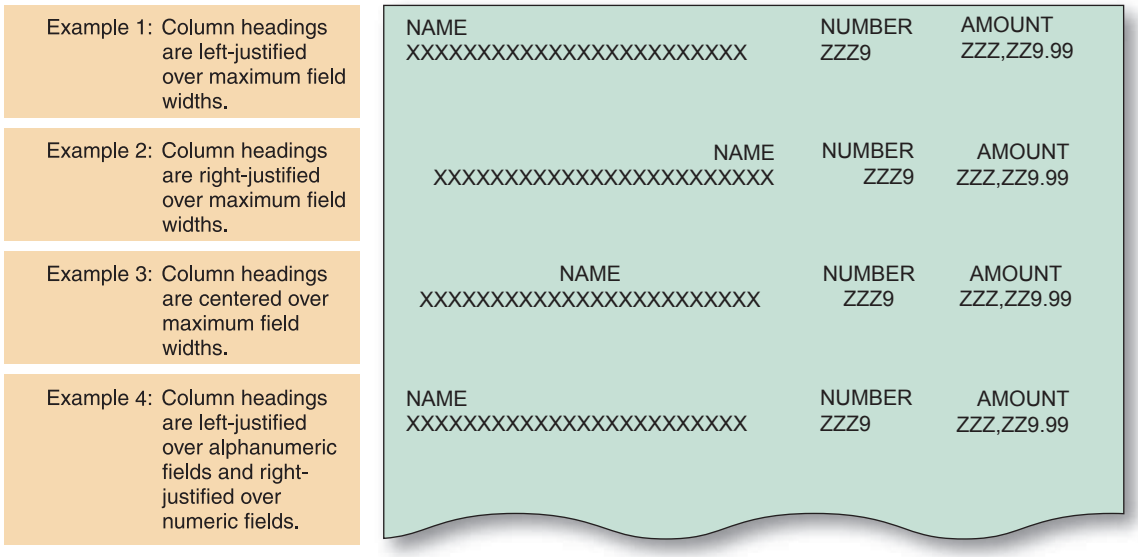


FIGURE 8-25 Four different column heading alignment options.

COLUMN SPACING You should space columns of information carefully. A crowded report is hard to read, and large gaps between columns make it difficult for the eye to follow a line. Columns should stretch across the report, with uniform spacing and suitable margins at top, bottom, right, and left. Some report designers use landscape orientation when working with a large number of columns; others prefer to break the information into more than one report. In some cases, a smaller point size will solve the problem, but the report must remain readable and acceptable to users.

FIELD ORDER Fields should be displayed and grouped in a logical order. For example, the report shown in Figure 8-24 shows the detail lines printed in alphabetical order within store number, so the store number is in the left column, followed by the employee name. The employee position relates to the employee's name, so the items are adjacent. The three hours fields also are placed together.

GROUPING DETAIL LINES Often, it is meaningful to arrange detail lines in groups, based on a control field. For example, using the department number as a control field, individual employees can be grouped by department. You can print a **group header**

above the first detail line and a **group footer** after the last detail line in a group, as shown in Figure 8-24.

Database programs such as Microsoft Access make it easy to create groups and subgroups based on particular fields. You also can have the report calculate and display totals, averages, record counts, and other data for any group or subgroup. For example, in a large company, you might want to see total sales and number of sales broken down by product within each of the 50 states. The screen shown in Figure 8-26 is an Access Help screen that refers to a step-by-step process for creating multilevel grouping.

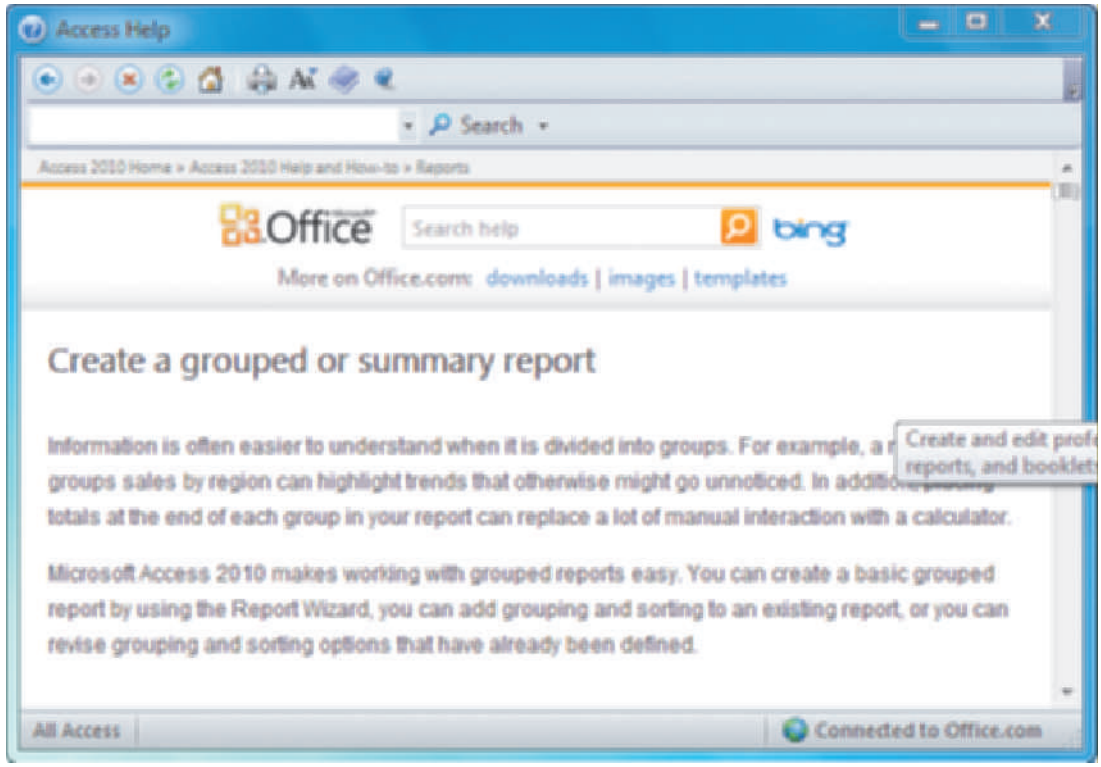


FIGURE 8-26 Microsoft Access includes an easy-to-use tool for grouping data.

REPEATING FIELDS Report design is an art, not a science. User involvement is essential, but users often don't know what they want without seeing samples. For example, consider the issue of repeating fields. The sample report in Figure 8-21 on page 352 repeats the store number on every row. Is that a good thing? The best advice is to ask users what they think, and be guided accordingly. A similar issue exists with regard to the overtime hours column. Is it better to print the zero overtime data, or only print actual hours, so the data stands out clearly? Again, the best answer is usually the one that works best for users.

CONSISTENT DESIGN Look and feel are important to users, so reports should be uniform and consistent. When a system produces multiple reports, each report should share common design elements. For example, the date and page numbers should print in the same place on each report page. Abbreviations used in reports also should be consistent. For example, when indicating a numeric value, it is confusing for one report to use #, another *NO*, and a third *NUM*. Items in a report also should be consistent. If one report displays the inventory location as a shelf number column followed by a bin number column, that same layout should be used on all inventory location reports.

CASE IN POINT 8.3: LAZY EDDIE

Lynn Jennings is the IT manager at Lazy Eddie, a chain that specializes in beanbag chairs and recliners. She asked Jan Lauten, a senior systems analyst, to review the large number of printed reports that are distributed to Lazy Eddie's 35 store managers. "Jan, I just can't believe that our people really read all of those reports," Lynn said. "We constantly add new reports, and we never seem to eliminate the old ones. Sometimes I think all we're doing is keeping the paper companies in business!" Jan replied, "I agree, but what can we do? The managers say they want the reports, but I always see them stacked on top of file cabinets. I've never seen anyone read a report."

"I have an idea," Lynn said. "I want you to come up with a procedure that requires users to review and justify their information needs to see if they really use the reports we send them. You could design a form that asks if the information still is required, and why. Try to get users to decide if a report is worth the cost of producing it. Do you think you can do it?"

"Sure I can," Jan replied. When Jan returned to her office, she wondered where to begin. What advice would you give to Jan?

Output Technology

Although business information systems still provide most output as screen displays and printed matter, technology is having an enormous impact on how people communicate and obtain information. This trend is especially important to firms that use information technology to lower their costs, improve employee productivity, and communicate effectively with their customers.

In addition to screen output and printed matter, output can take many forms. The system requirements document probably identified user output needs. Now, in the systems design phase, you will create the actual forms, reports, documents, and other types of output. During this process, you must consider the format and how it will be delivered, stored, and retrieved. The following sections explain various output types and the technologies that are available to systems developers.

INTERNET-BASED INFORMATION DELIVERY Millions of firms use the Internet to reach new customers and markets around the world. To support the explosive growth in e-commerce, Web designers must provide user-friendly screen interfaces that display output and accept input from customers. For example, a business can link its inventory system to its Web site so the output from the inventory system is displayed as an online catalog. Customers visiting the site can review the items, obtain current prices, and check product availability.

Another example of Web-based output is a system that provides customized responses to product or technical questions. When a user enters a product inquiry or requests technical support, the system responds with appropriate information from an on-site knowledge base. Web-based delivery allows users to download a universe of files and documents to support their information needs. For example, the Web provides consumers with instant access to brochures, product manuals, and parts lists; while prospective home buyers can obtain instant quotes on mortgages, insurance, and other financial services.

To reach prospective customers and investors, companies also use a live or prerecorded **Webcast**, which is an audio or video media file distributed over the Internet. Radio and TV stations also use this technique to broadcast program material to their audiences.

E-MAIL E-mail is an essential means of internal and external business communication. Employees send and receive e-mail on local or wide area networks, including the Internet. Companies send new product information to customers via e-mail, and financial services

ON THE WEB

To learn more about e-mail, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the E-Mail link.

companies use e-mail messages to confirm online stock trades. Employees use e-mail to exchange documents, data, and schedules and to share business-related information they need to perform their jobs. In many firms, e-mail has virtually replaced traditional memos and printed correspondence.

BLOGS Web-based logs, called **blogs**, are another form of Web-based output. Because blogs are journals written from a particular point of view, they not only deliver facts to Web readers, but also provide opinions. Blogs are useful for posting news, reviewing current events, and promoting products.

INSTANT MESSAGING This popular form of communication is another way for individuals and companies to communicate effectively over the Internet. Although some users feel that it can be a distraction, others like the constant flow of communication, especially as a team member in a collaborative situation.

WIRELESS DEVICES Messages and data can be transmitted to a wide array of mobile devices, including PDAs, handheld computers, smart cell phones, and similar wireless products that combine portable computing power, multimedia capability, and Internet access.

DIGITAL AUDIO, IMAGES, AND VIDEO Sounds, images, and video clips can be captured, stored in digital format, and transmitted as output to users who can reproduce the content.

Audio or video output can be attached to an e-mail message or inserted as a clip in a Microsoft Word document, as shown in Figure 8-27. Businesses also use automated systems to handle voice transactions and provide information to customers. For example, using a telephone keypad, a customer can confirm an airline seat assignment, check a credit card balance, or determine the current price of a mutual fund.

If a picture is worth a thousand words, then digital images and video clips certainly are high-value output types that offer a whole new dimension. For example, an insurance adjuster with a digital camera phone can take a picture, submit the image via a wireless device, and receive immediate authorization to pay a claim on the spot. If images are a valuable form of output, video clips are even better in some situations. For example, video clips provide online virtual tours that allow realtors to show off the best features of homes they are marketing. The user can zoom in or out, and rotate the image in any direction.

PODCASTS A **podcast** is a specially formatted digital audio file that can be downloaded by Internet users from a variety of content providers. Many firms use podcasts as sales and marketing tools, and to communicate with their own employees. Using software such as iTunes, you can receive a podcast, launch the file on your computer, and store it on your portable player. Podcasts can include images, sounds, and video.

AUTOMATED FACSIMILE SYSTEMS An **automated facsimile** or **faxback** system allows a customer to request a fax using e-mail, via the company Web site, or by telephone. The response is transmitted in a matter of seconds back to the user's fax machine. Although most users prefer to download documents from the Web, many organizations, including the U.S. Department of Transportation, still offer an automated faxback service as another way to provide immediate response 24 hours a day.

COMPUTER OUTPUT TO MICROFILM (COM) **Computer output to microfilm** (COM) is often used by large firms to scan and store images of original documents to provide high-quality records management and archiving. COM systems are especially important for legal reasons, or where it is necessary to display a signature, date stamp, or other visual features of a document.

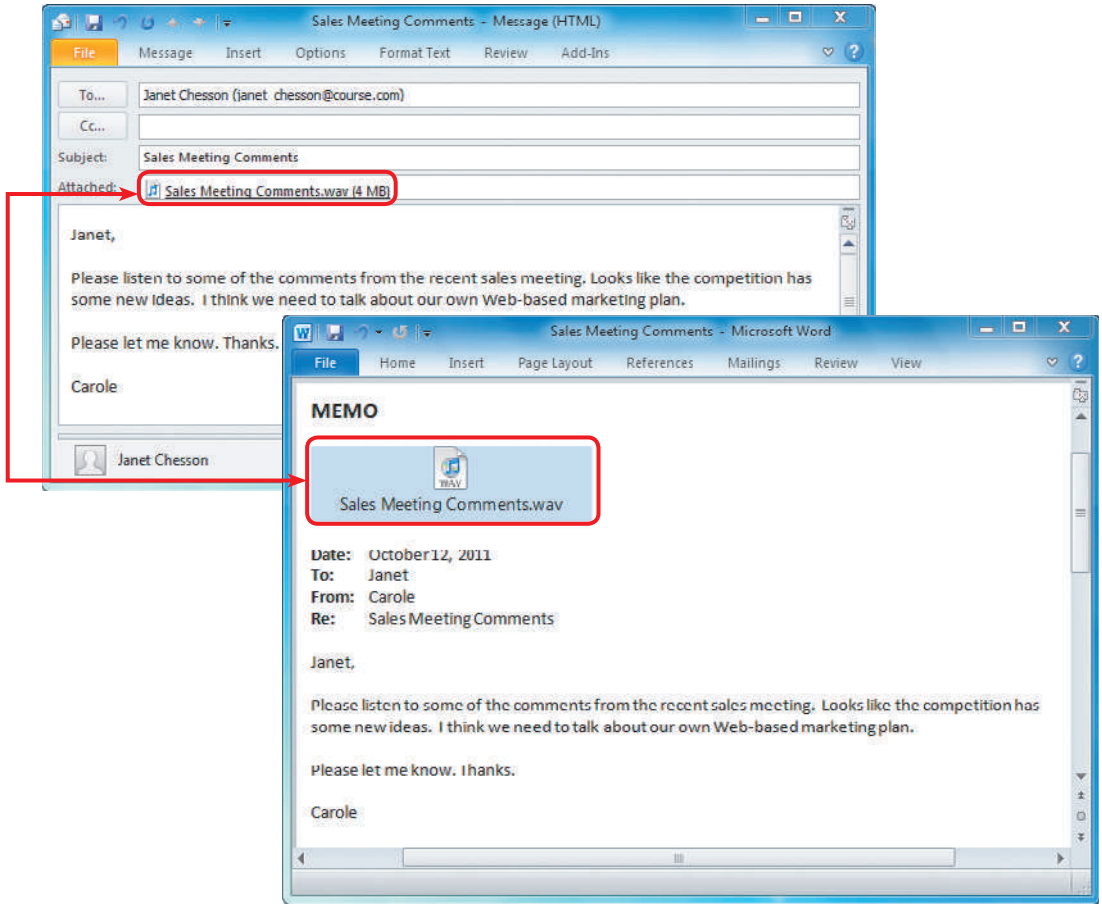


FIGURE 8-27 Audio or video clips can be attached to an e-mail message or inserted in a document. The recipient can double-click the link or icon and a media player opens the file.

COMPUTER OUTPUT TO DIGITAL MEDIA This process is used when many paper documents must be scanned, stored in digital format, and retrieved quickly. For example, if an insurance company stores thousands of paper application forms, special software can treat the documents as data and extract information from a particular column or area on the form. Digital storage media can include magnetic tape, CDs, DVDs, and high-density laser disks.

SPECIALIZED FORMS OF OUTPUT An incredibly diverse marketplace requires many forms of specialized output and devices such as these:

- Portable, Web-connected devices that can run applications, handle multimedia output, and provide powerful, multipurpose communication for users
- Retail point-of-sale terminals that handle computer-based credit card transactions, print receipts, and update inventory records
- Automatic teller machines (ATMs) that can process bank transactions and print deposit and withdrawal slips
- Special-purpose printers that can produce labels, employee ID cards, driver's licenses, gasoline pump receipts, and, in some states, lottery tickets
- Plotters that can produce high-quality images such as blueprints, maps, and electronic circuit diagrams
- Electronic detection of data embedded in credit cards, bank cards, and employee identification cards

Input Design

No matter how data enters an information system, the quality of the output is only as good as the quality of the input. The term **garbage in, garbage out** (GIGO), is familiar to IT professionals, who know that the best time to avoid problems is when the data is entered. The objective of input design is to ensure the quality, accuracy, and timeliness of input data.

Good input design requires attention to human factors as well as technology issues. This section includes a discussion of source documents, data entry screen design, input masks, and data validation rules. The final topic, input technology, examines devices and techniques that can speed up the input process, reduce costs, and handle new forms of data.

Source Documents and Forms

A **source document** collects input data, triggers or authorizes an input action, and provides a record of the original transaction. During the input design stage, you develop source documents that are easy to complete and use for data entry. Source documents generally are paper-based, but also can be provided online. Either way, the design considerations are the same.

Consider a time when you struggled to complete a poorly designed form. You might have encountered insufficient space, confusing instructions, or poor organization, all symptoms of incorrect **form layout**. Good form layout makes the form easy to complete and provides enough space, both vertically and horizontally, for users to enter the data. A form should indicate data entry positions clearly using blank lines or boxes and descriptive captions. Figure 8-28 shows several techniques for using line and boxed captions in source documents, and an example of check boxes, which are effective when a user must select choices from a list.

The placement of information on a form also is important. Source documents typically include most of the zones shown in Figure 8-29. The **heading zone** usually contains the company name or logo and the title and number of the form. The **control zone** contains codes, identification information, numbers, and dates that are used for storing completed forms.

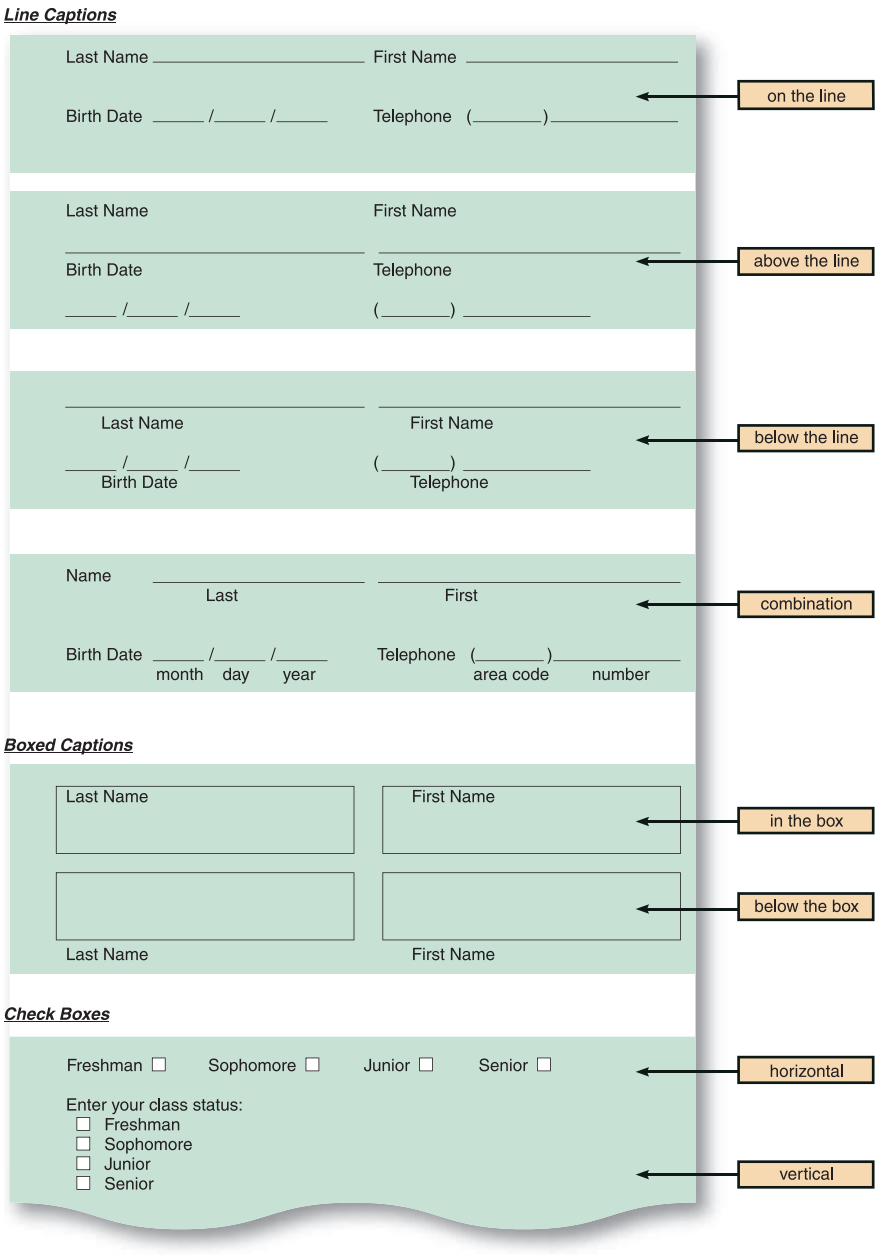


FIGURE 8-28 Examples of caption techniques for source documents.

The **instruction zone** contains instructions for completing the form. The main part of the form, called the **body zone**, usually takes up at least half of the space on the form and contains captions and areas for entering variable data. If totals are included on the form, they appear in the **totals zone**. Finally, the **authorization zone** contains any required signatures.

Information should flow on a form from left to right and top to bottom to match the way users read documents naturally. That layout makes the form easy to use for the individual who completes the form, and for users who enter data into the system using the completed form. You can review samples of source document design that appear in the SWL case study on pages 382–387.

The same user-friendly design principles also apply to printed forms such as invoices and monthly statements, except that heading information usually is preprinted. You should make column headings short but descriptive, avoid nonstandard abbreviations, and use reasonable spacing between columns for better readability.

The order and placement of printed fields should be logical, and totals should be identified clearly. When designing a preprinted form, you should contact the form's vendor for advice on paper sizes, type styles and sizes, paper and ink colors, field placement, and other important form details. Your goal is to design a form that is attractive, readable, and effective.

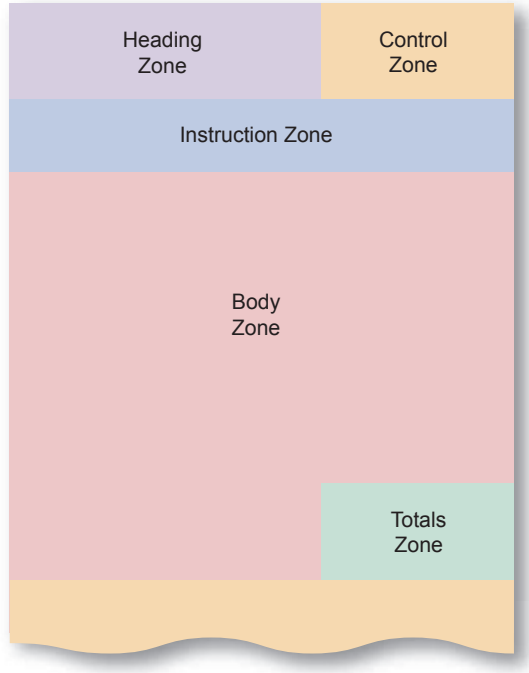


FIGURE 8-29 Source document zones.

Layout and design also are important on Web-based forms, and you can find many resources that will help you design efficient, user-friendly forms. For example, Figure 8-30 describes a book by Luke Wroblewski, a well-known author and consultant. His Web site offers valuable suggestions, guidelines, and examples.

A major challenge of Web-based form design is that most people read and interact differently with on-screen information compared to paper forms. In the view of Dr. Jakob Nielsen, a pioneer in Web usability design, users simply do not read on the Web,

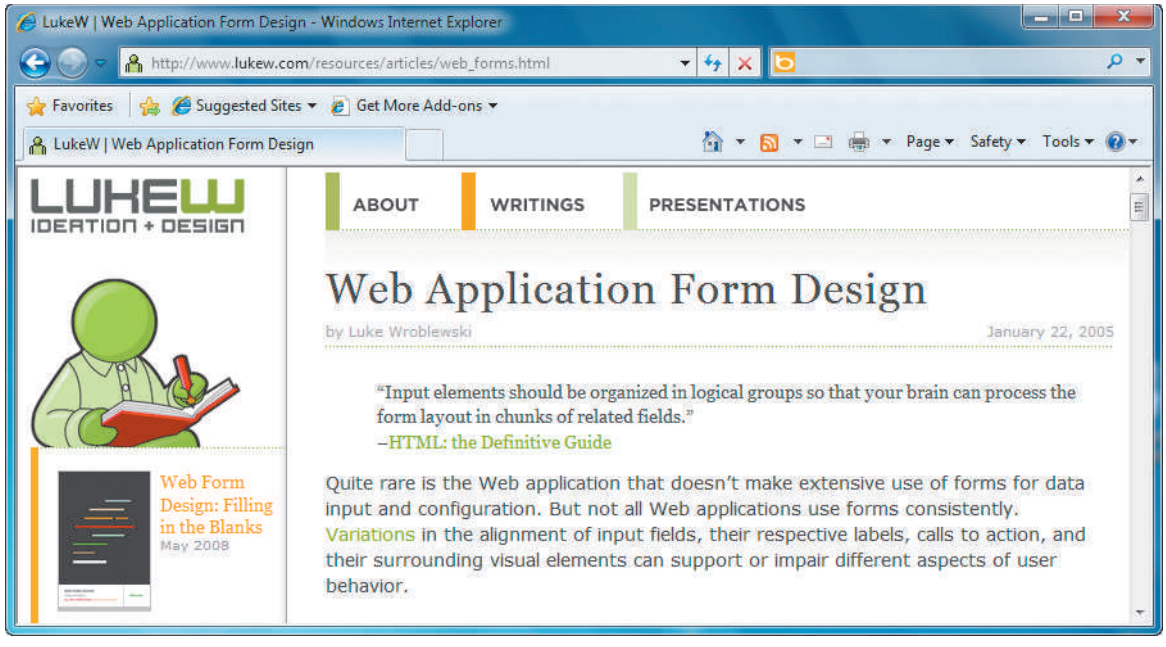


FIGURE 8-30 Luke Wroblewski's Web site is a good source of information about form design.

as shown in Figure 8-31. Dr. Nielsen believes that users *scan* a page, picking out individual words and sentences. As a result, Web designers must use **scannable text** to capture and hold a user's attention. On his site, Dr. Nielson offers several suggestions for creating scannable text. Also notice that Dr. Nielsen employs various usability metrics to measure user responses, comprehension, and recall.

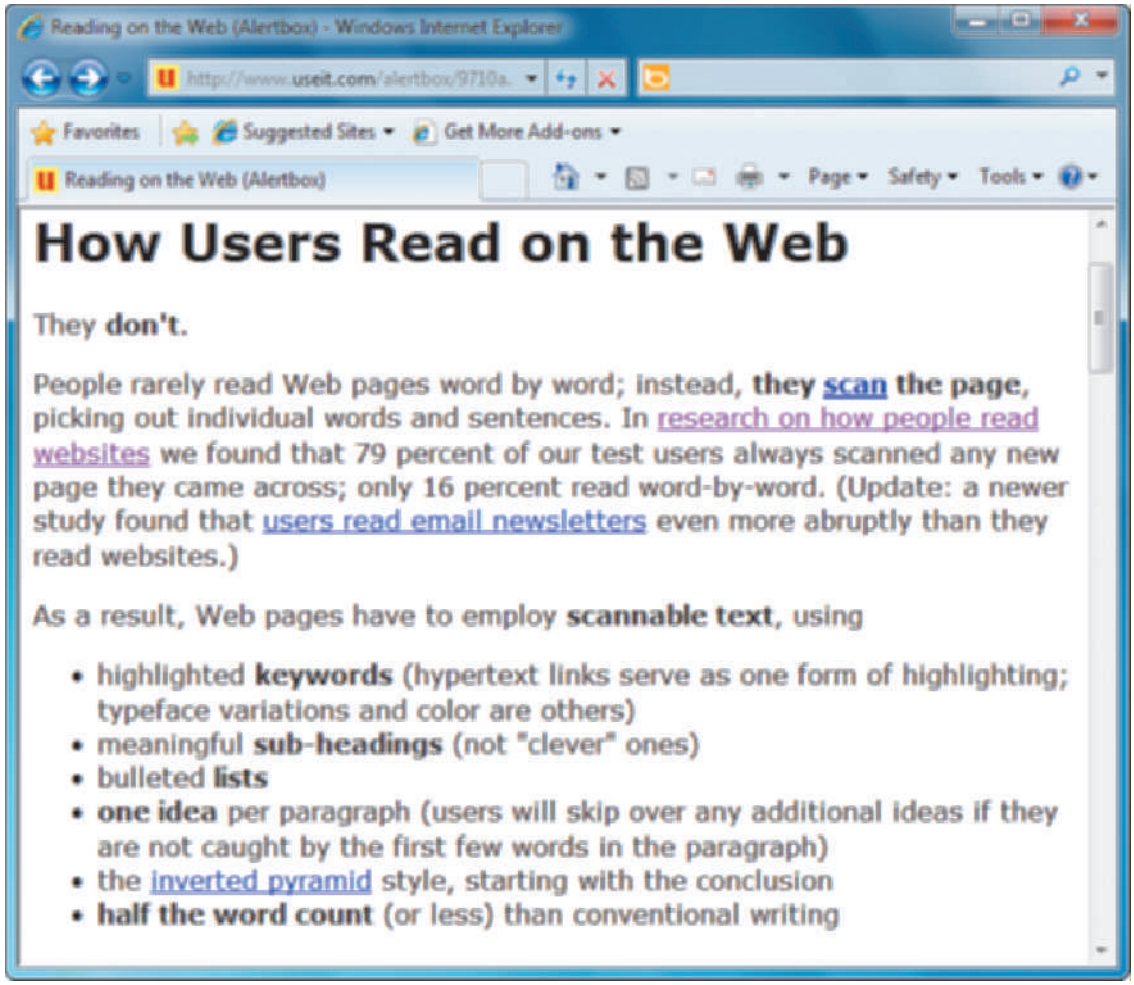


FIGURE 8-31 Dr. Jakob Nielsen believes that users scan Web material rather than reading it. He suggests that Web designers must use scannable text and employ usability metrics to measure the results.

CASE IN POINT 8.4: TRUSTWORTHY INSURANCE COMPANY

Trustworthy Insurance maintains its headquarters in a large Midwestern city. Right now, a debate is raging in the IT department. Several analysts want to use standard, company-wide desktop screen layouts and icons. Others want to allow users to set up their screens any way they choose. Those who argue for standardization point out that Trustworthy employs a number of part-time employees, who fill in for employees on vacation. Without a standard interface, these people would have to reorient to every workstation, and the proponents of standardization claim that would reduce productivity and increase costs. Those opposed to standardization believe that employees are most productive when they have control over their workplace, including the ability to design an interface they feel is attractive, even if no one else does.

You are on a committee that was charged with resolving this issue, and yours is the tie-breaking vote. What will you decide, and why?

Data Entry Screens

ON THE WEB

To learn more about data entry, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Data Entry link.

During input design, you determine how data will be captured and entered into the system. **Data capture** uses an automated or manually operated device to identify source data and convert it into computer-readable form. Examples of data capture devices include credit card scanners and bar code readers. **Data entry** is the process of manually entering data into the information system, usually in the form of keystrokes, mouse clicks, touch screens, or spoken words.

Some users work with many features of the user interface; others spend most of their time entering data. This section discusses design guidelines and concepts that primarily relate to repetitive data entry. Notice that many of the guidelines are based on general principles of interface design discussed in this chapter.

The most effective method of online data entry is **form filling**, in which a blank form that duplicates or resembles the source document is completed on the screen. The user enters the data and then moves to the next field. The following guidelines will help you design data entry screens that are easy to learn and use.

1. Restrict user access to screen locations where data is entered. For example, when the screen in Figure 8-32 appears, the system should position the insertion point in the first data entry location. After the operator enters a Customer ID, the insertion point should move automatically to the entry location for the next field (Item). A user should be able to position the insertion point only in places where data is entered on the form.

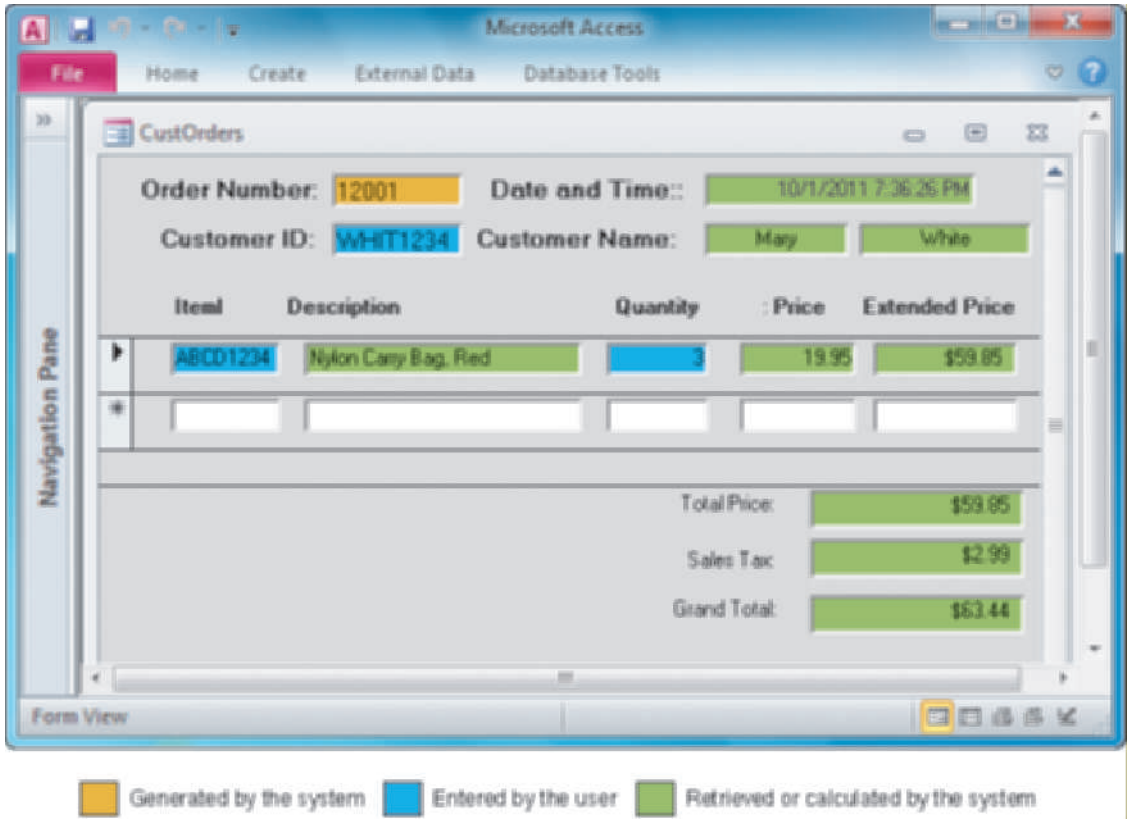


FIGURE 8-32 In this data screen for customer orders, the system generates an order number and logs the current date and time. The user enters a customer ID. If the entry is valid, the system displays the customer name so the user can verify it. The user then enters the item and quantity. Note that the description, price, extended price, total price, sales tax, and grand total are retrieved automatically or calculated by the system.

2. Provide a descriptive caption for every field, and show the user where to enter the data and the required or maximum field size. Typically, white boxes show the location and length of each field. Other methods used to indicate field locations are video highlighting, underscores, special symbols, or a combination of these features.
3. Display a sample format if a user must enter values in a field in a specific format. For example, provide an on-screen instruction to let users know that the date format is MMDDYY, and provide an example if the user must enter **separators**, such as slashes. It is better to use an input mask, so users simply can enter 112711 to represent November 27, 2011.
4. Require an ending keystroke for every field. Pressing the ENTER key or the TAB key should signify the end of a field entry. Avoid a design that moves automatically to the next item when the field is full. The latter approach requires an ending keystroke *only* when the data entered is less than the maximum field length. It is confusing to use two different data entry procedures.
5. Do not require users to type leading zeroes for numeric fields. For example, if a three-digit project number is 045, the operator should be able to type 45 instead of 045 before pressing the ENTER key. An exception to that rule might occur when entering a date, where a leading zero is needed to identify single-digit months or days, such as 06-04-2011.
6. Do not require users to type trailing zeroes for numbers that include decimals. For example, when a user types a value of 98, the system should interpret the value as 98.00 if the field has been formatted to include numbers with two decimal places. The decimal point is needed *only* to indicate nonzero decimal places, such as 98.76.
7. Display default values so operators can press the ENTER key to accept the suggested value. If the default value is not appropriate, the operator can change it.
8. Use a default value when a field value will be constant for successive records or throughout the data entry session. For example, if records are input in order by date, the date used in the first transaction should be used as the default date until a new date is entered, at which time the new date becomes the default value.
9. Display a list of acceptable values for fields, and provide meaningful error messages if the user enters an unacceptable value. An even better method, which was described in the user interface design section, is to provide a drop-down list box containing acceptable values that allows the user to select a value by clicking.
10. Provide a way to leave the data entry screen at any time without entering the current record. This feature is available in the screen shown in Figure 8-33, which is an enhanced version of the data entry screen shown in Figure 8-32 on the previous page. Notice that the new version has command buttons that provide flexibility and allow the user to perform various functions. For example, clicking the Cancel Order Without Entering button cancels the current order and moves the insertion point back to the beginning of the form.
11. Provide users with an opportunity to confirm the accuracy of input data before entering it by displaying a message such as, *Add this record? (Y/N)*. A positive response *(Y)* adds the record, clears the entry fields, and positions the insertion point in the first field so the user can input another record. If the response is negative (*N*), the current record is not added and the user can correct the errors.
12. Provide a means for users to move among fields on the form in a standard order or in any order they choose. For example, when a user opens the form shown in Figure 8-33, the insertion point automatically will be in the first field. After the user fills in each field and confirms the entry, the insertion point moves to the next field, in a

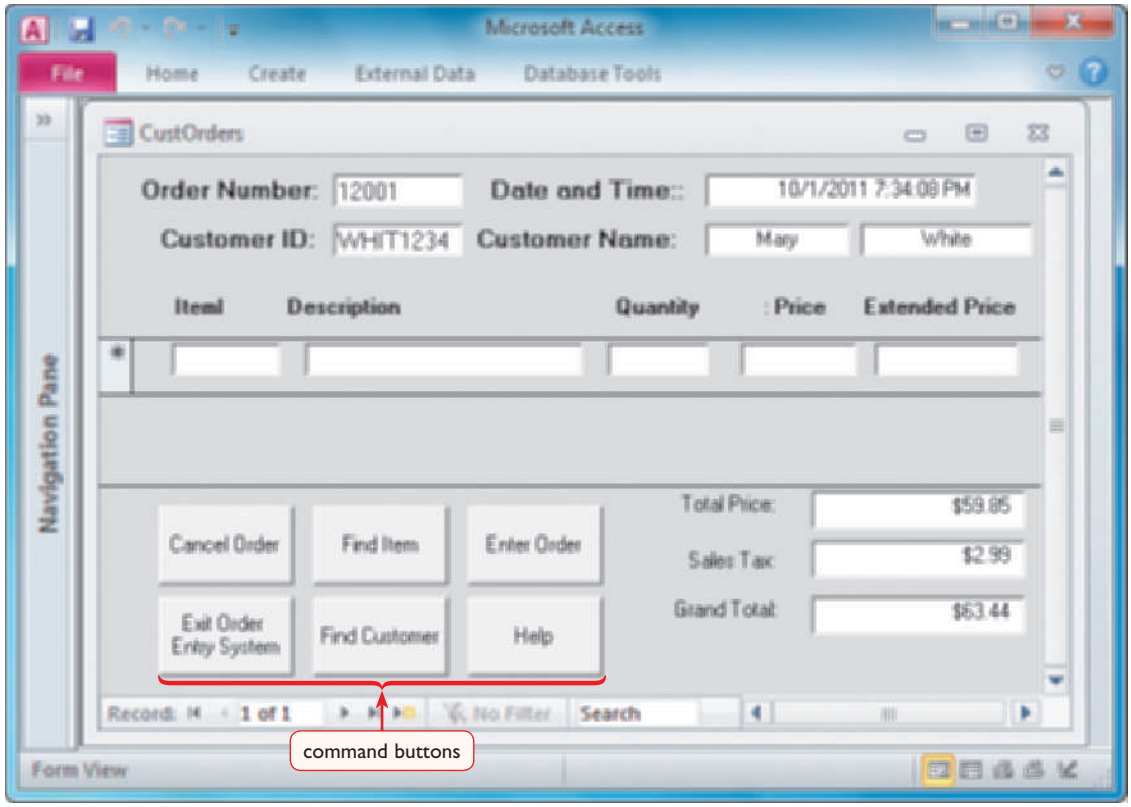


FIGURE 8-33 This is an enhanced version of the data entry screen shown in Figure 8-32. The new version has command buttons that allow the user to perform various functions.

predetermined order. In a graphical user interface (GUI), the user can override the standard field order and select field locations using the mouse or arrow keys.

13. Design the screen form layout to match the layout of the source document. If the source document fields start at the top of the form and run down in a column, the input screen should use the same design.
14. Allow users to add, change, delete, and view records. Figure 8-33 shows a screen that can be used for entering orders, finding items, and finding customers. After the operator enters a customer identification code, the order form displays current values for all appropriate fields. Then the operator can view the data, make changes, enter the order, or cancel without ordering. Messages such as: *Apply these changes? (Y/N)* or *Delete this record? (Y/N)* should require users to confirm the actions. Highlighting the letter *N* as a default response will avoid problems if the user presses the ENTER key by mistake.
15. Provide a method to allow users to search for specific information, as shown in Figure 8-33.

Input Masks

Use **input masks,** which are templates or patterns that restrict data entry and prevent errors. Microsoft Access 2010 provides standard input masks for fields such as dates, telephone numbers, postal codes, and Social Security numbers. In addition, you can create custom input masks for any type of data, as shown in Figure 8-34 on the next page. Notice that a mask can manipulate the input data and apply a specific format. For example, if a user enters text in lowercase letters, the input mask >L<??????????? will automatically capitalize the first letter.

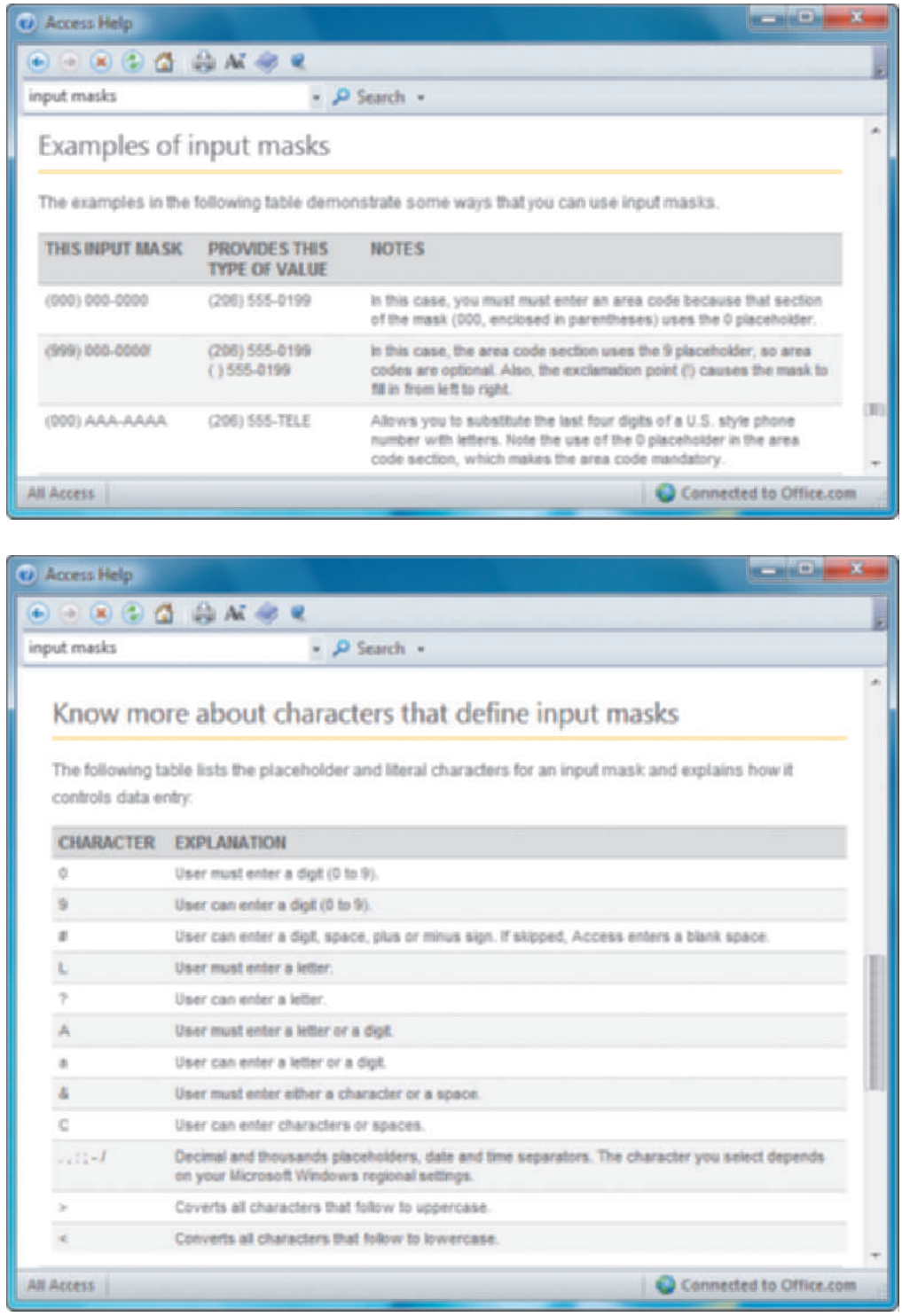


FIGURE 8-34 Microsoft Access 2010 provides various input masks for dates, phone numbers, and postal codes, among others. In addition, it is easy to create a custom mask using the characters shown here.

Validation Rules

Reducing the number of input errors improves data quality. One way to reduce input errors is to eliminate unnecessary data entry. For example, a user cannot misspell a customer name if it is not entered, or is entered automatically based on the user entering the customer ID. Similarly, an outdated item price cannot be used if the item price is retrieved from a master file instead of being entered manually.

The best defense against incorrect data is to identify and correct errors before they enter the system by using data validation rules, as shown in Figure 8-35. A **data validation rule** improves input quality by testing the data and rejecting any entry that fails to meet specified conditions. You can design at least eight types of data validation rules. For example:

1. A **sequence check** is used when the data must be in some predetermined sequence. If the user must enter work orders in numerical sequence, for example, then an out-of-sequence order number indicates an error, or if the user must enter transactions chronologically, then a transaction with an out-of-sequence date indicates an error.
2. An **existence check** is used for mandatory data items. For example, if an employee record requires a Social Security number, an existence check would not allow the user to save the record until he or she enters a suitable value in the Social Security number field.
3. A **data type check** tests to ensure that a data item fits the required data type. For example, a numeric field must have only numbers or numeric symbols, and an alphabetic field can contain only the characters A through Z (or a through z).
4. A **range check** tests data items to verify that they fall between a specified minimum and maximum value. The daily hours worked by an employee, for example, must fall within the range of 0 to 24. When the validation check involves a minimum or a maximum value, but not both, it is called a **limit check**. Checking that a payment amount is greater than zero, but not specifying a maximum value, is an example of a limit check.
5. A **reasonableness check** identifies values that are questionable, but not necessarily wrong. For example, input payment values of $.05 and $5,000,000.00 both pass a simple limit check for a payment value greater than zero, and yet both values could be errors. Similarly, a daily hours worked value of 24 passes a 0 to 24 range check; however, the value seems unusual, and the system should verify it using a reasonableness check.
6. A **validity check** is used for data items that must have certain values. For example, if an inventory system has 20 valid item classes, then any input item that does not match one of the valid classes will fail the check. Verifying that a customer number on an order matches a customer number in the customer file is another type of validity check. Because the value entered must refer to another value, that type of check also is called **referential integrity**, which is explained in Chapter 9, Data Design. Another validity check might verify that a new customer number does *not* match a number already stored in the customer master file.

FIGURE 8-35 Validation rules can improve data quality by requiring the input to meet specific requirements or conditions.

7. A **combination check** is performed on two or more fields to ensure that they are consistent or reasonable when considered together. Even though all the fields involved in a combination check might pass their individual validation checks, the combination of the field values might be inconsistent or unreasonable. For example, if an order input for 30 units of a particular item has an input discount rate applicable only for purchases of 100 or more units, then the combination is invalid; either the input order quantity or the input discount rate is incorrect.
8. **Batch controls** are totals used to verify batch input. Batch controls might check data items such as record counts and numeric field totals. For example, before entering a batch of orders, a user might calculate the total number of orders and the sum of all the order quantities. When the batch of orders is entered, the order system also calculates the same two totals. If the system totals do not match the input totals, then a data entry error has occurred. Unlike the other validation checks, batch controls do not identify specific errors. For example, if the sum of all the order quantities does not match the batch control total, you know only that one or more orders in that batch were entered incorrectly or not input. The batch control totals often are called **hash totals**, because they are not meaningful numbers themselves, but are useful for comparison purposes.

ON THE WEB

To learn more about input devices, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Input Devices link.

Input Technology

Input technology has changed dramatically in recent years. In addition to traditional devices and methods, there has been a rapid expansion of new hardware and ways to capture and enter data into a system, some of which are shown in Figure 8-36. Businesses are using the new technology to speed up the input process, reduce costs, and capture data in new forms, such as the digital signature shown in Figure 8-37.

The following sections discuss input and data entry methods, and the impact of input volume reduction.

Input methods should be cost-efficient, timely, and as simple as possible. Systems analysts study transactions and business operations to determine how and when data should enter the system. Usually, the first decision is whether to use batch or online input methods. Each method has advantages and disadvantages, and the systems analyst must consider the following factors.

BATCH INPUT Using **batch input**, data entry usually is performed on a specified time schedule, such as daily, weekly, monthly, or longer. For example, batch input occurs when a payroll department collects time cards at the end of the week and enters the

INPUT TECHNOLOGY		
Traditional	**Evolving**	**Emerging**
Keyboard	Body motion detection	Brain-Computer Interface (BCI)
Mouse	Advanced voice recognition	Neural networks
Pointing devices	Biological feedback	Artificial intelligence (AI)
Microphone	Embedded magnetic data	Advanced motion sensors
OCR (optical character recognition)	RFID	Two-way satellite interface
MICR (magnetic ink character recognition)	Advanced optical recognition	Virtual environments
Graphic input devices	Physical adaptation devices	3-D technology

FIGURE 8-36 Input devices can be very traditional, or based on the latest technology.

FIGURE 8-37 When a customer's signature is stored in digital form, it becomes input to the information system.

data as a **batch**. Another example is a school that enters all grades for the academic term in a batch.

ONLINE INPUT Although batch input is used in specific situations, most business activity requires **online data entry**. The online method offers major advantages, including the immediate validation and availability of data. A popular online input method is **source data automation**, which combines online data entry and automated data capture using input devices such as **RFID tags** or **magnetic data strips**. Source data automation is fast and accurate, and minimizes human involvement in the translation process.

Many large companies use a combination of source data automation and a powerful communication network to manage global operations instantly. Some common examples of source data automation are:

- Businesses that use point-of-sale (POS) terminals equipped with bar code scanners and magnetic swipe scanners to input credit card data.
- Automatic teller machines (ATMs) that read data strips on bank cards.
- Factory employees who use magnetic ID cards to clock on and off specific jobs so the company can track production costs accurately.
- Hospitals that imprint bar codes on patient identification bracelets and use portable scanners when gathering data on patient treatment and medication.
- Retail stores that use portable bar code scanners to log new shipments and update inventory data.
- Libraries that use handheld scanners to read optical strips on books.

TRADEOFFS Although online input offers many advantages, it does have some disadvantages. For example, unless source data automation is used, manual data entry is slower and more expensive than batch input because it is performed at the time the transaction occurs and often done when computer demand is at its highest.

The decision to use batch or online input depends on business requirements. For example, hotel reservations must be entered and processed immediately, but hotels can enter their monthly performance figures in a batch. In fact, some input occurs naturally in batches. A cable TV provider, for example, receives customer payments in batches when the mail arrives.

Input Volume Reduction

To reduce input volume, you must reduce the number of data items required for each transaction. Data capture and data entry require time and effort, so when you reduce input volume, you avoid unnecessary labor costs, get the data into the system more quickly, and decrease the number of errors. The following guidelines will help reduce input volume:

1. Input necessary data only. Do not input a data item unless it is needed by the system. A completed order form, for example, might contain the name of the clerk who took the order. If that data is not needed by the system, the user should not enter it.
2. Do not input data that the user can retrieve from system files or calculate from other data. In the order system example shown in Figure 8-33 on page 365, the system generates an order number and logs the current date and time. Then the user enters a customer ID. If the entry is valid, the system displays the customer name so the user can verify it. The user then enters the item and quantity. Note that the description, price, extended price, total price, sales tax, and grand total are retrieved automatically or calculated by the system.
3. Do not input constant data. If orders are in batches with the same date, then a user should enter the order date only once for the first order in the batch. If orders are entered online, then the user can retrieve the order date automatically using the current system date.
4. Use codes. Codes are shorter than the data they represent, and coded input can reduce data entry time. You will learn more about various types of codes in Chapter 9, Data Design.

Security and Control Issues

A company must do everything in its power to protect its data. This includes not only the firm's own information, but that of its customers, employees, and suppliers. Most assets have a value, but corporate data is priceless, because without safe, secure, accurate data, a company cannot function.

The following sections discuss output and input data security and control.

Output Security and Control

Output must be accurate, complete, current, and secure. Companies use various **output control** methods to maintain output integrity and security. For example, every report should include an appropriate title, report number or code, printing date, and time period covered. Reports should have pages that are numbered consecutively, identified as *Page nn of nn*, and the end of the report should be labeled clearly. Control totals and record counts should be reconciled against input totals and counts. Reports should be selected at random for a thorough check of correctness and completeness. All processing errors or interruptions must be logged so they can be analyzed.

Output security protects privacy rights and shields the organization's proprietary data from theft or unauthorized access. To ensure output security, you must perform several important tasks. First, limit the number of printed copies and use a tracking procedure to account for each copy. When printed output is distributed from a central location, you should use specific procedures to ensure that the output is delivered to authorized recipients only. That is especially true when reports contain sensitive information, such as payroll data. All sensitive reports should be stored in secure areas. All pages of confidential reports should be labeled appropriately.

ON THE WEB

To learn more about output control and security, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Output Control and Security link.

As shown in Figure 8-38, it is important to shred sensitive reports, out-of-date reports, and output from aborted print runs. Blank check forms must be stored in a secure location and be inventoried regularly to verify that no forms are missing. If signature stamps are used, they must be stored in a secure location away from the forms storage location.

In most organizations, the IT department is responsible for output control and security measures. Systems analysts must be concerned with security issues as they design, implement, and support information systems. Whenever possible, security should be designed into the system by using passwords, shielding sensitive data, and controlling user access. Physical security always will be necessary, especially in the case of printed output that is tangible and can be viewed and handled easily.

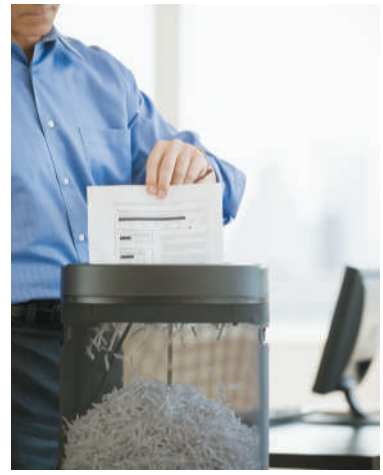


FIGURE 8-38 To maintain output security, it is important to shred sensitive material.

Enterprise-wide data access creates a whole new set of security and control issues. Many firms have responded to those concerns by installing diskless workstations. A **diskless workstation** is a network terminal that supports a full-featured user interface, but limits the printing or copying of data, except to certain network resources that can be monitored and controlled. This concept worked well with terminals that had limited hardware and software features.

However, over time, the number of removable media devices has expanded greatly, along with a wide variety of physical interfaces such as USB, FireWire, and PCMCIA, as well as wireless interfaces such as Wi-Fi and Bluetooth. A popular security solution is the use of a network-based application, often called a **port protector**, that controls access to and from workstation interfaces. The SafeGuard® PortProtector is shown in Figure 8-39.

Input Security and Control

Input control includes the necessary measures to ensure that input data is correct, complete, and secure. You must focus on input control during every phase of input design, starting with source documents that promote data accuracy and quality. When a batch input method is used, the computer can produce an input log file that identifies and documents the data entered.

Every piece of information should be traceable back to the input data that produced it. That means that you must provide an **audit trail** that records the source of each data item and when it entered the system. In addition to recording the original source, an audit trail must show how and when data is accessed or changed, and by whom. All those actions must be logged in an audit trail file and monitored carefully.

A company must have procedures for handling source documents to ensure that data is not lost before it enters the system. All source documents that originate from outside the organization should be logged when they are received. Whenever source documents pass between departments, the transfer should be recorded.

Data security policies and procedures protect data from loss or damage, which is a vital goal in every organization. If the safeguards

FIGURE 8-39 In addition to being diskless, like the lab terminal shown here, a workstation should have a port protector to control input and output access.

are not 100% effective, data recovery utilities should be able to restore lost or damaged data. Once data is entered, the company should store source documents in a safe location for some specified length of time. The company should have a **records retention policy** that meets all legal requirements and business needs.

Audit trail files and reports should be stored and saved. Then, if a data file is damaged, you can use the information to reconstruct the lost data. Data security also involves protecting data from unauthorized access. System sign-on procedures should prevent unauthorized individuals from entering the system, and users should change their passwords regularly. Having several levels of access also is advisable. For example, a data entry person might be allowed to *view* a credit limit, but not *change* it. Sensitive data can be **encrypted**, or coded, in a process called **encryption**, so only users with decoding software can read it.

A QUESTION OF ETHICS

Jacob thought that he did a good job of designing the company's tech support Web page, but Emily, his supervisor, isn't so sure. She is concerned that Jacob's design is very similar to a page used by the company's major competitor, and she asked him whether he had used any HTML code from that site in his design. Although Jacob didn't copy any of the code, he did examine it in his Web browser to see how they handled some design issues.

Emily asked Jacob to investigate Web page copyright issues, and report back to her. In his research, he learned that outright copying would be a copyright violation, but merely viewing other sites to get design ideas would be permissible. What is not so clear is the gray area in the middle. Jacob asked you, as a friend, for your opinion on this question: Even if no actual copying is involved, are there ethical constraints on how far you should go in using the creative work of others? How would you answer Jacob?

Chapter Summary

The purpose of systems design is to create a physical model of the system that satisfies the design requirements that were defined during the systems analysis phase. The chapter began with a discussion of user interface design and human-computer interaction (HCI) concepts. A graphical user interface (GUI) uses visual objects and techniques that allow users to communicate effectively with the system. User-centered design principles include: understanding the business, maximizing graphic effectiveness, thinking like a user, using models and prototypes, focusing on usability, inviting feedback, and documenting everything.

When you design the interface itself, you should try to make it transparent; create an interface that is easy to learn and use; enhance user productivity; make it easy to obtain help or correct errors; minimize input data problems; provide feedback; create an attractive layout and design; and use familiar terms and images. You also can add control features, such as menu bars, toolbars, drop-down list boxes, dialog boxes, toggle buttons, list boxes, option buttons, check boxes, and command buttons. Controls are placed on a main switchboard, which is like a graphical version of a main menu.

The chapter described various types of printed reports, including detail, exception, and summary reports. You learned about the features and sections of reports, including control fields, control breaks, report headers and footers, page headers and footers, and

group headers and footers. You also learned about other types of output, such as Web-based information delivery, audio output, instant messaging, podcasts, e-mail, and other specialized forms of output.

The discussion of input design began with a description of source documents and the various zones in a document, including the heading zone, the control zone, the instruction zone, the body zone, the totals zone, and the authorization zone. The discussion of data entry screen design explained the use of input masks and validation rules to reduce data errors. Input masks are like templates that only permit certain combinations of characters, and data validation rules can provide checks to ensure that inappropriate data is prevented from entering the system. These checks can include data sequence, existence, range and limit, reasonableness, and validity, among others.

You also learned about batch and online input methods, input media and procedures, and input volume. Input methods include data capture and data entry. Data capture, which may be automated, involves identifying and recording source data. Data entry involves converting source data into a computer-readable form and entering it into the system. New technology offers optical and voice recognition systems, biological feedback devices, motion sensors, and a variety of graphical input devices.

Finally, you learned about security and control. Output control includes physical protection of data and reports, and control of unauthorized ports or devices that can extract data from the system. Input controls include audit trails, encryption, password security, data security, and the creation of access levels to limit persons authorized to view or use data.

Key Terms and Phrases

aesthetics *343*
audit trail *371*
authorization zone *361*
automated facsimile *358*
batch *369*
batch control *368*
batch input *368*
blog *358*
body zone *361*
calendar control *348*
character-based report *352*
check box *348*
combination check *368*
command button *348*
computer output to microfilm (COM) *358*
context-sensitive *345*
control break *354*
control break report *354*
control field *354*
control field order *354*
control zone *360*
data capture *363*
data entry *363*
data security *371*
data type check *367*
data validation rule *367*
detail line *352*
detail report *352*
dialog box *348*
diskless workstation *371*
drop-down list box *348*
encrypted *374*
encryption *374*
ergonomics *343*
exception report *352*
existence check *367*
faxback *358*
form filling *363*
form layout *360*
garbage in, garbage out (GIGO) *360*
graphical user interface (GUI) *338*
group footer *355*
group header *355*
hash totals *368*
heading zone *360*
human-computer interaction (HCI) *338*
input control *371*
input masks *346*
instruction zone *361*
interface technology *343*
limit check *367*
list box *348*
magnetic data strip *369*
menu bar *348*
mock-up *353*
natural language *344*
online data entry *369*
option button *348*
output control *370*
output security *370*
page footer *355*
page header *355*
podcast *358*
port protector *371*
process-control *336*
radio button *348*
range check *367*
reasonableness check *367*
records retention policy *372*
referential integrity *367*
report analysis form *353*
report footer *354*
report header *354*
RFID tag *369*
scannable text *362*
scroll bar *348*
separator *364*
sequence check *367*
source data automation *369*
source document *360*
storyboard *342*
summary report *353*
switchboard *348*
text box *348*
toggle button *348*
toolbar *348*
totals zone *361*
turnaround document *350*
usability metrics *342*
user-centered *337*
user interface (UI) *336*
user-selected *345*
validity check *367*
Webcast *357*

Learn It Online

Instructions: To complete the Learn It Online exercises, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the resources for this chapter, and click the link for the exercise you want to complete.

1 Chapter Reinforcement

TF, MC, and SA

Click one of the Chapter Reinforcement links for Multiple Choice, True/False, or Short Answer. Answer each question and submit to your instructor.

2 Flash Cards

Click the Flash Cards link and read the instructions. Type 20 (or a number specified by your instructor) in the Number of playing cards text box, type your name in the Enter your Name text box, and then click the Flip Card button. When the flash card is displayed, read the question and then click the ANSWER box arrow to select an answer. Flip through the Flash Cards. If your score is 15 (75%) correct or greater, click Print on the File menu to print your results. If your score is less than 15 (75%) correct, then redo this exercise by clicking the Replay button.

3 Practice Test

Click the Practice Test link. Answer each question, enter your first and last name at the bottom of the page, and then click the Grade Test button. When the graded practice test is displayed on your screen, click Print on the File menu to print a hard copy. Continue to take practice tests until you score 80% or better.

4 Who Wants To Be a Computer Genius?

Click the Computer Genius link. Read the instructions, enter your first and last name at the bottom of the page, and then click the Play button. When your score is displayed, click the PRINT RESULTS link to print a hard copy.

5 Wheel of Terms

Click the Wheel of Terms link. Read the instructions, and then enter your first and last name and your school name. Click the PLAY button. When your score is displayed on the screen, right-click the score and then click Print on the shortcut menu to print a hard copy.

6 Crossword Puzzle Challenge

Click the Crossword Puzzle Challenge link. Read the instructions, and then click the Continue button. Work the crossword puzzle. When you are finished, click the Submit button. When the crossword puzzle is redisplayed, submit it to your instructor.

SCR Associates Case Simulation Session 8: User Interface Design

Overview

The SCR Associates case study is a Web-based simulation that allows you to practice your skills in a real-world environment. The case study transports you to SCR's intranet, where you complete 12 work sessions, each aligning with a chapter. As you work on the case, you will receive e-mail and voice mail messages, obtain information from SCR's online libraries, and perform various tasks.

How do I use the case?

- Review the SCR background material in Chapter 1.
- Read the Preview for this session and study the Task List.
- Visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the **SCR Case Simulation**, and locate the intranet link.
- Enter your name and the password **sad9e.** An opening screen will display the 12 sessions.
- Select this session. Check your e-mail and voice mail carefully, and then work on the tasks.

Preview: Session 8

Now that the overall data design is complete, Jesse Baker wants you to work on output and user interface design. You will consider user needs, and apply principles of human-computer interaction to build a user-centered interface that is easy to learn and use. You also will consider data validation checks, source documents, forms, and reports.

Task List

1. *Create a detail report that will display all SCR courses in alphabetical order, with the course name and the instructor name in a group header; the Social Security number, name, and telephone number of each current student in the detail section; and the student count in a group footer.*
2. *Create a switchboard design with control buttons that lead to students, instructors, courses, course schedules, and course rosters. Allow a user to add, update, or delete records in each area. Jesse wants to see storyboards that show the proposed screens.*
3. *Suggest data validation checks for data entry screens.*
4. *Create a source document for an SCR mail-in registration form. Also need a design for a Web-based course registration form.*

FIGURE 8-40 Task list: Session 8.

Chapter Exercises

Review Questions

1. Explain the concept of human-computer interaction (HCI).
2. Explain the concept of a GUI and a switchboard. How does a GUI design differ from a character-based screen design?
3. Describe seven principles for a user-centered interface design.
4. Describe six types of user interface controls, and provide an example of how you could use each type in a data entry screen.
5. Define detail reports, exception reports, and summary reports. Explain the concept of a control field and how it is used to produce a control-break report.
6. List and describe various types of output, including technology-based forms of information delivery.
7. Explain each of the data validation rules mentioned in this chapter.
8. What are the main principles of source document design?
9. Explain batch and online input methods. Define source data automation and provide an example.
10. Provide four guidelines for reducing input volume.

Discussion Topics

1. Some systems analysts maintain that source documents are unnecessary. They say that all input can be entered directly into the system, without wasting time in an intermediate step. Do you agree? Can you think of any situations where source documents are essential?
2. Some systems analysts argue, "Give users what they ask for. If they want lots of reports and reams of data, then that is what you should provide. Otherwise, they will feel that you are trying to tell them how to do their jobs." Others say, "Systems analysts should let users know what information can be obtained from the system. If you listen to users, you'll never get anywhere, because they really don't know what they want and don't understand information systems." What do you think of these arguments?
3. Suppose your network support company employs 75 technicians who travel constantly and work at customer sites. Your task is to design an information system that provides technical data and information to the field team. What types of output and information delivery would you suggest for the system?
4. A user interface can be quite restrictive. For example, the interface design might not allow a user to exit to a Windows desktop or to log on to the Internet. Should a user interface include such restrictions? Why or why not?

Projects

1. Visit the administrative office at your school or a local company. Ask to see examples of output documents, such as computer-printed invoices, form letters, or class rosters. Analyze the design and appearance of each document, and try to identify at least one possible improvement for each.
2. Search the Web to find an example of an attractive user interface. Document your research and discuss it with your class.
3. Examine various application software packages to find examples of good (or bad) user interface design. Document your research and discuss it with your class.
4. Search your own files or review other sources to find good (or bad) examples of source document design. Document your research and discuss it with your class.

Apply Your Knowledge

The Apply Your Knowledge section contains four mini-cases. Each case describes a situation, explains your role in the case, and asks you to respond to questions. You can answer the questions by applying knowledge you learned in the chapter.

1 North Shore Boat Sales

Situation:

North Shore Boat Sales sells new and used boats and operates a Web-based boat brokerage business in Toronto. The company has grown, and North Shore needs a new information system to manage the inventory, the brokerage operation, and information about prospective buyers and sellers. Dan Robeson, the owner, asked you to design samples of computer screens and reports that the new system might produce.

1. Design a switchboard that includes the main information management functions that North Shore might require. Create a storyboard with a design layout that allows customers to perform the following functions: Obtain information about new boats, obtain information about used boats, send an e-mail to North Shore, learn more about the company, or review links to other marine-related sites.
2. Prospective buyers might want to search for boats by type, size, price range, or manufacturer. Develop a screen design that would permit those choices.
3. Suggest reports that might be useful to North Shore's management.
4. Suggest the general layout for a Web-based source document that prospective sellers could use to describe their boats. The information should include boat type (sail or power), manufacturer, year, length, type of engine, hull color, and asking price.

2 Terrier News

Situation:

Terrier News is a monthly newsletter devoted to various breeds of terriers and topics of interest to terrier owners and breeders. Annie West, the editor and publisher, asked you to help her design a system to enter and manage the hundreds of classified ads that Terrier News publishes. Some ads are for dogs wanted; some are for dogs for sale; and some offer products and services.

1. Design a suitable source document for ads that are telephoned or mailed in.
2. Explain user-centered design principles in a brief memo to Annie.
3. Suggest at least four user interface design guidelines that could be used for the new system.
4. Suggest several types of controls that might be used on the switchboard you plan to design. Explain why you chose each control, and create a storyboard that shows the switchboard layout.

3 Sky-High Internet Services

Situation:

Sky-High Internet Services is a leading Internet service provider in a metropolitan area. The new customer billing system has caused an increase in complaints. Tammy Jones, the office manager, asked you to investigate the situation. After interviewing data entry operators and observing the online data input process, you are fairly certain that most errors occur when data is entered.

1. Write a brief memo to Tammy explaining the importance of data validation during the input process.
2. Suggest at least three specific data validation checks that might help reduce input errors.
3. Would a batch input system offer any advantages? Write a brief memo to Tammy stating your views.
4. Suppose that Sky-High is predicting 25% annual growth, on a current base of 90,000 customers. If the growth pattern holds, how many customers will Sky-High have in three years? If it takes about 12 minutes to enter a new customer into the system, how many additional data entry operators will be needed to handle the growth next year? Assume that an operator works about 2,000 hours per year. Also assume a 30% annual attrition rate for existing customers.

4 Castle Point Antique Auction

Situation:

Castle Point Antique Auction operates a successful Web site that offers an auction forum for buyers and sellers of fine antiques. Monica Creighton, the owner, asked you to help her design some new documents and reports.

1. Suggest the general layout for a Web-based source document that prospective bidders would submit. The information should include user ID, password, name, address, telephone, e-mail address, item number, bid offered, and method of payment (money order, check, American Express, MasterCard, or Visa).
2. Suggest the general layout for a Web-based source document that prospective sellers could use to describe their antiques. The information should include the user ID, password, item, dimensions, origin, condition, and asking price.
3. Write a brief memo to Monica explaining the difference between detail reports, exception reports, and summary reports. Suggest at least one example of each type of report that she might want to consider.
4. Suggest several types of data validation checks that could be used when input data is entered.

Case Studies

Case studies allow you to practice specific skills learned in the chapter. Each chapter contains several case studies that continue throughout the textbook, and a chapter capstone case.

New Century Health Clinic

The associates at New Century Health Clinic approved your recommendations for a new computer system. Your next step is to develop a design for the new system, including output and user interface issues.

Background

To complete the output and user interface design for the new information system at New Century, you should review the DFDs and object-oriented diagrams you prepared previously, and the rest of the documentation from the systems analysis phase. Perform the following tasks.

Assignments

1. Dr. Jones has asked you to create a monthly Claim Status Summary report. He wants you to include the insurance company number, the patient number and name, the procedure date, the procedure code and description, the fee, the date the claim was filed, the amount of the claim, the amount of reimbursement, and the amount remaining unpaid. He wants you to group the data by insurance company number, with subtotals by company and grand totals for each numeric field. When you design the report, make sure to include a mock-up report and a report analysis form.
2. Design the daily appointment list and a monthly statement to make it readable and visually attractive. Include a mock-up report and a report analysis form for each report.
3. Determine the data required for a new patient. Design an input source document that will be used to capture the data and a data entry screen to input the information.
4. What data validation checks would the clinic need for the new patient data entry screen? Write a brief memo with your recommendations.

PERSONAL TRAINER, INC.

Personal Trainer, Inc., owns and operates fitness centers in a dozen Midwestern cities. The centers have done well, and the company is planning an international expansion by opening a new "supercenter" in the Toronto area. Personal Trainer's president, Cassia Umi, hired an IT consultant, Susan Park, to help develop an information system for the new facility. During the project, Susan will work closely with Gray Lewis, who will manage the new operation.

Background

Following the decision to use an in-house team to develop a design prototype, Susan began to work on the physical design for Personal Trainer's new information system. At this stage, she is ready to begin working with Gray on the output and user interface design. Together, Susan and Gray will seek to develop a user-centered design that is easy to learn and use. Personal Trainer users will include managers, fitness instructors, support staff, and members themselves.

Assignments

1. Create a detail report that will display all Personal Trainer courses in alphabetical order, with the course name and the instructor name in a group header; the Social Security number, name, and telephone number of each current student in the detail section; and the student count in a group footer.
2. Create a switchboard design with control buttons that lead to members, fitness instructors, activities and services, schedules, and fitness class rosters. Allow a user to add, update, or delete records in each area.
3. Suggest context-sensitive and specific Help for the switchboard and lower-level menus and forms. Prepare storyboards that show the proposed screens. Also suggest at least six types of data validation rules for data entry screens.
4. Design a mail-in source document that members can use to register for fitness classes. Also design a Web-based registration form.

VIDEO SUPERSTORE

Video Superstore has hired you to design two online data entry screens. Based on what you know about the operation of a video rental store, complete the following assignments.

Assignments

1. Design a weekly operations summary report that will include overall data on rentals, new customers, late charges, and anything else you think a store manager might want to review. Be sure to include numeric activity and dollar totals.
2. Design a data entry screen for entering new members.
3. Design a video rental input screen. In addition to the video data, the video rental form must include the following fields: Member Number, Name, and Date.
4. Suggest at least three data validation rules that might help reduce input errors for the video rental system.

CHAPTER CAPSTONE CASE: SoftWear, Limited

SoftWear, Limited (SWL), is a continuing case study that illustrates the knowledge and skills described in each chapter. In this case study, the student acts as a member of the SWL systems development team and performs various tasks.

Background

SoftWear, Limited, decided to use the payroll package developed by Pacific Software Solutions and customize it by adding its own ESIP system to handle SWL's Employee Savings and Investment Plan.

Because most of the payroll requirements would be handled by Pacific's payroll package, the IT team decided that Carla Moore would work on the new ESIP modules, and Rick Williams would concentrate on the rest of the payroll system. A new systems analyst, Becky Evans, was assigned to help Rick with the payroll package.

Pacific Software Solutions offered free training for new customers, so Rick and Becky attended a three-day train-the-trainer workshop at Pacific's site in Los Angeles. When they returned, they began developing a one-day training session for SWL users, including people from the accounting, payroll, and human resources departments. The initial training would include the features and processing functions of the new payroll package that was scheduled to arrive the following week.

Carla's first task was to work on the ESIP outputs. She had to design several reports: the ESIP Deduction Register, the ESIP Payment Summary, and the checks that SWL sends to the credit union and the stock purchase plan. Carla also needed to develop an ESIP Accounting Summary as a data file to be loaded into the accounting system.

Carla learned that standard SWL company checks were used to make the payments to the credit union and stock purchase department. In addition, the output entry to the accounting system had been specified by the accounting department in a standard format for all entries into that system.

To prepare the new ESIP Deduction Register, Carla reviewed the documentation from the systems analysis phase to make sure that she understood the logical design and the available data fields.

Carla decided to use a monthly format because accounting would apply some deductions on a monthly cycle. She started her design with a standard SWL report heading. Then she added a control heading and footing for each employee's Social Security number. The total of employee deductions would be compared with the transferred ESIP funds to make sure that they matched. After preparing a rough layout, Carla prepared the mock-up report shown in Figure 8-41, using test data for the month of May 2011.

During the systems analysis phase, Carla learned that the accounting department required a control report to show the ESIP deduction amounts that were not yet applied. The control report is used to verify the amount of the checks or fund transfers SWL makes to the credit union and stock purchase plan. The accounting department needs the control report to verify the accounting system outputs and balance the ESIP deduction totals against the payroll system's Payroll Register report.

Figure 8-42 on page 384 shows a mock-up of the ESIP Payment Summary report. Carla met with Buddy Goodson, director of accounting, to review the design. Buddy was pleased with the report and felt it would be acceptable to the company's outside auditors. Buddy met with the auditing firm later that week and secured the team's approval.

As Carla turned her attention to the user interface for the ESIP system, she realized that she would need to develop two new source documents: an ESIP Option Form and an ESIP Deduction Authorization Form. She also planned to design a main switchboard and all necessary screen forms.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

SoftWear, Ltd. *ESIP Deduction Register*

Month	Employee SSN	Period Ending	ESIP Option	Deduction
May 2011				
	111-11-1111			
		5/6/2011	Stock Purchase Plan	$15.00
		5/6/2011	Credit Union	$20.00
		5/13/2011	Credit Union	$25.00
		5/13/2011	Stock Purchase Plan	$15.00
		5/20/2011	Stock Purchase Plan	$15.00
		5/20/2011	Credit Union	$20.00
		5/27/2011	Stock Purchase Plan	$15.00
		5/27/2011	Credit Union	$20.00
			Employee Total	***$145.00***
	123-45-6789			
		5/6/2011	Stock Purchase Plan	$10.00
			Employee Total	***$10.00***
	222-22-2222			
		5/6/2011	Stock Purchase Plan	$10.00
		5/6/2011	Credit Union	$30.00
		5/13/2011	Stock Purchase Plan	$10.00
		5/13/2011	Credit Union	$30.00
		5/20/2011	Stock Purchase Plan	$10.00
		5/20/2011	Credit Union	$30.00
		5/27/2011	Credit Union	$30.00
		5/27/2011	Stock Purchase Plan	$10.00
			Employee Total	***$160.00***
	333-33-3333			
		5/6/2011	ESIP: Option 1	$9.99
		5/13/2011	ESIP: Option 1	$9.99
		5/18/2011	ESIP: Option 1	$9.99
		5/25/2011	ESIP: Option 1	$9.99
			Employee Total	***$39.96***
Summary for Period Ending 5/25/2011 (21 detail records)				
			Monthly Total	***$354.96***

FIGURE 8-41 Mock-up for the ESIP Deduction Register.

Carla started by designing an ESIP Option Form that could be used for adding new ESIP options and modifying existing ones when authorized by the vice president of human resources.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

SoftWear, Ltd. ***ESIP Payment Summary Report***

Last Deduction Period in Month: 5/27/2011

ESIP Code	Name	*Deduction*
CREDUN	Credit Union	*$205.00*
ESIP01	ESIP: Option 1	*$39.96*
SWLSTK	Stock Purchase Plan	*$110.00*
	Grand Total:	***$354.96***

FIGURE 8-42 Mock-up of the ESIP Payment Summary Report.

Next, Carla worked on a data entry screen based on the ESIP Option Form. Using SWL's existing screen design standards, she quickly developed the screen mock-up shown in Figure 8-43. Her design allowed users to add, delete, save, clear, or find a record by clicking the appropriate command button.

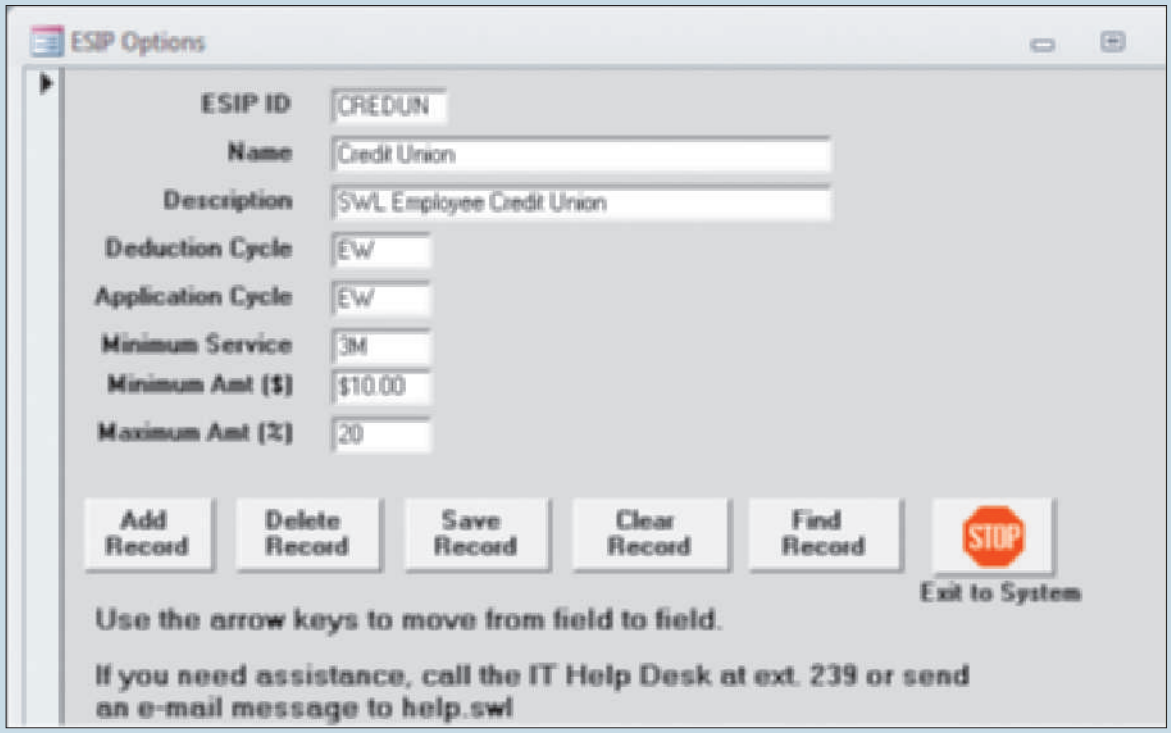


FIGURE 8-43 Carla's ESIP data entry screen form design.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

The ESIP Deduction Authorization Form required more data and several signatures. Carla divided the form into three sections: The employee completes the information in the top section; human resources completes the middle section; and payroll representatives complete the bottom section.

Carla designed a data entry screen based on the ESIP Deduction Authorization Form shown in Figure 8-44 and made the screen consistent with the other ESIP screen designs. Now a user could add, delete, save, clear, or find a record by clicking the appropriate command buttons. Carla also provided instructions to the operator for exiting from the system.

Carla decided to create a series of mock-ups to show users how the new ESIP deduction screen would work. In Figure 8-45 on the next page, the system has retrieved the employee's name, Joseph J. Smith, so the user can verify it against the source document.

The users approved the new design, with one suggestion — the system date should be added automatically as the entry date. Carla made the change and then designed the switchboard shown in Figure 8-46 on the next page, based on comments that users made during the design process. Now that she had a working model, Carla went back to the users to show them the complete package.

SWL

Employee Saving and Investment Plan
ESIP Deduction Authorization Form

Name	Social Security Number
_____	☐☐☐-☐☐-☐☐☐☐

Option (Check One):	Deduction Amount:
SWL Credit Union ☐ Employee Stock Purchase ☐ Other _____ ☐	$_____

I want to select the ESIP option and deduction amount specified above, and I authorize the company to make this deduction from my regular earnings. I have read the plan description and I understand that SWL's ESIP plan is subject to company policies and various federal and state regulations that might change in the future.

Signed _____ Date _____

Human Resources Department Approval:

By _____ Date _____

Payroll Department Verification:

By _____ Date _____

Effective Date _____

FIGURE 8-44 The ESIP Deduction Authorization Form.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

SWL Team Tasks

1. Review the mock-up report shown in Figure 8-41 on page 383. When Carla showed this report design to Mike Feiner, director of human resources, he said that he wanted to see the data grouped by the type of ESIP deduction with the appropriate subtotals. Carla wants you to modify the report design to satisfy his request. You can use Microsoft Access, a report generator, or simply construct a sample layout using any word processing or drawing program. Be sure to show the placement and grouping of all fields.
2. Carla Moore also wants employees to have an online information request form that they can use to learn more about ESIP options and request up-to-date balances for their ESIP accounts. Follow the guidelines and suggestions in this chapter, and design an online screen form for Carla.
3. In addition to being available online, Carla wants the information request form to be available as a paper source document, which can be used by employees who do not have easy access to the online form. Follow the guidelines and suggestions in this chapter, and design a paper source document for Carla.
4. Carla wants an update on usability, how users read on the Web. Review the material in this chapter and visit the Web to learn more about this topic. Summarize the results of your research in a memo to Carla.

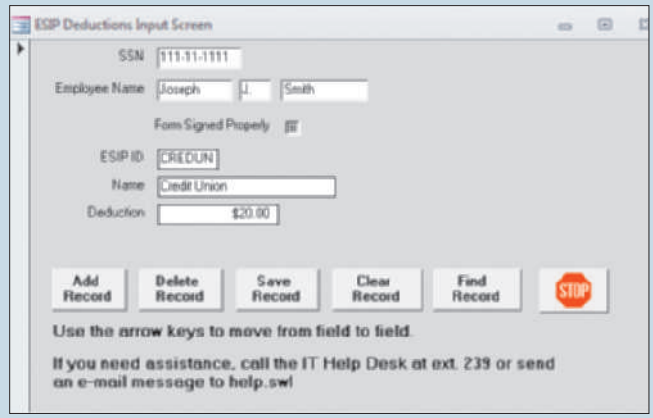


FIGURE 8-45 After the user presses the ENTER key, the system retrieves the employee name and displays it so the user can verify it. Notice that the user must check a box to verify that the form has been signed properly.

FIGURE 8-46 The ESIP switchboard includes option buttons and command buttons to select various processing choices.

Manage the SWL Project

You have been asked to manage SWL's new information system project. One of your most important activities will be to identify project tasks and determine when they will be performed. Before you begin, you should review the SWL case in this chapter. Then list and analyze the tasks, as follows:

LIST THE TASKS Start by listing and numbering at least 10 tasks that the SWL team needs to perform to fulfill the objectives of this chapter. Your list can include SWL Team Tasks and any other tasks that are described in this chapter. For example, Task 3 might be to Find out what output is needed, and Task 6 might be to Design an output screen.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

ANALYZE THE TASKS Now study the tasks to determine the order in which they should be performed. First identify all concurrent tasks, which are not dependent on other tasks. In the example shown in Figure 8-47, Tasks 1, 2, 3, 4, and 5 are concurrent tasks, and could begin at the same time if resources were available.

Other tasks are called dependent tasks, because they cannot be performed until one or more earlier tasks have been completed. For each dependent task, you must identify specific tasks that need to be completed before this task can begin. For example, you would want to find out what output is needed before you could design an output screen, so Task 6 cannot begin until Task 3 is completed, as Figure 8-47 shows.

Chapter 3 describes project management tools, techniques, and software. To learn more, you can use the Features section on your Student Study Tool CD-ROM, or visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and locate the project management resources library for this book. On the Web, Microsoft offers demo versions, training, and tips for using Project 2010. You also can visit the OpenWorkbench.org site to learn more about this free, open-source software.

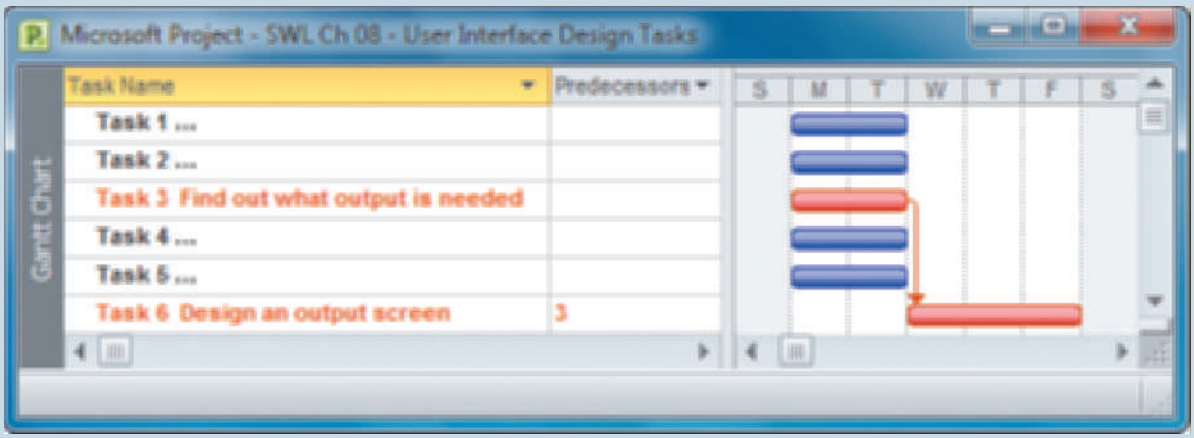


FIGURE 8-47 Tasks 1, 2, 3, 4, and 5 are concurrent tasks that could be performed at the same time. Task 6 is a dependent task that cannot be performed until Task 3 has been completed.

Ready for a Challenge?

In additional to technical skills, IT professionals need critical thinking skills such as perception, organization, analysis, problem-solving, and decision-making. The Ready for a Challenge feature can help you learn, practice, and apply critical thinking skills that you can take to the workplace.

This week, the IT team is working on user interface design for the new C³ system. One goal is to reduce input errors by using validation rules and input masks. The team leader has assigned you to do research and develop a recommendation. To perform these tasks, you must navigate to the Microsoft Access Help area, where you will see examples of validation rules and input masks. You should also review Chapter 8 of your systems analysis textbook.

Based on requirements modeling, you know that the new C³ system will store personal data about customers, their buying habits, and their interests. To reduce errors, the C³ user interface will use input masks where possible.

Here are five sample data items, with specific descriptions and examples:

Data Item	Description	Examples
First Name	Must start with a capital letter followed by at least one, and up to 9 more lowercase letters.	Li Stephanie
Middle Initial	May have up to one capital letter, or none.	J
Last Name	Must start with a capital letter followed by at least one, and up to 11 more lowercase letters.	Steinbrenner Ho
Category	Must start with two capital letters followed by two digits.	AB12 XY01
Postal Code	Must have five digits, and may be followed by a hyphen and four more digits.	12345 12345-9999

In addition to input masks, the team leader wants you to learn about validation rules and text. Specifically, she wants you to go back to the Microsoft Access Help area and review the explanation of validation rules, and the examples.

Practice Tasks

A. Create an input mask for each data item shown in the table.
B. Develop a brief handout that explains the concept of validation rules. Include an example of a validation rule that will only accept the letters A or B.

After you complete the Practice Tasks, to check your work and view sample answers, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the resources for this chapter, and locate Ready for a Challenge?.

The Challenge

Now that you are familiar with input masks and validation rules, your team leader wants you to review two more data items, as follows:

Customer ID	Must include the first capital letter of the customer's last name, followed by the last four digits of the customer's Social Security Number.	R0118 B1234
Password	Must include at least six, and no more than 12 letters or digits.	1X2y3Z 123456789999

Challenge Tasks

A. Create input masks for the two additional data items shown in the table.
B. Create a validation rule that will only accept numbers greater than 10 **and** less than 20.

This page intentionally left blank

CHAPTER 9 Data Design

Chapter 9 is the second of three chapters in the systems design phase of the SDLC. In this chapter, you will focus on data design skills that are necessary to construct the physical model of the information system.

INTRODUCTION

OBJECTIVES

When you finish this chapter, you will be able to:

- Explain file-oriented systems and how they differ from database management systems
- Explain data design terminology, including entities, fields, common fields, records, files, tables, and key fields
- Describe data relationships, draw an entity-relationship diagram, define cardinality, and use cardinality notation
- Explain the concept of normalization
- Explain the importance of codes and describe various coding schemes
- Explain data warehousing and data mining
- Differentiate between logical and physical storage and records
- Explain data control measures

During the systems analysis phase, you created a logical model of the system. Now, you must decide how data will be organized, stored, and managed. These are important issues that affect data quality and consistency.

This chapter begins with a review of data design concepts and terminology, then discusses file-based systems and database systems, including Web-based databases. You will learn how to create entity-relationship diagrams that show the relationships among data elements, and you will learn how to use normalization concepts. You also will learn about using codes to represent data items. The chapter concludes with a discussion of data storage and access issues, including data warehousing and data mining, physical design, logical and physical records, data storage formats, and data control.

CHAPTER INTRODUCTION CASE: Mountain View College Bookstore

Background: Wendy Lee, manager of college services at Mountain View College, wants a new information system that will improve efficiency and customer service at the three college bookstores.

In this part of the case, Tina Allen (systems analyst) and David Conroe (student intern) are talking about data design issues.

Participants: Tina and David
Location: Mountain View College Cafeteria, Monday morning, December 5, 2011
Project status: Tina and David have completed their user interface design tasks and are ready to work on data design for the new system.
Discussion topics: Data design terms and concepts, cardinality, relational databases, normalization, Web-based design, codes, and physical design issues

Tina: Good morning, David. Now that we have a logical model of the bookstore information system, we're ready for the next step. We have to select an overall data design strategy. I think we should start by looking at a relational database, rather than a file processing design.

David: *What are the pros and cons?*

Tina: Well, in some situations file processing systems are better, especially when you have to process large numbers of records in a sequence. But in our case, I think a relational database would be more powerful and flexible.

David: *I know what a database is, but what do you mean by the term relational?*

Tina: In a relational database, all the entities — the individual people, places, events, and transactions — are stored in separate locations called tables, which are related or linked together. That means you have to enter an item of data only once, and you can access all the data items just as if they were all stored in a single location.

David: *That makes sense. How do we decide what data goes where?*

Tina: We'll start by creating an entity-relationship diagram. Then we'll develop a set of table designs that follow a set of rules called normalization.

David: *I've heard that term before. Aren't there several different levels of normalization, called normal forms?*

Tina: Yes, and we want our data to be in what's called third normal form, which is what most business-related systems use. We also will consider using various codes to represent data items.

David: *I know that we decided to build the system to run on the college network and then migrate to a Web-based system in the future. But shouldn't we use a design that will make it easy to migrate to the Web?*

Tina: Yes, and a relational database will be the most flexible approach.

David: *Sounds good. Any other issues?*

Tina: Well, we need to consider some physical design issues, too. Here's a task list to get us started:

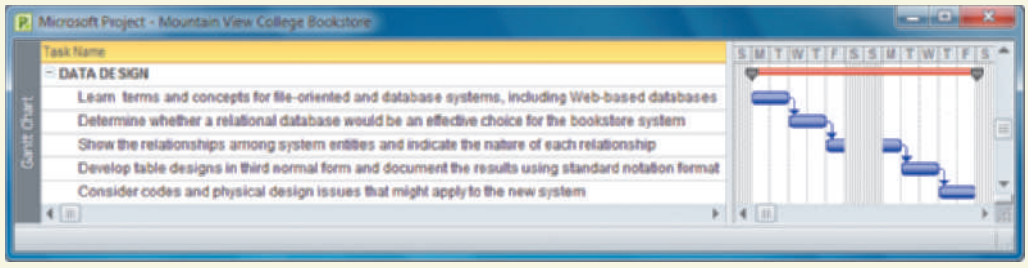


FIGURE 9-1 Typical data design task list.

Data Design Concepts

Before constructing an information system, a systems analyst must understand basic data design concepts, including data structures and the characteristics of file-oriented and database management systems, including Web-based database design.

Data Structures

A **data structure** is a framework for organizing, storing, and managing data. Data structures consist of files or tables that interact in various ways. Each **file** or **table** contains data about people, places, things, or events. For example, one file or table might contain data about customers, and other files or tables might store data about products, orders, suppliers, or employees. Depending on how the files and tables are designed, an information system is called a file-oriented system or a database management system (DBMS).

Figure 9-2 shows a typical auto repair shop that handles tune-ups, brakes, and alignment. The following sections describe how this shop might manage its business data.

FIGURE 9-2 In the example shown here, data about the mechanic, the customer, and the brake job might be stored in a file-oriented system or in a database system.

FILE-ORIENTED SYSTEM A **file-oriented system**, sometimes called a file processing system, stores data in one or more separate files. For example, Figure 9-3 shows the auto repair shop with two separate file-oriented systems: A Job Records system that uses a JOB data file, and an Employee Records system that uses a MECHANIC data file. In the example, the JOB file contains the data necessary to answer inquiries and generate reports about work performed at the shop. Similarly, the MECHANIC file stores the data necessary to answer inquiries and generate reports about the shop's employees.

Notice that the same data is stored in more than one location. For example, three items of information (Mechanic No, Name, and Pay Rate) are stored in both data files. This redundancy is a major disadvantage of file-oriented systems, because it reduces efficiency and data quality.

DATABASE MANAGEMENT SYSTEM In a **database management system** (DBMS), all the tables are connected by common fields. A typical common field might be a Customer Number, which could be used to locate information about that customer in other tables. A common field that connects two tables is said to *link*, *join*, or *relate* the tables.

In a DBMS, the linked tables form a unified data structure that greatly improves data quality and access. This design, also called a **relational database** or **relational model**, was introduced in the 1970s and continues to be the most popular approach for organizing, storing, and managing business data.

Now consider Figure 9-4, which shows how the same auto repair shop might use a relational database instead of a file-oriented system. Notice that the two tables are linked by the Mechanic No field. This link allows information to be accessed from either table as if the two tables were one large table, making it unnecessary to store duplicate information.

Overview of File Processing

Although file processing is an older approach, you should understand how these systems were designed, constructed, and maintained. Some companies still use file processing to handle large volumes of structured data on a regular basis. Many older legacy systems

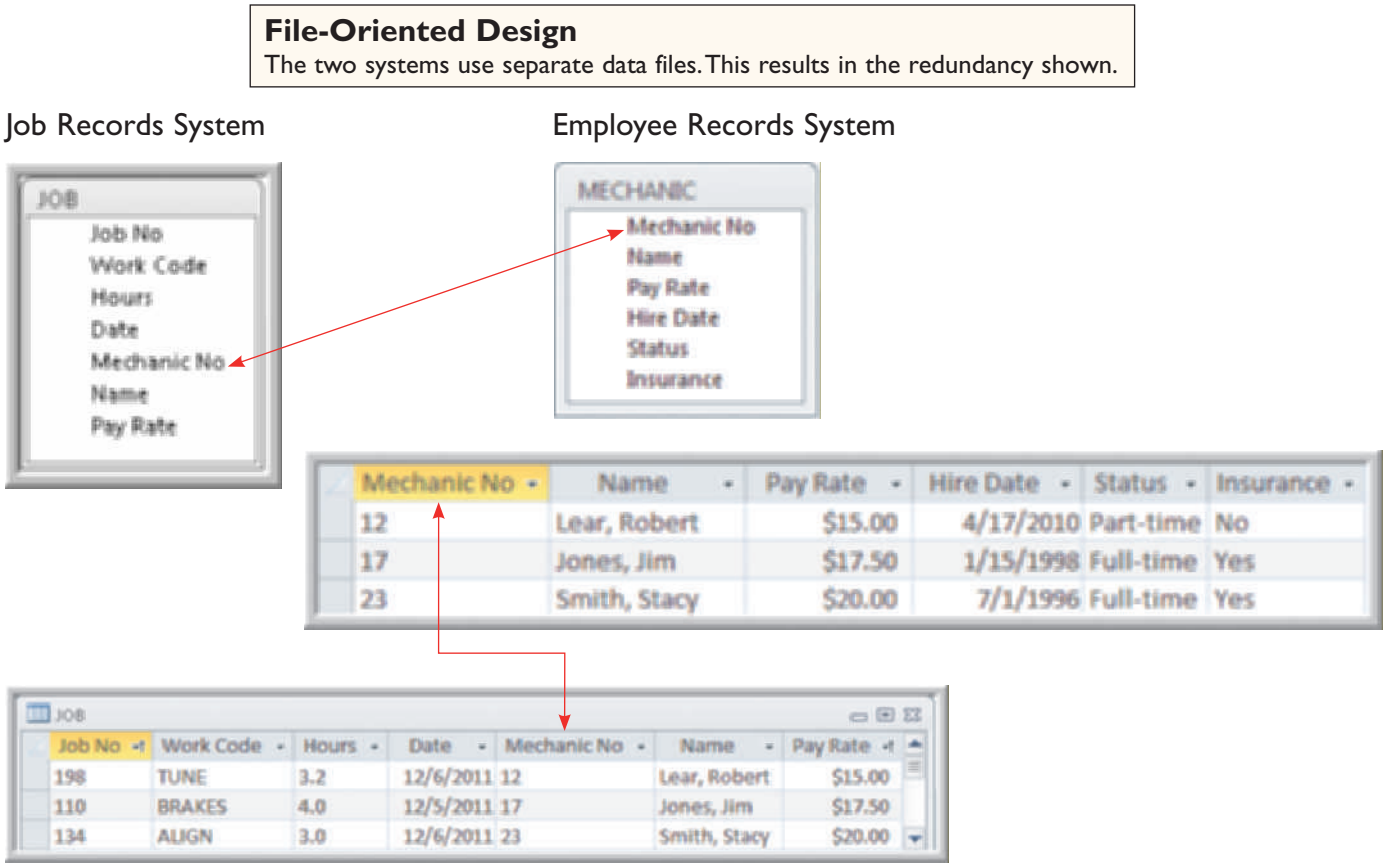


FIGURE 9-3 This auto repair shop uses two separate systems: a Job Records system and an Employee Records system. This design requires the same data to be entered in two different places.

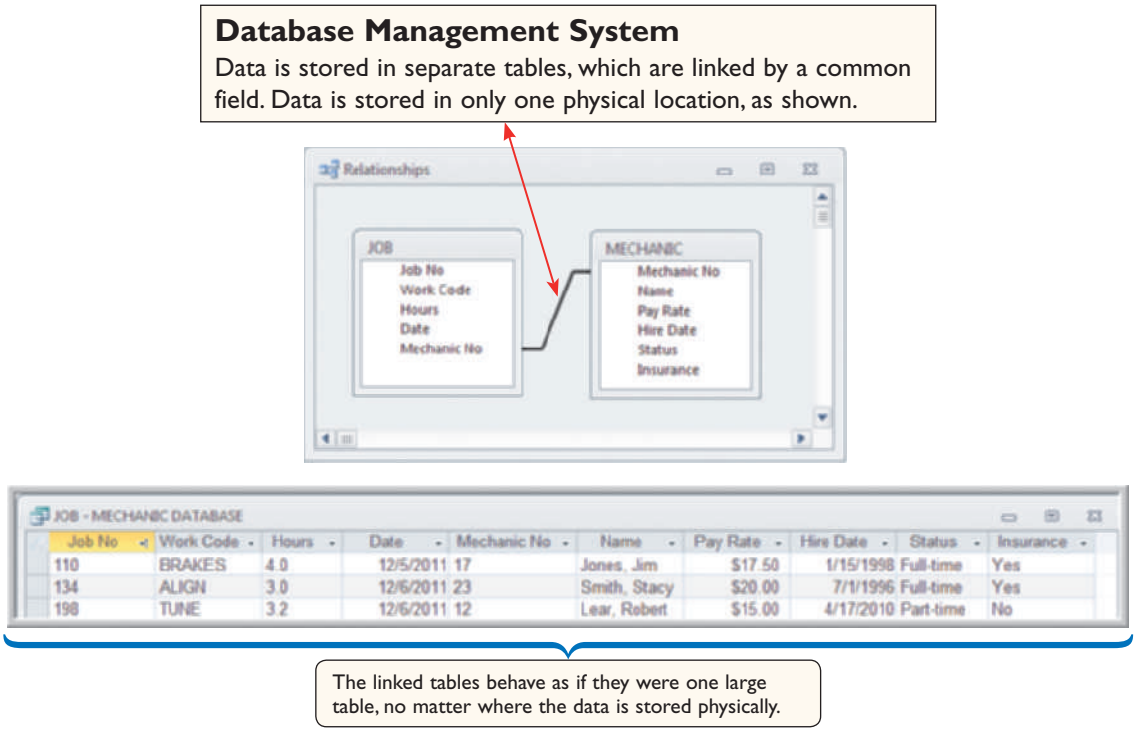


FIGURE 9-4 This is how the same repair shop could use a database design that avoids duplication. A common field joins the tables, and the data behaves like one large table, regardless of where it physically is stored.

utilized file processing designs because that method worked well with mainframe hardware and batch input. Although much less common today, file processing can be efficient and cost-effective in certain situations. For example, consider a credit card company that posts thousands of daily transactions from a TRANSACTION file to customer balances stored in a CUSTOMER file, as shown in Figure 9-5. For that relatively simple process, file processing is highly effective.

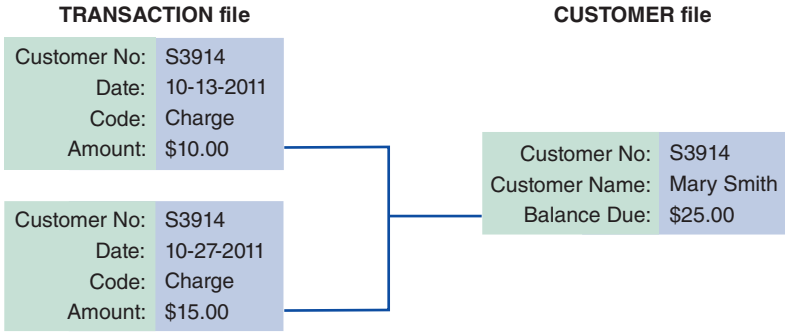


FIGURE 9-5 A credit card company might use a file processing system to post daily sales transactions from a TRANSACTION file to the CUSTOMER file.

In a typical file processing environment, a company might have three departments, each with its own information system and data files. Three potential problems exist in a file processing environment. The first problem is **data redundancy**, which means that data common to two or more information systems is stored in several places. Data redundancy requires more storage space, and maintaining and updating data in several locations is expensive.

Second, **data integrity** problems can occur if updates are not applied in every file. Changing the data in only one of the systems will cause inconsistent data and result in incorrect information in the second system.

The third problem is the rigid data structure of a typical file processing environment. Businesses must make decisions based on company-wide data, and managers often require information from multiple business units and departments. In a file processing environment, that means retrieving information from independent, file-based systems, which is slow and inefficient.

A file-oriented information system can contain various types of files, including master files, table files, transaction files, work files, security files, and history files.

- A **master file** stores relatively permanent data about an entity. For example, a PRODUCT master file contains one logical record for each product the company sells. The quantity field in each record might change daily, but other data, such as the product's code, name, and description, would not change.
- A **table file** contains reference data used by the information system. As with master files, table files are relatively static and are not updated by the information system. Examples of table files include tax tables and postage rate tables.
- A **transaction file** stores records that contain day-to-day business and operational data. A transaction file is an input file that updates a master file; after the update is completed, the transaction file has served its purpose. An example of a transaction file is a charges and payments file that updates a customer balance file.
- A **work file** is a temporary file created by an information system for a single task. Examples of work files include sorted files and report files that hold output reports until they are printed.
- A **security file** is created and saved for backup and recovery purposes. Examples of security files include audit trail files and backups of master, table, and transaction files. New security files must be created regularly to replace outdated files.
- A **history file** is a file created for archiving purposes. For example, students who have not registered for any course in the last two semesters might be deleted from the active student master file and added to an inactive student file, which is a history file that can be used for queries or reports.

These problems do not exist in a database system, which is explained in the following section.

The Evolution from File Systems to Database Systems

A properly designed database system offers a solution to the problems of file processing. A database provides an overall framework that avoids data redundancy and supports a real-time, dynamic environment, two potential problems of a file processing system.

In a file processing environment, data files are designed to fit individual business systems. In contrast, in a database environment, several systems can be built around a single database. Figure 9-6 shows a database environment with a database serving four separate information systems.

A **database management system** (**DBMS**) is a collection of tools, features, and interfaces that enables users to add, update, manage, access, and analyze the contents of a set of data. From a user's point of view, the main advantage of a DBMS is that it offers timely, interactive, and flexible data access. Specific DBMS advantages include the following:

FIGURE 9-6 In this example, a sales database can support four separate business systems.

- **Scalability**, which means that a system can be expanded, modified, or downsized easily to meet the rapidly changing needs of a business enterprise. For example, if a company decides to add data about secondary suppliers of material it uses, a new table can be added to the relational database and linked with a common field.
- Better support for client/server systems. In a **client/server** system, processing is distributed throughout the organization. Client/server systems require the power and flexibility of a database design. You will learn more about client/server systems in Chapter 10.
- Economy of scale. Database design allows better utilization of hardware. If a company maintains an enterprise-wide database, processing is less expensive using a powerful mainframe server instead of using several smaller computers. The inherent efficiency of high-volume processing on larger computers is called **economy of scale**.
- Flexible data sharing. Data can be shared across the enterprise, allowing more users to access more data. A database can be highly flexible, allowing users to view the same information in different ways. Users are empowered because they have access to the information they need to do their jobs.
- Enterprise-wide application. Typically, a DBMS is managed by a person called a **database administrator** (**DBA**), who assesses overall requirements and maintains the database for the benefit of the entire organization rather than a single department or user. Database systems can support enterprise-wide applications more effectively than file processing systems.
- Stronger standards. Effective database administration helps ensure that standards for data names, formats, and documentation are followed uniformly throughout the organization.
- Controlled redundancy. Redundancy means storing data in more than one place, which can result in inconsistency and data errors. Because the data is stored in a set of related tables, data items do not need to be duplicated in multiple locations. Even where some duplication is desirable for performance reasons, or disaster recovery, the database approach allows control of the redundancy.

- Better security. The DBA can define authorization procedures to ensure that only legitimate users can access the database and can allow different users to have different levels of access. Most DBMSs provide sophisticated security support.
- Increased programmer productivity. Programmers do not have to create the underlying file structure for a database. That allows them to concentrate on logical design and, therefore, a new database application can be developed more quickly than in a file-oriented system.
- Data independence. Systems that interact with a DBMS are relatively independent of how the physical data is maintained. That design provides the DBA flexibility to alter data structures without modifying information systems that use the data.

Although the trend is toward enterprise-wide database design, many companies still use a combination of centralized DBMSs and smaller, department-level database systems. Why is this so? Most large businesses view data as a company-wide resource that must be accessible to users throughout the company. At the same time, other factors encourage a decentralized design, including network expense; a reluctance to move away from smaller, more flexible systems; and a realization that enterprise-wide DBMSs can be highly complex and expensive to maintain. The compromise, in many cases, is a client/server design, where processing is shared among several computers. Client/server systems are described in detail in Chapter 10. As with many design decisions, the best solution depends on the individual circumstances.

DBMS Components

A DBMS provides an interface between a database and users who need to access the data. Although users are concerned primarily with an easy-to-use interface and support for their business requirements, a systems analyst must understand all of the components of a DBMS. In addition to interfaces for users, database administrators, and related systems, a DBMS also has a data manipulation language, a schema and subschemas, and a physical data repository, as shown in Figure 9-7.

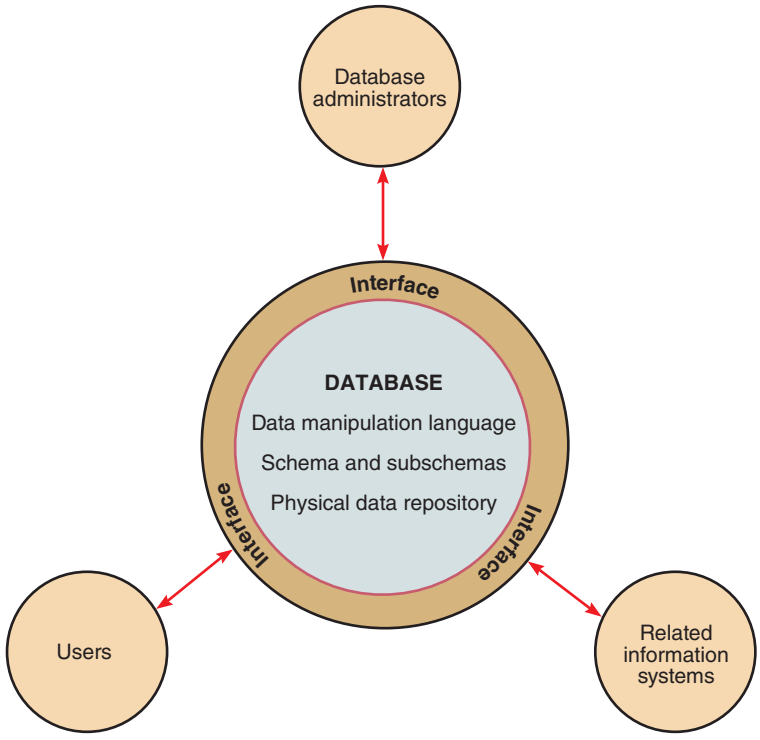


FIGURE 9-7 In addition to interfaces for users, database administrators, and related information systems, a DBMS also has a data manipulation language, a schema and subschemas, and a physical data repository.

Interfaces for Users, Database Administrators, and Related Systems

When users, database administrators, and related information systems request data and services, the DBMS processes the request, manipulates the data, and provides a response.

USERS Users typically work with predefined queries and switchboard commands, but also use query languages to access stored data. A **query language** allows a user to specify a task without specifying how the task will be accomplished. Some query languages use natural language commands that resemble ordinary English sentences. With a **query by example** (QBE) language, the user

provides an example of the data requested. Many database programs also generate SQL (**Structured Query Language**), which is a language that allows client workstations to communicate with servers and mainframe computers. Figure 9-8 shows a QBE request for all Red Candy Metallic or Blue Flame Metallic 2011 Ford Fusions with navigation. The QBE request generates the SQL commands shown at the bottom of Figure 9-8.

DATABASE ADMINISTRATORS A DBA is responsible for DBMS management and support. DBAs are concerned with data security and integrity, preventing unauthorized access, providing backup and recovery, audit trails, maintaining the database, and supporting user needs. Most DBMSs provide utility programs to assist the DBA in creating and updating data structures, collecting and reporting patterns of database usage, and detecting and reporting database irregularities.

RELATED INFORMATION SYSTEMS A DBMS can support several related information systems that provide input to, and require specific data from, the DBMS. Unlike a user interface, no human intervention is required for two-way communication between the DBMS and the related systems.

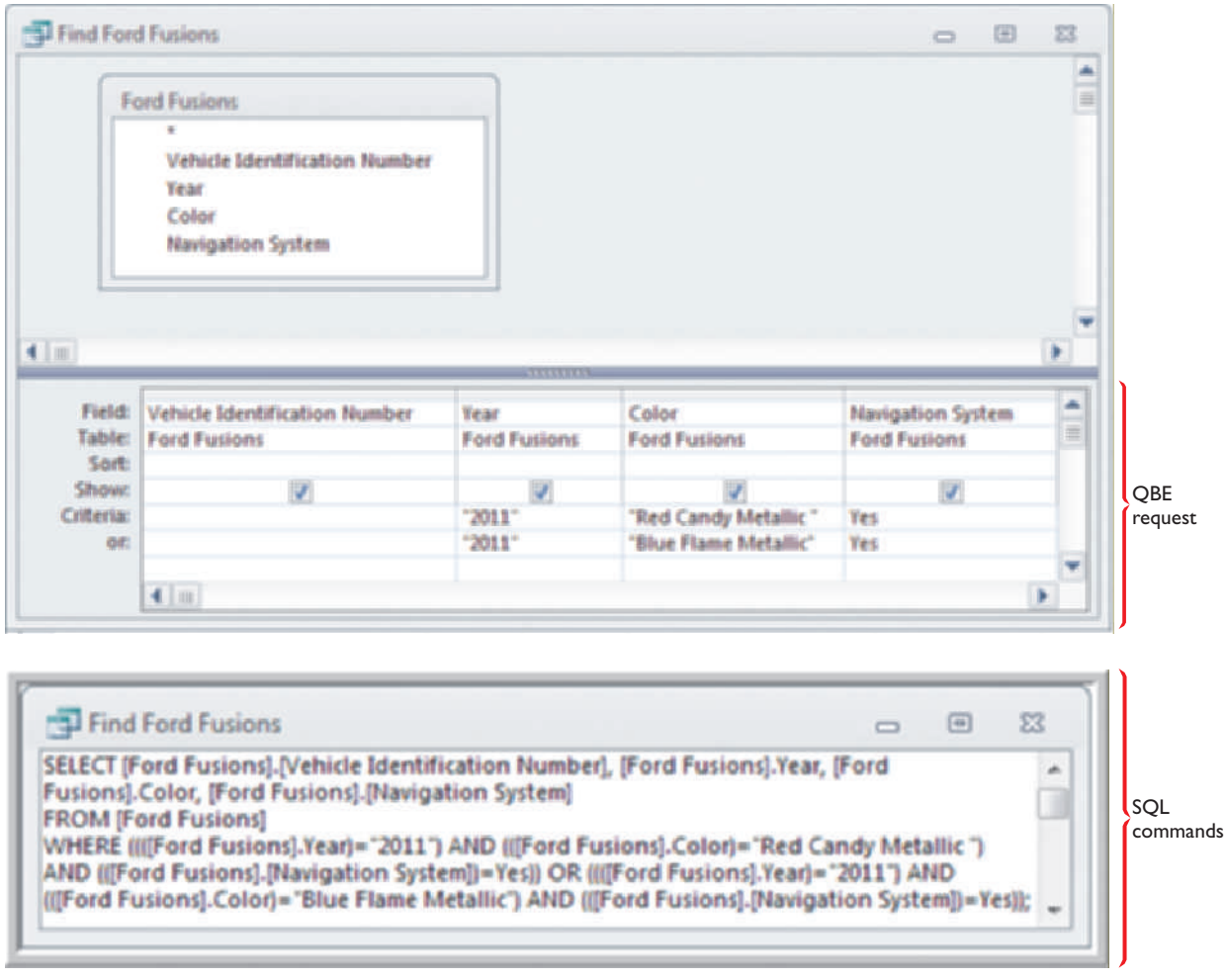


FIGURE 9-8 Using QBE, a user can request a list of all Red Candy Metallic or Blue Flame Metallic 2011 Ford Fusions with navigation.

Data Manipulation Language

A **data manipulation language** (**DML**) controls database operations, including storing, retrieving, updating, and deleting data. Most commercial DBMSs, such as Oracle and IBM's DB/2, use a DML. Some database products, such as Microsoft Access, also provide an easy-to-use graphical environment that enables users to control operations with menu-driven commands.

Schema

The complete definition of a database, including descriptions of all fields, tables, and relationships, is called a **schema**. You also can define one or more subschemas. A **subschema** is a view of the database used by one or more systems or users. A subschema defines only those portions of the database that a particular system or user needs or is allowed to access. For example, to protect individual privacy, you might not want to allow a project management system to retrieve employee pay rates. In that case, the project management system subschema would not include the pay rate field. Database designers also use subschemas to restrict the level of access permitted. For example, specific users, systems, or locations might be permitted to create, retrieve, update, or delete data, depending on their needs and the company's security policies.

Physical Data Repository

In Chapter 5, you learned about a data dictionary, which describes all data elements included in the logical design. At this stage of the systems development process, the data dictionary is transformed into a physical data repository, which also contains the schema and subschemas. The physical repository might be centralized, or it might be distributed at several locations. In addition, the stored data might be managed by a single DBMS, or several systems. To resolve potential database connectivity and access problems, companies use ODBC-compliant software that enables communication among various systems and DBMSs. **ODBC**, which stands for **open database connectivity**, is an industry-standard protocol that makes it possible for software from different vendors to interact and exchange data. ODBC uses SQL statements that the DBMS understands and can execute, similar to the ones shown in Figure 9-7 on page 396. Another common standard is called **JDBC**, or **Java database connectivity**. JDBC enables Java applications to exchange data with any database that uses SQL statements and is JDBC-compliant.

You will learn more about physical design issues in Chapter 10, which discusses system architecture, and in Chapter 11, which discusses system implementation and data conversion.

Web-Based Database Design

The concept of Web-based systems was discussed in Chapter 7, Development Strategies. In this chapter, you will revisit this concept and examine the main components of a Web-based database system. The following sections discuss the characteristics of Web-based design, Internet terminology, connecting a database to the Web, and data security on the Web.

Characteristics of Web-Based Design

Figure 9-9 on the next page lists some major characteristics of Web-based database design. In a Web-based design, the Internet serves as the front end, or interface, for the database management system. Internet technology provides enormous power and flexibility because

the system is not tied to any specific combination of hardware and software. Access to the database requires only a Web browser and an Internet connection. Web-based systems are popular because they offer ease of access, cost-effectiveness, and worldwide connectivity — all of which are vital to companies that must compete in a global economy.

Web-Based Database Design Characteristics

CHARACTERISTIC	EXPLANATION
Global access	The Internet enables worldwide access, using existing infrastructure and standard telecommunications protocols.
Ease of use	Web browsers provide a familiar interface that is user-friendly and easily learned.
Multiple platforms	Web-based design is not dependent on a specific combination of hardware or software. All that is required is a browser and an Internet connection.
Cost effectiveness	Initial investment is relatively low because the Internet serves as the communication network. Users require only a browser, and Web-based systems do not require powerful workstations. Flexibility is high because numerous outsourcing options exist for development, hosting, maintenance, and system support.
Security issues	Security is a universal issue, but Internet connectivity raises special concerns. These can be addressed with a combination of good design, software that can protect the system and detect intrusion, stringent rules for passwords and user identification, and vigilant users and managers.
Adaptability issues	The Internet offers many advantages in terms of access, connectivity, and flexibility. Migrating a traditional database design to the Web, however, can require design modification, additional software, and some added expense.

FIGURE 9-9 Web-based design characteristics include global access, ease of use, multiple platforms, cost effectiveness, security issues, and adaptability issues. In a Web-based design, the Internet serves as the front end, or interface, for the database management system. Access to the database requires only a Web browser and an Internet connection.

Internet Terminology

To understand Web-based data design, it is helpful to review some basic Internet terms and concepts. To access information on the Internet, a person uses a **Web browser**, which is an application that enables the user to navigate, or browse, the Internet and display Web pages on his or her local computer. A **Web page** is a text document written in **HTML** (**Hypertext Markup Language**). HTML uses formatting codes called **tags**, which specify how the text and visual elements will be displayed in a Web browser. Web pages are stored on a **Web server**, which is a computer that receives requests and makes Web pages available to users. Together, the Web server and the Web pages are referred to as a **Web site**.

In addition to maintaining a Web site, many companies use intranets and extranets to support business operations and communications. An **intranet** is a private, company-owned

ON THE WEB

To learn more about HTML, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the HTML link.

network to provide Web-based access to internal users. An **extranet** is an extension of a company intranet that allows access by external users, such as customers and suppliers. Extranets are typical examples of B2B (business-to-business) data sharing and EDI (electronic data interchange), which were discussed in Chapter 1, where you also learned about Extensible Markup Language (XML). XML is a flexible data description language that allows Web-based communication between different hardware and software environments. Because intranets and extranets use the same **protocols**, or data transmission standards, as the Internet, they are called **Web-centric**.

The Internet and company intranets/extranets are forms of client/server architecture. In a client/server design, tasks are divided between **clients**, which are workstations that users interact with, and **servers**, which are computers that supply data, processing, and services to the client workstations. Client/server architecture is discussed in more detail in Chapter 10, System Architecture.

Connecting a Database to the Web

To access data in a Web-based system, the database must be connected to the Internet or intranet. The database and the Internet speak two different languages, however. Databases are created and managed by using various languages and commands that have nothing to do with HTML, which is the language of the Web. The objective is to connect the database to the Web and enable data to be viewed and updated.

To bridge the gap, it is necessary to use **middleware**, which is software that integrates different applications and allows them to exchange data. Middleware can interpret client requests in HTML form and translate the requests into commands that the database can execute. When the database responds to the commands, middleware translates the results into HTML pages that can be displayed by the user's browser, as shown in Figure 9-10. Notice that the four steps in the process can take place using the Internet or a company intranet as the communications channel.

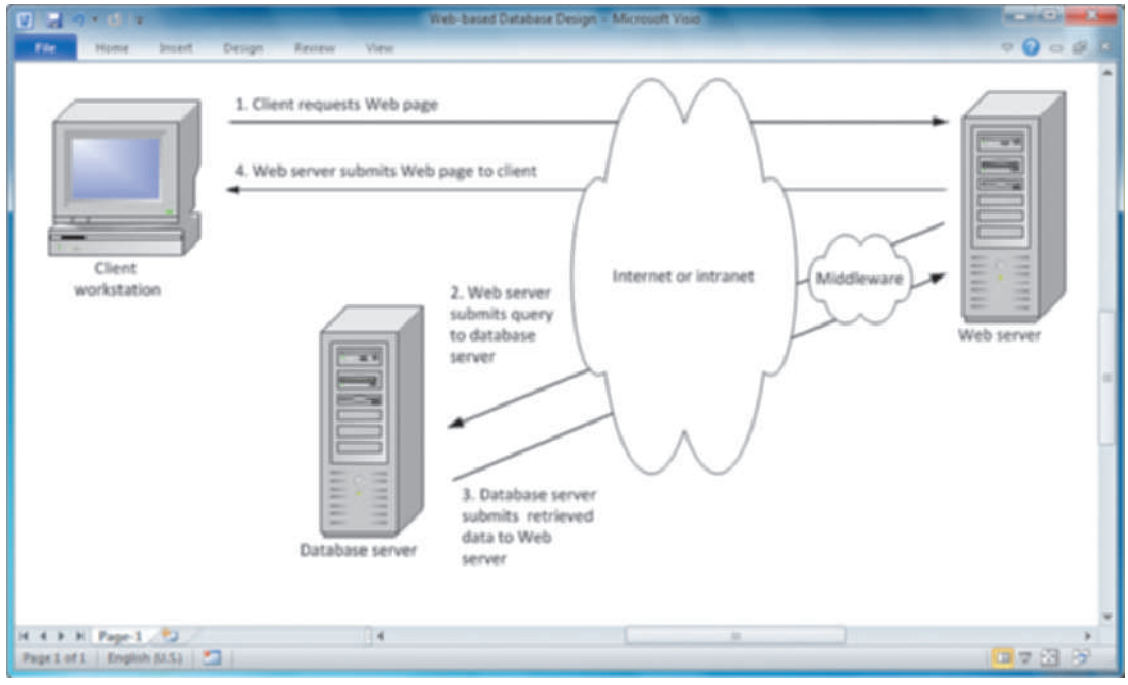


FIGURE 9-10 When a client workstation requests a Web page (1), the Web server uses middleware to generate a data query to the database server (2). The database server responds (3) and middleware translates the retrieved data into an HTML page that can be sent by the Web server and displayed by the user's browser (4).

A popular example of middleware is Adobe ColdFusion, which is shown in Figure 9-11. Middleware is discussed in more detail in Chapter 10.

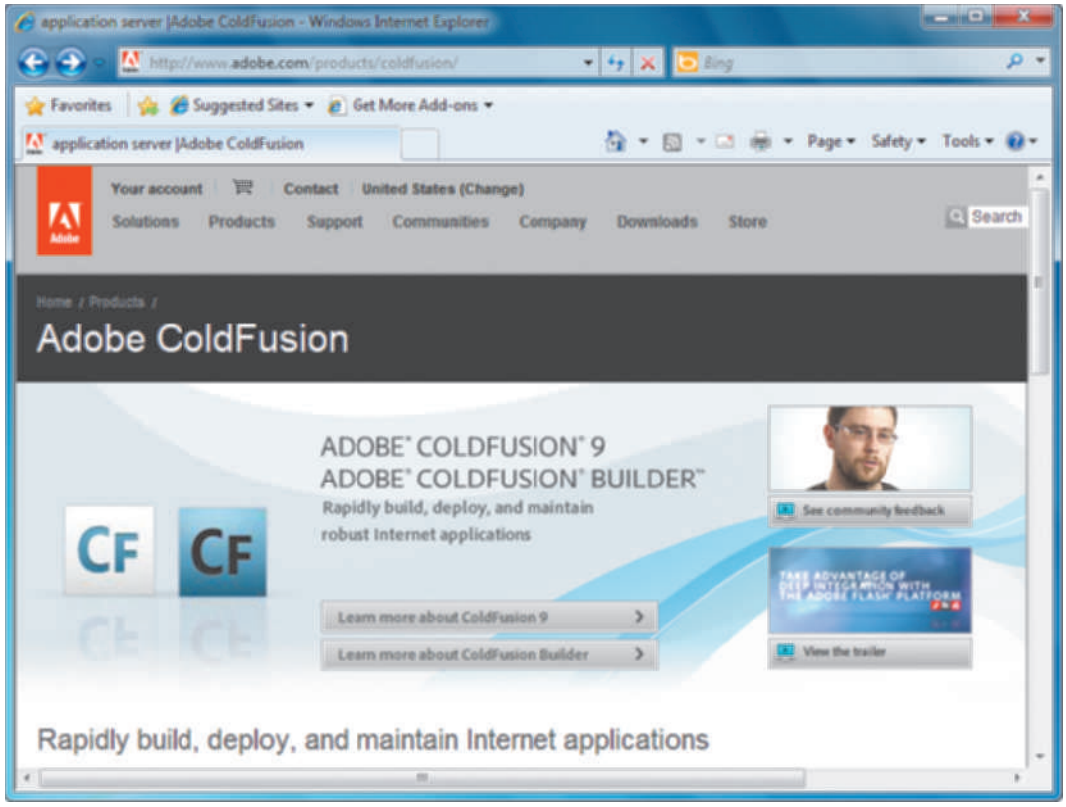


FIGURE 9-11 Adobe ColdFusion is a popular example of middleware.

Data Security

Web-based data must be secure, yet easily accessible to authorized users. To achieve this goal, well-designed systems provide security at three levels: the database itself, the Web server, and the telecommunication links that connect the components of the system.

Data security is discussed in this chapter and in Chapter 12, Managing System Support and Security.

Data Design Terminology

Using the concepts discussed in the previous section, a systems analyst can select a design approach and begin to construct the system. The first step is to understand data design terminology.

Definitions

Data design terms include entity, table, file, field, record, tuple, and key field. These terms are explained in the following sections.

ENTITY An **entity** is a person, place, thing, or event for which data is collected and maintained. For example, an online sales system may include entities named CUSTOMER, ORDER, PRODUCT, and SUPPLIER. When you prepared DFDs during the systems analysis phase, you identified various entities and data stores. Now you will consider the relationships among the entities.

TABLE OR FILE Data is organized into tables or files. A table, or file, contains a set of related records that store data about a specific entity. Tables and files are shown as two-dimensional structures that consist of vertical columns and horizontal rows. Each column represents a field, or characteristic of the entity, and each row represents a record, which is an individual instance, or occurrence of the entity. For example, if a company has 10,000 customers, the CUSTOMER table will include 10,000 records, each representing a specific customer.

Although they can have different meanings in a specific context, the terms *table* and *file* often can be used interchangeably.

FIELD A **field**, also called an **attribute**, is a single characteristic or fact about an entity. For example, a CUSTOMER entity might include the Customer ID, First Name, Last Name, Address, City, State, Zip, and E-mail Address.

A **common field** is an attribute that appears in more than one entity. Common fields can be used to link entities in various types of relationships.

RECORD A **record**, also called a **tuple** (rhymes with couple), is a set of related fields that describes one instance, or occurrence of an entity, such as one customer, one order, or one product. A record might have one or dozens of fields, depending on what information is needed.

Key Fields

During the systems design phase, you use **key fields** to organize, access, and maintain data structures. The four types of keys are primary keys, candidate keys, foreign keys, and secondary keys.

PRIMARY KEY A **primary key** is a field or combination of fields that uniquely and minimally identifies a particular member of an entity. For example, in a customer table the customer number is a unique primary key because no two customers can have the same customer number. That key also is minimal because it contains no information beyond what is needed to identify the customer. In a CUSTOMER table, a Customer ID might be used as a unique primary key. Customer ID is an example of a primary key based on a single field.

A primary key also can be composed of two or more fields. For example, if a student registers for three courses, his or her student number will appear in three records in the registration system. If one of those courses has 20 students, 20 separate records will exist for that course number — one record for each student who registered.

In the registration file, neither the student number nor the course ID is unique, so neither field can be a primary key. To identify a specific student in a specific course, the primary key must be a combination of student number and course ID. In that case, the primary key is called a **combination key**. A combination key also can be called a **composite key**, a **concatenated key**, or a **multivalued key**.

Figure 9-12 shows four different tables. The first three tables have single-field primary keys. Notice that in the fourth table, however, the primary key is a combination of two fields: STUDENT-NUMBER and COURSE-ID.

CANDIDATE KEY Sometimes you have a choice of fields or field combinations to use as the primary key. Any field that could serve as a primary key is called a **candidate key**. For example, if every employee has a unique employee number, then you could use either the employee number or the Social Security number as a primary key. Because you can designate only one field as a primary key, you should select the field that contains the least amount of data and is the easiest to use. Any field that is not a primary key or a candidate key is called a **nonkey field**.

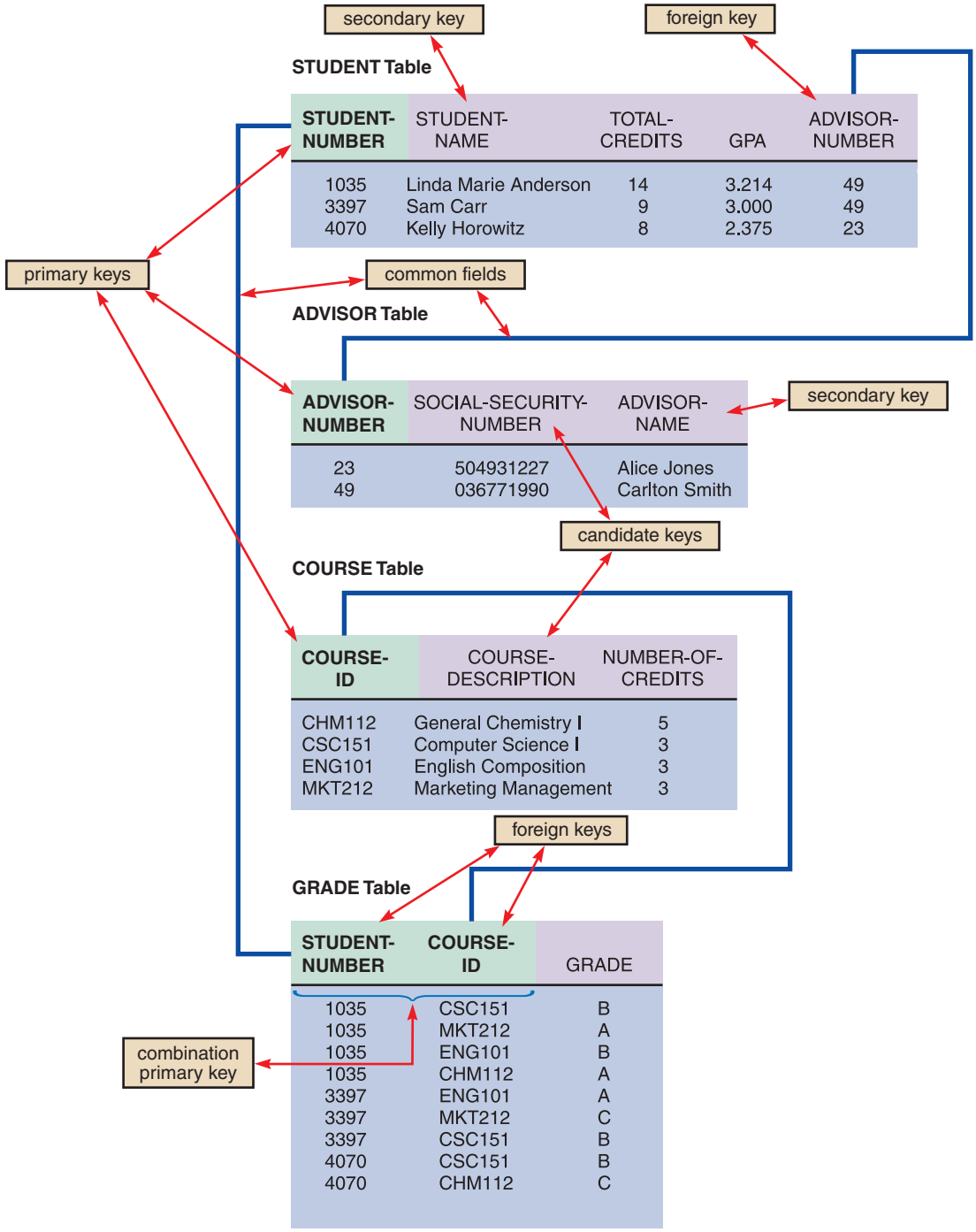


FIGURE 9-12 Examples of common fields, primary keys, candidate keys, foreign keys, and secondary keys.

The primary keys shown in Figure 9-12 also are candidate keys. Two other candidate keys exist: the SOCIAL-SECURITY-NUMBER field in the ADVISOR table and the COURSE-DESCRIPTION field in the COURSE table.

FOREIGN KEY Recall that a common field exists in more than one table and can be used to form a relationship, or link, between the tables. For example, in Figure 9-12, the ADVISOR-NUMBER field appears in both the STUDENT table and the ADVISOR table and joins the tables together. Notice that ADVISOR-NUMBER is a primary key in the ADVISOR table, where it uniquely identifies each advisor, and is a foreign key in

the STUDENT table. A **foreign key** is a field in one table that must match a primary key value in another table in order to establish the relationship between the two tables.

Unlike a primary key, a foreign key need not be unique. For example, Carlton Smith has advisor number 49. The value 49 must be a unique value in the ADVISOR table because it is the primary key, but 49 can appear any number of times in the STUDENT table, where the advisor number serves as a foreign key.

Figure 9-12 on the previous page also shows how two foreign keys can serve as a composite primary key in another table. Consider the GRADE table at the bottom of the figure. The two fields that form the primary key for the GRADE table are both foreign keys: the STUDENT-NUMBER field, which must match a student number in the STUDENT table, and the COURSE-ID field, which must match one of the course IDs in the COURSE table.

How can these two foreign keys serve as a primary key in the GRADE table? When you study the table, you will notice that student numbers and course IDs can appear any number of times, but the *combination* of a specific student and a specific course occurs only once. For example, student 1035 appears four times and course CSC151 appears three times — but there is only *one* combined instance of student 1035 *and* course CSC151. Because the combination of the specific student (1035) and the specific course (CSC151) is unique, it ensures that the grade (B) will be assigned to the proper student in the proper course.

SECONDARY KEY A **secondary key** is a field or combination of fields that can be used to access or retrieve records. Secondary key values are not unique. For example, if you need to access records for only those customers in a specific ZIP code, you would use the ZIP code field as a secondary key. Secondary keys also can be used to sort or display records in a certain order. For example, you could use the GPA field in a STUDENT file to display records for all students in grade point order.

ON THE WEB

To learn more about referential integrity, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Referential Integrity link.

The need for a secondary key arises because a table can have only one primary key. In a CUSTOMER file, the CUSTOMER-NUMBER is the primary key, so it must be unique. You might know a customer's name, but not the customer's number. For example, you might want to access a customer named James Morgan, but you do not know his customer number. If you search the table using the CUSTOMER-NAME field as a secondary key, you can retrieve the records for all customers named James Morgan and then select the correct one.

In Figure 9-12, student name and advisor names are identified as secondary keys, but other fields also could be used. For example, to find all students who have a particular advisor, you could use the ADVISOR-NUMBER field in the STUDENT file as a secondary key.

Referential Integrity

Validity checks can help avoid data input errors. One type of validity check, called **referential integrity**, is a set of rules that avoids data inconsistency and quality problems. In a relational database, referential integrity means that a foreign key value cannot be entered in one table unless it matches an existing primary key in another table. For example, referential integrity would prevent you from entering a customer order in an order table unless that customer already exists in the customer table. Without referential integrity, you might have an order called an **orphan**, because it had no related customer.

In the example shown in Figure 9-12 on page 403, referential integrity will not allow a user to enter an advisor number (foreign key value) in the STUDENT table unless a valid advisor number (primary key value) already exists in the ADVISOR table.

Referential integrity also can prevent the deletion of a record if the record has a primary key that matches foreign keys in another table. For example, suppose that an advisor resigns to accept a position at another school. You cannot delete the advisor from the ADVISOR table while records in the STUDENT file still refer to that advisor number. Otherwise, the STUDENT records would be orphans. To avoid the problem, students must be reassigned to other advisors by changing the value in the ADVISOR-NUMBER field; then the advisor record can be deleted.

When creating a relational database, you can build referential integrity into the design. Figure 9-13 shows a Microsoft Access screen that identifies a common field and allows the user to enforce referential integrity rules.

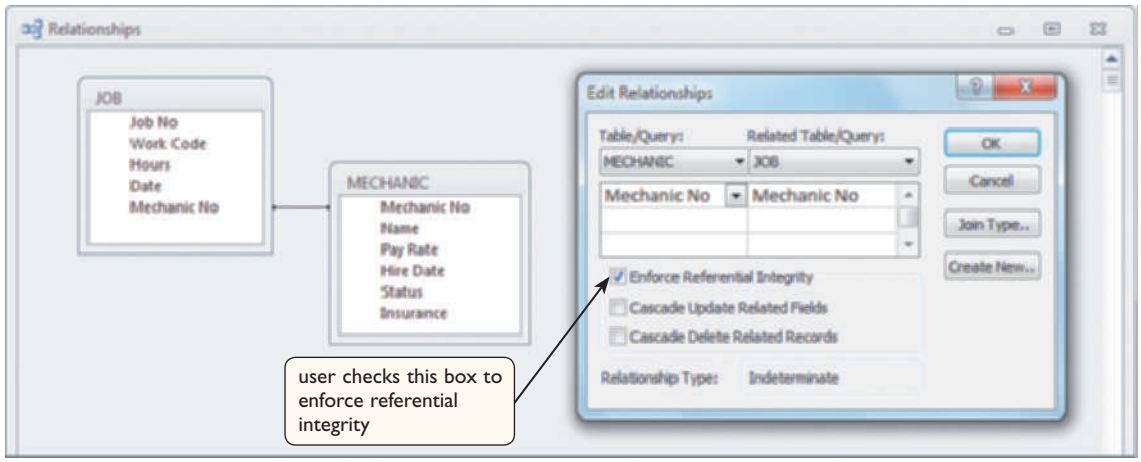


FIGURE 9-13 Microsoft Access allows a user to specify that referential integrity rules will be enforced in a relational database design.

VIDEO LEARNING SESSION: ENTITY-RELATIONSHIP DIAGRAMS

Video Learning Sessions can help you understand key concepts, practice your skills, and check your work. To access the sessions, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and navigate to the **Video Learning Sessions** for this book. In this session, you'll learn how to create an entity-relationship diagram (ERD) that shows the system entities and the nature of their relationships.

ENTITY-RELATIONSHIP DIAGRAMS

Recall that an entity is a person, place, thing, or event for which data is collected and maintained. For example, entities might be customers, sales regions, products, or orders. An information system must recognize the relationships among entities. For example, a *customer* entity can have several instances of an *order* entity, and an *employee* entity can have one instance, or none, of a *spouse* entity.

ON THE WEB

To learn more about entity-relationship diagrams, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Entity-Relationship Diagrams link.

An **entity-relationship diagram** (ERD) is a model that shows the logical relationships and interaction among system entities. An ERD provides an overall view of the system and a blueprint for creating the physical data structures.

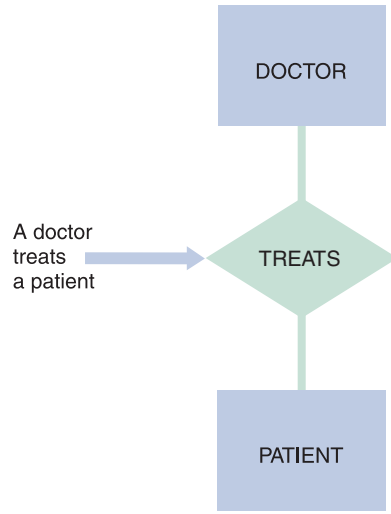


FIGURE 9-14 In an entity-relationship diagram, entities are labeled with singular nouns and relationships are labeled with verbs. The relationship is interpreted as a simple English sentence.

Drawing an ERD

The first step is to list the entities that you identified during the systems analysis phase and to consider the nature of the relationships that link them. At this stage, you can use a simplified method to show the relationships between entities.

Although there are different ways to draw ERDs, a popular method is to represent entities as rectangles and relationships as diamond shapes. The entity rectangles are labeled with singular nouns, and the relationship diamonds are labeled with verbs, usually in a top-to-bottom and left-to-right fashion. For example, in Figure 9-14, a doctor entity *treats* a patient entity. Unlike data flow diagrams, entity-relationship diagrams depict relationships, not data or information flows.

Types of Relationships

Three types of relationships can exist between entities: one-to-one, one-to-many, and many-to-many.

A **one-to-one relationship**, abbreviated **1:1**, exists when exactly one of the second entity occurs for each instance of the first entity. Figure 9-15 shows examples of several 1:1 relationships. A number 1 is placed alongside each of the two connecting lines to indicate the 1:1 relationship.

A **one-to-many relationship**, abbreviated **1:M**, exists when one occurrence of the first entity can relate to many instances of the second entity, but each instance of the second entity can associate with only one instance of the first entity. For example, the relationship between DEPARTMENT and EMPLOYEE is one-to-many: One department can have many employees, but each employee works in only one department at a time. Figure 9-16 shows several 1:M relationships. The line connecting the *many* entity is labeled with the letter M, and the number 1 labels the other connecting line. How many is *many*? The first 1:M relationship shown in Figure 9-16 shows the entities INDIVIDUAL and AUTOMOBILE. One individual might own five automobiles, or one, or none. Thus, *many* can mean any number, including zero.

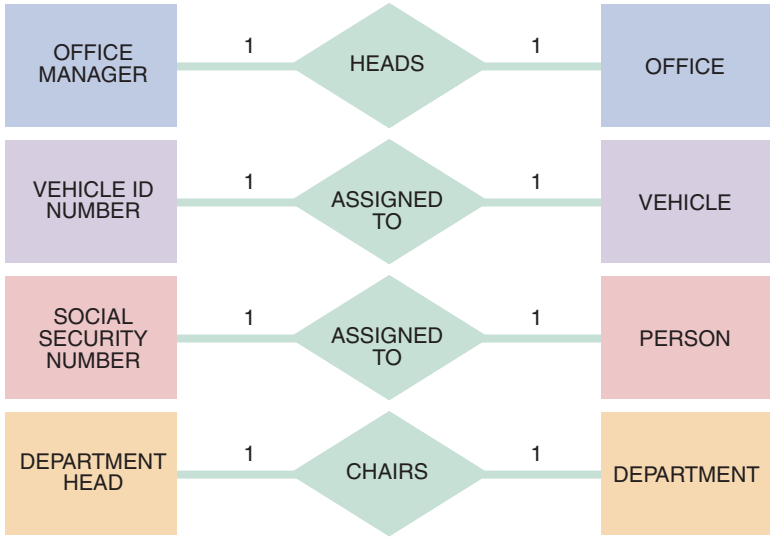


FIGURE 9-15 Examples of one-to-one (1:1) relationships.

A **many-to-many relationship**, abbreviated **M:N**, exists when one instance of the first entity can relate to many instances of the second entity, and one instance of the second entity can relate to many instances of the first entity. The relationship between STUDENT and CLASS, for example, is many-to-many — one student can take many classes, and one class can have many students enrolled. Figure 9-17 shows several M:N entity-relationships. One of the connecting lines is labeled with the letter M, and the letter N labels the other connection.

Notice that an M:N relationship is different from 1:1 or 1:M relationships because the event or transaction that links the two entities is actually a third entity, called an **associative entity** that has its own characteristics. In the first example in Figure 9-17, the ENROLLS IN symbol represents a REGISTRATION entity that records each instance of a specific student enrolling in a specific course. Similarly, the RESERVES SEAT ON symbol represents a RESERVATION entity that records each instance of a specific passenger reserving a seat on a specific flight. In the third example, the LISTS symbol represents an ORDER-LINE entity that records each instance of a specific product listed in a specific customer order.

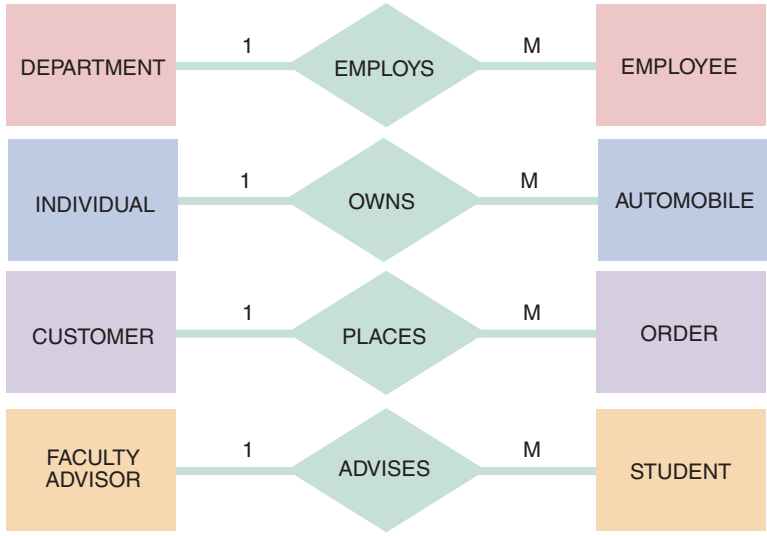


FIGURE 9-16 Examples of one-to-many (1:M) relationships.

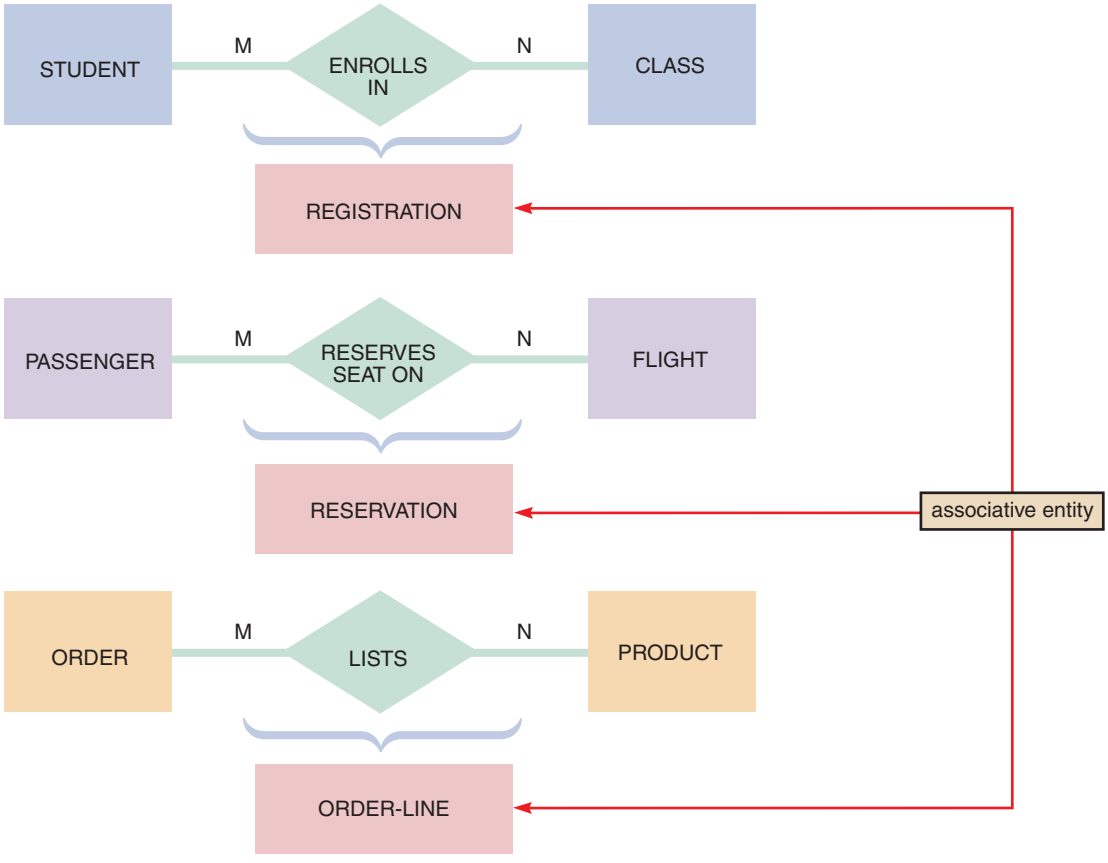


FIGURE 9-17 Examples of many-to-many (M:N) relationships. Notice that the event or transaction that links the two entities is an associative entity with its own set of attributes and characteristics.

Figure 9-18 shows an ERD for a sales system. Notice the various entities and relationships shown in the figure, including the associative entity named ORDER-LINE. The detailed nature of these relationships is called cardinality. As an analyst, you must understand cardinality in order to create a data design that accurately reflects all relationships among system entities.

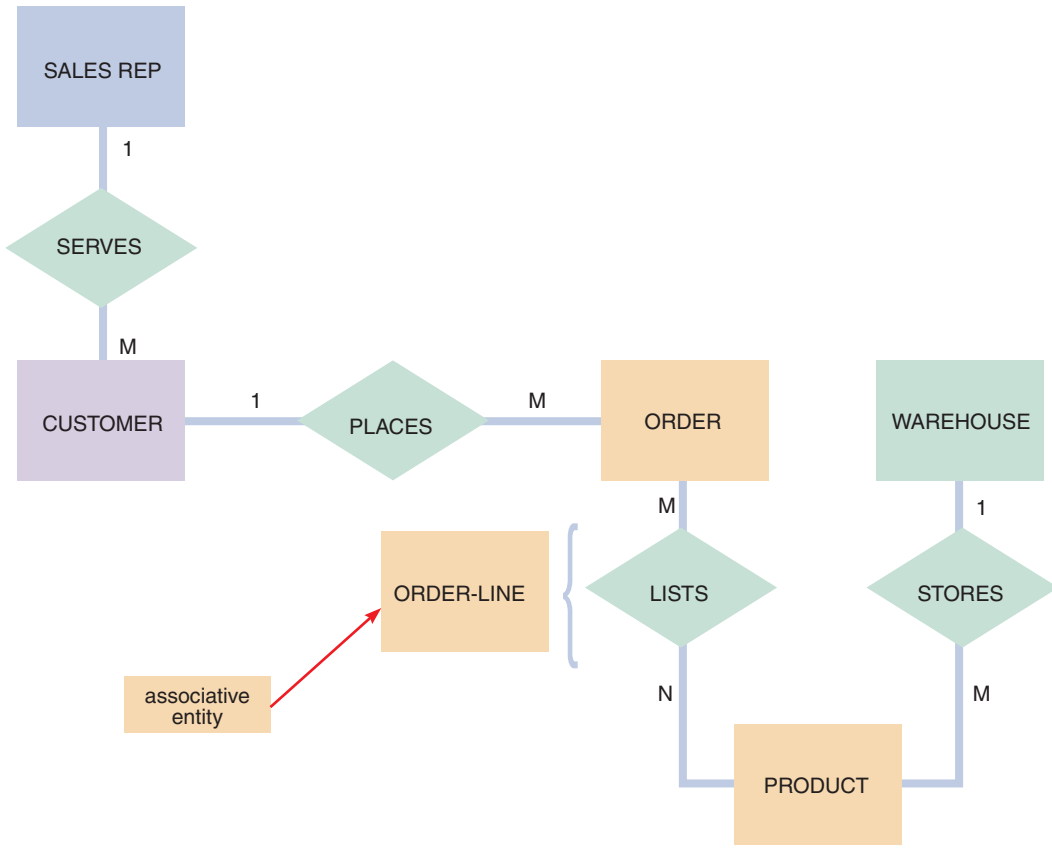


FIGURE 9-18 An entity-relationship diagram for SALES REP, CUSTOMER, ORDER, PRODUCT, and WAREHOUSE. Notice that the ORDER and PRODUCT entities are joined by an associative entity named ORDER-LINE.

Cardinality

ON THE WEB

To learn more about cardinality, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Cardinality link.

After an analyst draws an initial ERD, he or she must define the relationships in more detail by using a technique called cardinality. **Cardinality** describes the numeric relationship between two entities and shows how instances of one entity relate to instances of another entity. For example, consider the relationship between two entities: CUSTOMER and ORDER. One customer can have one order, many orders, or none, but each order must have one and only one customer. An analyst can model this interaction by adding **cardinality notation**, which uses special symbols to represent the relationship.

A common method of cardinality notation is called **crow's foot notation** because of the shapes, which include circles, bars, and symbols, that indicate various possibilities. A single bar indicates one, a double bar indicates one and only one, a circle indicates zero, and a crow's foot indicates many. Figure 9-19 shows various cardinality symbols, their meanings, and the UML representations of the relationships. As you learned in Chapter 4, the **Unified Modeling Language** (**UML**) is a widely used method of visualizing and documenting software systems design.

In Figure 9-20, four examples of cardinality notation are shown. In the first example, one and only one CUSTOMER can place anywhere from zero to many of the ORDER entity. In the second example, one and only one ORDER can include one ITEM ORDERED or many. In the third example, one and only one EMPLOYEE can have one SPOUSE or none. In the fourth example, one EMPLOYEE, or many employees, or none, can be assigned to one PROJECT, or many projects, or none.

Most CASE products support the drawing of ERDs from entities in the data repository. Figure 9-21 on the next page shows part of a library system ERD drawn using the Visible Analyst CASE tool. Notice that crow's foot notation is used to show the nature of the relationships, which are described in both directions.

Now that you understand database elements and their relationships, you can start designing tables. The first step is the normalization of your table designs, which is described next.

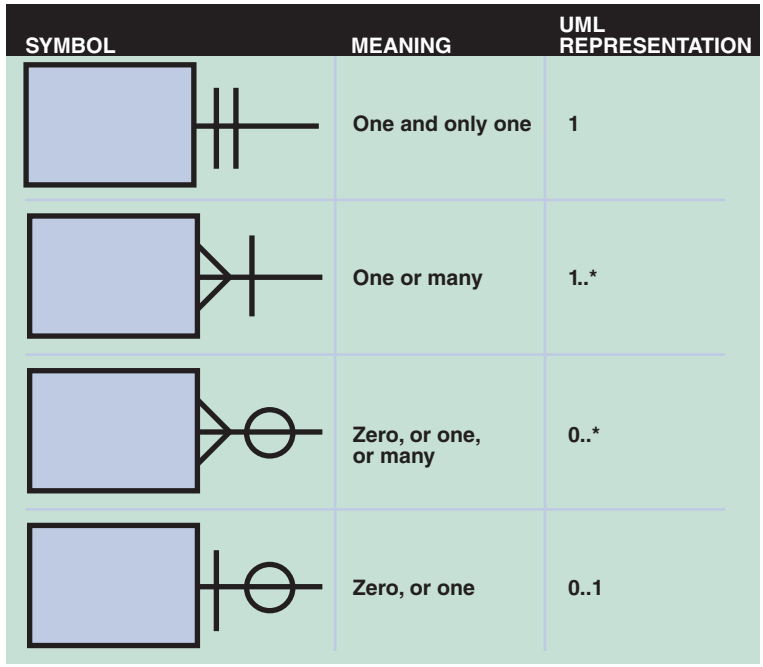


FIGURE 9-19 Crow's foot notation is a common method of indicating cardinality. The four examples show how you can use various symbols to describe the relationships between entities.

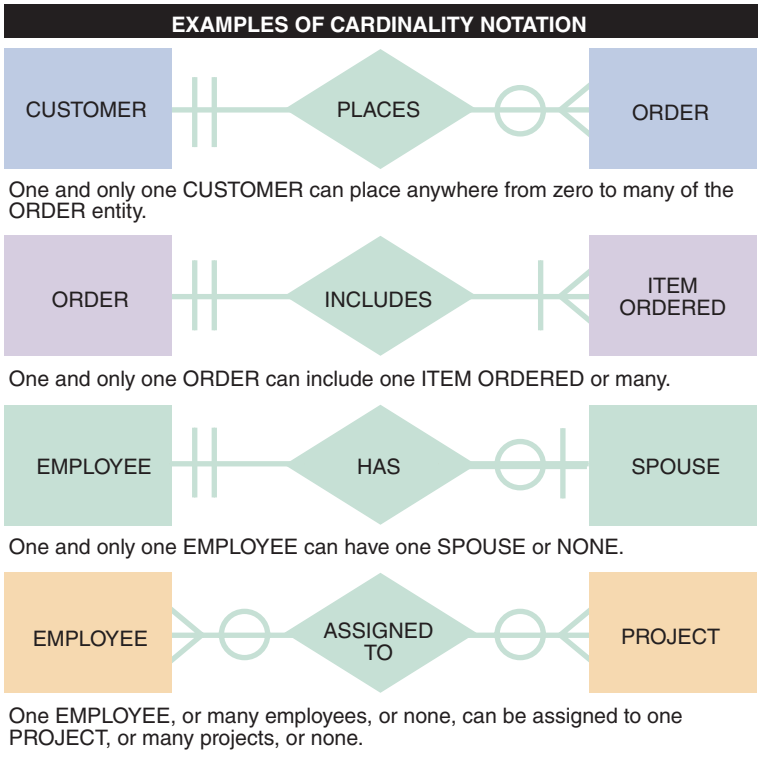


FIGURE 9-20 In the first example of cardinality notation, one and only one CUSTOMER can place anywhere from zero to many of the ORDER entity. In the second example, one and only one ORDER can include one ITEM ORDERED or many. In the third example, one and only one EMPLOYEE can have one SPOUSE or none. In the fourth example, one EMPLOYEE, or many employees, or none, can be assigned to one PROJECT, or many projects, or none.

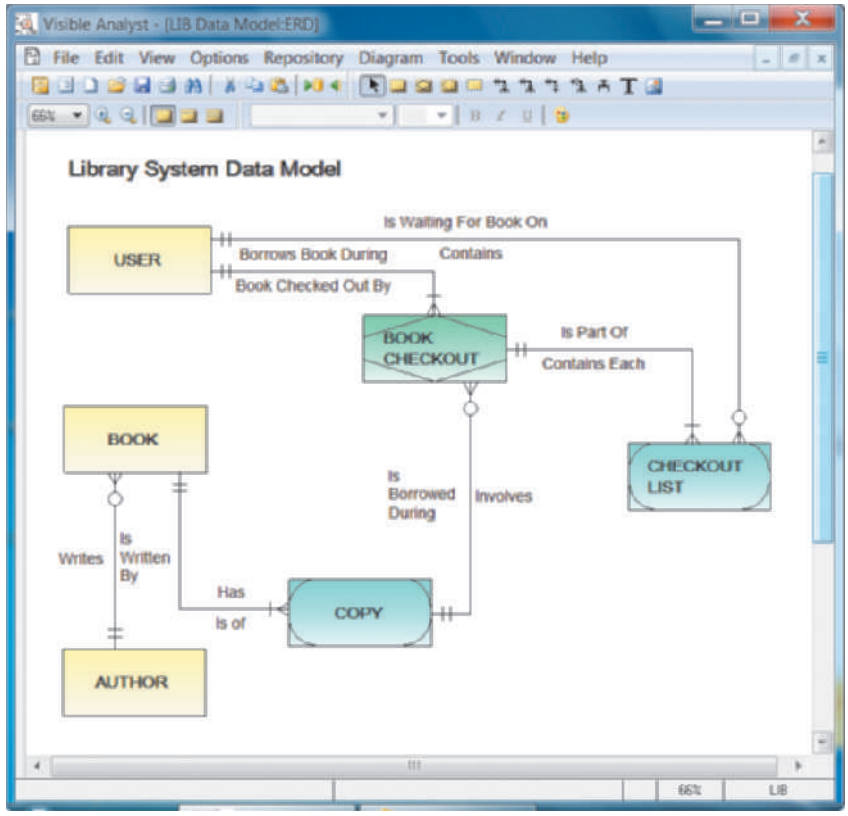


FIGURE 9-21 An ERD for a library system drawn with Visible Analyst. Notice that crow's foot notation has been used and relationships are described in both directions.

CASE IN POINT 9.1: TopText Publishing

TopText Publishing is a textbook publishing company with a headquarters location, a warehouse, and three sales offices that each have a sales manager and sales reps. TopText sells to schools, colleges, and individual customers. Many authors write more than one book for TopText, and some books are written by more than one author. TopText maintains an active list of more than 100 books, each identified by a universal code called an ISBN number. You have been asked to draw an ERD for the TopText information system, and to include cardinality notation.

Normalization

ON THE WEB

To learn more about normalization, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Normalization link.

Normalization is the process of creating table designs by assigning specific fields or attributes to each table in the database. A **table design** specifies the fields and identifies the primary key in a particular table or file. Working with a set of initial table designs, you use normalization to develop an overall database design that is simple, flexible, and free of data redundancy. Normalization involves applying a set of rules that can help you identify and correct inherent problems and complexities in your table designs. The concept of normalization is based on the work of Edgar Codd, a British computer scientist who formulated the basic principles of relational database design.

The normalization process typically involves four stages: unnormalized design, first normal form, second normal form, and third normal form. The three normal forms constitute a progression in which third normal form represents the best design. Most business-related databases must be designed in third normal form.

Standard Notation Format

Designing tables is easier if you use a **standard notation format** to show a table's structure, fields, and primary key. The standard notation format in the following examples starts with the name of the table, followed by a parenthetical expression that contains the field names separated by commas. The primary key field(s) is underlined, like this:

NAME (FIELD 1, FIELD 2, FIELD 3)

Repeating Groups and Unnormalized Designs

During data design, you must be able to recognize a repeating group of fields. A **repeating group** is a set of one or more fields that can occur any number of times in a single record, with each occurrence having different values.

Repeating groups often occur in manual documents prepared by users. For example, consider a school registration form with the student's information at the top of the form, followed by a list of courses the student is taking. If you were to design a table based on this registration form, the courses would represent a repeating group of values for each student.

An example of a repeating group is shown in Figure 9-22. The first two records in the ORDER table contain multiple products, which represent a repeating group of fields. Notice that in addition to the order number and date, the records with multiple products contain repetitions of the product number, description, and number ordered. You can think of a repeating group as a set of child (subsidiary) records contained within the parent (main) record.

A table design that contains a repeating group is called **unnormalized**. The standard notation method for representing an unnormalized design is to enclose the repeating group of fields within a second set of parentheses. An example of an unnormalized table would look like this:

NAME (FIELD 1, FIELD 2, FIELD 3, (REPEATING FIELD 1, REPEATING FIELD 2))

Now review the unnormalized ORDER table design shown in Figure 9-22. Following the notation guidelines, you can describe the design as follows:

ORDER (ORDER-NUM, ORDER-DATE, (PRODUCT-NUM, PRODUCT-DESC, NUM-ORDERED))

The notation indicates that the ORDER table design contains five fields, which are listed within the outer parentheses. The ORDER-NUM field is underlined to show that it is the primary key. The PRODUCT-NUM, PRODUCT-DESC, and NUM-ORDERED fields are enclosed within an inner set of parentheses to indicate that they are fields within a repeating group. Notice that PRODUCT-NUM also is underlined because it acts as the

ORDER (UNNORMALIZED)

primary key

primary key for repeating group

RECORD#	ORDER-NUM	ORDER-DATE	PRODUCT-NUM	PRODUCT-DESC	NUM-ORDERED
1	40311	03112011	304	All-purpose gadget	7
			633	Assembly	1
			684	Super gizmo	4
2	40312	03112011	128	Steel widget	12
			304	All-purpose gadget	3
3	40313	03122011	304	All-purpose gadget	144

repeating groups

FIGURE 9-22 In the ORDER table design, records 1 and 2 have repeating groups because they contain several products. ORDER-NUM is the primary key for the ORDER table, and PRODUCT-NUM serves as a primary key for the repeating group. Because it contains a repeating group, the ORDER table design is unnormalized.

primary key of the repeating group. If a customer orders three different products in one order, then the fields PRODUCT-NUM, PRODUCT-DESC, and NUM-ORDERED repeat three times, as shown in Figure 9-22 on the previous page.

VIDEO LEARNING SESSION: First Normal Form

Video Learning Sessions can help you understand key concepts, practice your skills, and check your work. To access the sessions, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and navigate to the **Video Learning Sessions** for this book. This is the first of three Video Learning Sessions about data normalization. In this session, you'll learn how to transform unnormalized data into first normal form (1NF).

First Normal Form

A table is in **first normal form** (**1NF**) if it does not contain a repeating group. To convert an unnormalized design to 1NF, you must expand the table's primary key to include the primary key of the repeating group.

For example, in the ORDER table shown in Figure 9-22, the repeating group consists of three fields: PRODUCT-NUM, PRODUCT-DESC, and NUM-ORDERED. Of the three fields, only PRODUCT-NUM can be a primary key because it uniquely identifies each instance of the repeating group. The product description cannot be a primary key because it might or might not be unique. For example, a company might sell a large number of parts with the same descriptive name, such as *washer*, relying on a coded part number to identify uniquely each washer size.

When you expand the primary key of ORDER table to include PRODUCT-NUM, you eliminate the repeating group and the ORDER table is now in 1NF, as shown:

ORDER (ORDER-NUM, ORDER-DATE, PRODUCT-NUM, PRODUCT-DESC, NUM-ORDERED)

Figure 9-23 shows the ORDER table in 1NF. Notice that when you eliminate the repeating group, additional records emerge — one for each combination of a specific order and a specific product. The result is more records, but a greatly simplified design. In the new version, the repeating group for order number 40311 has become three separate records, and the repeating group for order number 40312 has become two separate records. Therefore, when a table is in 1NF, each record stores data about a single instance of a specific order and a specific product.

Also notice that the 1NF design shown in Figure 9-23 has a combination primary key. The primary key of the 1NF design cannot be the ORDER-NUM field alone, because the order number does not uniquely identify each product in a multiple-item order. Similarly, PRODUCT-NUM cannot be the primary key, because it appears more than once if several orders include the same product. Because each record must reflect a specific product in a specific order, you need *both* fields, ORDER-NUM and PRODUCT-NUM, to identify a single record uniquely. Therefore, the primary key is the *combination* of two fields: ORDER-NUM and PRODUCT-NUM.

ORDER IN 1NF

combination primary key

RECORD#	ORDER-NUM	ORDER-DATE	PRODUCT-NUM	PRODUCT-DESC	NUM-ORDERED
1	40311	03112011	304	All-purpose gadget	7
2	40311	03112011	633	Assembly	1
3	40311	03112011	684	Super gizmo	4
4	40312	03112011	128	Steel widget	12
5	40312	03112011	304	All-purpose gadget	3
6	40313	03122011	304	All-purpose gadget	144

repeating groups have been eliminated

FIGURE 9-23 The ORDER table as it appears in 1NF. The repeating groups have been eliminated. Notice that the repeating group for order 40311 has become three separate records, and the repeating group for order 40312 has become two separate records. The 1NF primary key is a combination of ORDER-NUM and PRODUCT-NUM, which uniquely identifies each record.

VIDEO LEARNING SESSION: SECOND NORMAL FORM

Video Learning Sessions can help you understand key concepts, practice your skills, and check your work. To access the sessions, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and navigate to the **Video Learning Sessions** for this book. This is the second of three Video Learning Sessions about data normalization. In this session, you'll learn how to transform data from first normal form (1NF) to second normal form (2NF).

Second Normal Form

To understand second normal form (2NF), you must understand the concept of functional dependence. For example, *Field A* is **functionally dependent** on *Field B* if the value of *Field A* depends on *Field B*. For example, in Figure 9-23, the ORDER-DATE value is functionally dependent on the ORDER-NUM, because for a specific order number, there can be only one date. In contrast, a product description is *not* dependent on the order number. For a particular order number, there might be several product descriptions — one for each item ordered.

A table design is in **second normal form** (**2NF**) if it is in 1NF *and* if all fields that are not part of the primary key are functionally dependent on the *entire* primary key. If any field in a 1NF table depends on only one of the fields in a combination primary key, then the table is not in 2NF.

Notice that if a 1NF design has a primary key that consists of only one field, the problem of partial dependence does not arise — because the entire primary key is a single field. Therefore, a 1NF table with a single-field primary key is automatically in 2NF.

Now reexamine the 1NF design for the ORDER table shown in Figure 9-23:

ORDER (ORDER-NUM, ORDER-DATE, PRODUCT-NUM, PRODUCT-DESC, NUM-ORDERED)

Recall that the primary key is the combination of the order number and the product number. The NUM-ORDERED field depends on the *entire* primary key, because NUM-ORDERED refers to a specific product number *and* a specific order number. In contrast, the ORDER-DATE field depends on the order number, which is only a part of the

primary key. Similarly, the PRODUCT-DESC field depends on the product number, which also is only a part of the primary key. Because some fields are not dependent on the *entire* primary key, the design is not in 2NF.

A standard process exists for converting a table from 1NF to 2NF. The objective is to break the original table into two or more new tables and reassign the fields so that each nonkey field will depend on the entire primary key in its table. To accomplish this, you follow these steps:

1. First, create and name a separate table for each field in the existing primary key. For example, in Figure 9-23 on the previous page, the ORDER table's primary key has two fields, ORDER-NUM and PRODUCT-NUM, so you must create two tables. The ellipsis (…) indicates that fields will be assigned later. The result is:

   ORDER (ORDER-NUM,...)

   PRODUCT (PRODUCT-NUM,...)

2. Next, create a new table for each possible combination of the original primary key fields. In the Figure 9-23 example, you would create and name a new table with a combination primary key of ORDER-NUM and PRODUCT-NUM. This table describes individual lines in an order, so it is named ORDER-LINE, as shown:

   ORDER-LINE (ORDER-NUM, PRODUCT-NUM,...)

3. Finally, study the three tables and place each field with its appropriate primary key, which is the minimal key on which it functionally depends. When you finish placing all the fields, remove any table that did not have any additional fields assigned to it. The remaining tables are the 2NF version of your original table. In the Figure 9-23 example, the three tables would be shown as:

   ORDER (ORDER-NUM, ORDER-DATE)

   PRODUCT (PRODUCT-NUM, PRODUCT-DESC)

   ORDER-LINE (ORDER-NUM, PRODUCT-NUM, NUM-ORDERED)

Figure 9-24 shows the 2NF table designs. By following the steps, you have converted the original 1NF table into three 2NF tables.

Why is it important to move from 1NF to 2NF? Four kinds of problems are found with 1NF designs that do not exist in 2NF:

- Consider the work necessary to change a particular product's description. Suppose 500 current orders exist for product number 304. Changing the product description involves modifying 500 records for product number 304. Updating all 500 records would be cumbersome and expensive.
- 1NF tables can contain inconsistent data. Because someone must enter the product description in each record, nothing prevents product number 304 from having different product descriptions in different records. In fact, if product number 304 appears in a large number of order records, some of the matching product descriptions might be inaccurate or improperly spelled. Even the presence or absence of a hyphen in the orders for *All-purpose gadget* would create consistency problems. If a data entry person must enter a term such as *IO1 Queue Controller* numerous times, it certainly is possible that some inconsistency will result.
- Adding a new product is a problem. Because the primary key must include an order number and a product number, you need values for both fields in order to add a record. What value do you use for the order number when you want to add a new product that has not been ordered by any customer? You could use a dummy order number, and then replace it with a real order number when the product is ordered to solve the problem, but that solution also creates difficulties.

ORDER IN 2NF — primary key

RECORD#	ORDER-NUM	ORDER-DATE
1	40311	03112011
2	40312	03112011
3	40313	03122011

PRODUCT IN 2NF — primary key

RECORD#	PRODUCT-NUM	PRODUCT-DESC
1	128	Steel widget
2	304	All-purpose gadget
3	633	Assembly
4	684	Super gizmo

ORDER-LINE IN 2NF — primary key based on combination of two fields

RECORD#	ORDER-NUM	PRODUCT-NUM	NUM-ORDERED
1	40311	304	7
2	40311	633	1
3	40311	684	4
4	40312	128	12
5	40312	304	3
6	40313	304	144

FIGURE 9-24 ORDER, PRODUCT, and ORDER-LINE tables in 2NF. All fields are functionally dependent on the primary key.

- Deleting a product is a problem. If all the related records are deleted once an order is filled and paid for, what happens if you delete the only record that contains product number 633? The information about that product number and its description is lost.

Has the 2NF design eliminated all potential problems? To change a product description, now you can change just one PRODUCT record. Multiple, inconsistent values for the product description are impossible because the description appears in only one location. To add a new product, you simply create a new PRODUCT record, instead of creating a dummy order record. When you remove the last ORDER-LINE record for a particular product number, you do not lose that product number and its description because the PRODUCT record still exists. The four potential problems are eliminated, and the three 2NF designs are superior to both the original unnormalized table and the 1NF design.

VIDEO LEARNING SESSION: THIRD NORMAL FORM

Video Learning Sessions can help you understand key concepts, practice your skills, and check your work. To access the sessions, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and navigate to the **Video Learning Sessions** for this book. This is the third of three Video Learning Sessions about data normalization. In this session, you'll learn how to transform data from second normal form (2NF) to third normal form (3NF).

Third Normal Form

A popular rule of thumb is that a design is in 3NF if every nonkey field depends on *the key, the whole key, and nothing but the key*. As you will see, a 3NF design avoids redundancy and data integrity problems that still can exist in 2NF designs.

Consider the following CUSTOMER table design, as shown in Figure 9-25:

CUSTOMER (CUSTOMER-NUM, CUSTOMER-NAME, ADDRESS, SALES-REP-NUM, SALES-REP-NAME)

The table is in 1NF because it has no repeating groups. The design also is in 2NF because the primary key is a single field. But the table still has four potential problems similar to the four 1NF problems described earlier. Changing the name of a sales rep still requires changing every record in which that sales rep name appears. Nothing about the design prohibits a particular sales rep from having different names in different records. In addition, because the sales rep name is included in the CUSTOMER table, you must create a dummy CUSTOMER record to add a new sales rep who has not yet been assigned any customers. Finally, if you delete all the records for customers of sales rep number 22, you will lose that sales rep's number and name.

Those potential problems are caused because the design is not in 3NF. A table design is in **third normal form** (**3NF**) if it is in 2NF and if no nonkey field is dependent on another nonkey field. Remember that a nonkey field is a field that is not a candidate key for the primary key. The CUSTOMER example in Figure 9-25 is not in 3NF because one nonkey field, SALES-REP-NAME, depends on another nonkey field, SALES-REP-NUM.

To convert the table to 3NF, you must remove all fields from the 2NF table that depend on another nonkey field and place them in a new table that uses the nonkey field as a primary key. In the CUSTOMER example, the SALES-REP-NAME field depends on another field, SALES-REP-NUM, which is not part of the primary key. Therefore, to reach 3NF, you must remove SALES-REP-NAME and place it into a new table that uses SALES-REP-NUM as the primary key. As shown in Figure 9-26, the third normal form produces two separate tables:

CUSTOMER (CUSTOMER-NUM, CUSTOMER-NAME, ADDRESS, SALES-REP-NUM)

SALES-REP (SALES-REP-NUM, SALES-REP-NAME)

in 2NF, the nonkey field SALES-REP-NAME is functionally dependent on another nonkey field, SALES-REP-NUM

CUSTOMER IN 2NF

RECORD#	CUSTOMER-NUM	CUSTOMER-NAME	ADDRESS	SALES-REP-NUM	SALES-REP-NAME
1	108	Benedict, Louise	San Diego, CA	41	Kaplan, James
2	233	Corelli, Helen	Nashua, NH	22	McBride, Jon
3	254	Gomez, J.P.	Butte, MT	38	Stein, Ellen
4	431	Lee, M.	Snow Camp, NC	74	Roman, Harold
5	779	Paulski, Diane	Lead, SD	38	Stein, Ellen
6	800	Zuider, Z.	Greer, SC	74	Roman, Harold

FIGURE 9-25 2NF design for the CUSTOMER table.

CUSTOMER IN 3NF

RECORD#	CUSTOMER-NUM	CUSTOMER-NAME	ADDRESS	SALES-REP-NUM
1	108	Benedict, Louise	San Diego, CA	41
2	233	Corelli, Helen	Nashua, NH	22
3	254	Gomez, J.P.	Butte, MT	38
4	431	Lee, M.	Snow Camp, NC	74
5	779	Paulski, Diane	Lead, SD	38
6	800	Zuider, Z.	Greer, SC	74

in 3NF, no nonkey field is dependent on another nonkey field

SALES-REP IN 3NF

RECORD#	SALES-REP-NUM	SALES-REP-NAME
1	22	McBride, Jon
2	38	Stein, Ellen
3	41	Kaplan, James
4	74	Roman, Harold

FIGURE 9-26 When the CUSTOMER table is transformed from 2NF to 3NF, the result is two tables: CUSTOMER and SALES-REP.

A Normalization Example

To show the normalization process, consider the familiar situation in Figure 9-27, which depicts several entities in a school advising system: ADVISOR, COURSE, and STUDENT. The relationships among the three entities are shown in the ERD in Figure 9-28 on the next page. The following sections discuss normalization rules for these three entities.

Before you start the normalization process, you notice that the STUDENT table contains fields that relate to the ADVISOR and COURSE entities, so you decide to begin with the initial design for the STUDENT table, which is shown in Figure 9-29. Notice that the table design includes the student number, student name, total credits taken, grade point average (GPA), advisor number, advisor name, and, for every course the student has taken, the course number, course description, number of credits, and grade received.

The STUDENT table in Figure 9-29 on the next page is unnormalized, because it has a repeating group. The STUDENT table design can be written as:

STUDENT (STUDENT-NUMBER, STUDENT-NAME, TOTAL-CREDITS, GPA, ADVISOR-NUMBER, ADVISOR-NAME, (COURSE-NUMBER, COURSE-DESC, NUM-CREDITS, GRADE))

To convert the STUDENT record to 1NF, you must expand the primary key to include the key of the repeating group, producing:

STUDENT (STUDENT-NUMBER, STUDENT-NAME, TOTAL-CREDITS, GPA, ADVISOR-NUMBER, ADVISOR-NAME, COURSE-NUMBER, COURSE-DESC, NUM-CREDITS, GRADE)

FIGURE 9-27 A faculty advisor, who represents an entity, can advise many students, each of whom can register for one or many courses.

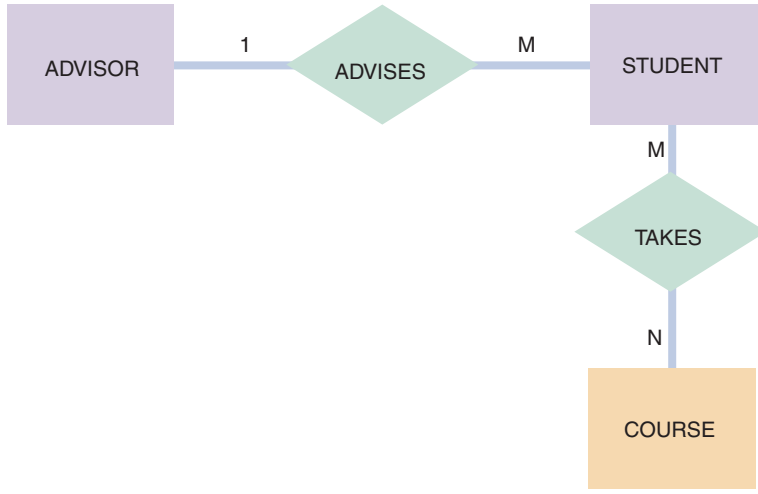


FIGURE 9-28 An initial entity-relationship diagram for ADVISOR, STUDENT, and COURSE.

STUDENT

STUDENT-NUMBER	STUDENT-NAME	TOTAL-CREDITS	GPA	ADVISOR-NUMBER	ADVISOR-NAME	COURSE-NUMBER	COURSE-DESC	NUM-CREDITS	GRADE
1035	Linda	47	3.647	49	Smith	CSC151	Computer Science I	4	B
						MKT212	Marketing Management	3	A
						ENG101	English Composition	3	B
						CHM112	General Chemistry I	4	A
						BUS105	Introduction to Business	2	A
3397	Sam	29	3.000	49	Smith	ENG101	English Composition	3	A
						MKT212	Marketing Management	3	C
						CSC151	Computer Science I	4	B
4070	Kelly	14	2.214	23	Jones	CSC151	Computer Science I	4	B
						CHM112	General Chemistry I	4	C
						ENG101	English Composition	3	C
						BUS105	Introduction to Business	2	C

repeating groups

FIGURE 9-29 The STUDENT table is unnormalized because it contains a repeating group that represents the courses each student has taken.

Figure 9-30 shows the 1NF version of the sample STUDENT data. Do any of the fields in the 1NF STUDENT record depend on only a portion of the primary key? The student name, total credits, GPA, advisor number, and advisor name all relate only to the student number and have no relationship to the course number. The course description depends on the course number, but not on the student number. Only the GRADE field depends on the entire primary key.

Following the 1NF – 2NF conversion process described earlier, you would create a new table for each field and combination of fields in the primary key, and place the other fields with their appropriate key. The result is:

STUDENT (STUDENT-NUMBER, STUDENT-NAME, TOTAL-CREDITS, GPA, ADVISOR-NUMBER, ADVISOR-NAME)

COURSE (COURSE-NUMBER, COURSE-DESC, NUM-CREDITS)

GRADE (STUDENT-NUMBER, COURSE-NUMBER, GRADE)

You now have converted the original 1NF STUDENT table to three tables, all in 2NF. In each table, every nonkey field depends on the entire primary key.

Figure 9-31 on page 420 shows the 2NF STUDENT, COURSE, and GRADE designs and sample data. Are all three tables in 3NF? The COURSE and GRADE are in 3NF. STUDENT is not in 3NF, however, because the ADVISOR-NAME field depends on the

STUDENT

STUDENT-NUMBER	STUDENT-NAME	TOTAL-CREDITS	GPA	ADVISOR-NUMBER	ADVISOR-NAME	COURSE-NUMBER	COURSE-DESC	NUM-CREDITS	GRADE
1035	Linda	47	3.647	49	Smith	CSC151	Computer Science I	4	B
1035	Linda	47	3.647	49	Smith	MKT212	Marketing Management	3	A
1035	Linda	47	3.647	49	Smith	ENG101	English Composition	3	B
1035	Linda	47	3.647	49	Smith	CHM112	General Chemistry I	4	A
1035	Linda	47	3.647	49	Smith	BUS105	Introduction to Business	2	A
3397	Sam	29	3.000	49	Smith	ENG101	English Composition	3	A
3397	Sam	29	3.000	49	Smith	MKT212	Marketing Management	3	C
3397	Sam	29	3.000	49	Smith	CSC151	Computer Science I	4	B
4070	Kelly	14	2.214	23	Jones	CSC151	Computer Science I	4	B
4070	Kelly	14	2.214	23	Jones	CHM112	General Chemistry I	4	C
4070	Kelly	14	2.214	23	Jones	ENG101	English Composition	3	C
4070	Kelly	14	2.214	23	Jones	BUS105	Introduction to Business	2	C

FIGURE 9-30 The STUDENT table in 1NF. Notice that the primary key has been expanded to include STUDENT-NUMBER and COURSE-NUMBER. Also, the repeating group has been eliminated.

ADVISOR-NUMBER field, which is not part of the STUDENT primary key. To convert STUDENT to 3NF, you remove the ADVISOR-NAME field from the STUDENT table and place it into a table with ADVISOR-NUMBER as the primary key.

Figure 9-32 on page 421 shows the 3NF versions of the sample data for STUDENT, ADVISOR, COURSE, and GRADE. The final 3NF design is:

STUDENT (STUDENT-NUMBER, STUDENT NAME, TOTAL-CREDITS, GPA, ADVISOR-NUMBER)

ADVISOR (ADVISOR-NUMBER, ADVISOR-NAME)

COURSE (COURSE-NUMBER, COURSE-DESC, NUM-CREDITS)

GRADE (STUDENT-NUMBER, COURSE-NUMBER, GRADE)

Figure 9-33 on page 422 shows the complete ERD after normalization. Now there are four entities: STUDENT, ADVISOR, COURSE, and GRADE, which is an associative entity. If you go back to Figure 9-28 on page 418, which was drawn before you identified GRADE as an entity, you can see that the M:N relationship between STUDENT and COURSE has been converted into two 1:M relationships: one relationship between STUDENT and GRADE and the other relationship between COURSE and GRADE.

To create 3NF designs, you must understand the nature of first, second, and third normal forms. In your work as a systems analyst, you will encounter designs that are much more complex than the examples in this chapter. You also should know that normal forms beyond 3NF exist, but they rarely are used in business-oriented systems.

CASE IN POINT 9.2: CyberToys

You handle administrative support for CyberToys, a small chain that sells computer hardware and software and specializes in personal service. The company has four stores located at malls and is planning more. Each store has a manager, a technician, and between one and four sales reps.

Bruce and Marcia Berns, the owners, want to create a personnel records database, and they asked you to review a table that Marcia designed. She suggested fields for store number, location, store telephone, manager name, and manager home telephone. She also wants fields for technician name and technician home telephone and fields for up to four sales rep names and sales rep home telephones.

Draw Marcia's suggested design and analyze it using the normalization concepts you learned in the chapter. What do you think of Marcia's design and why? What would you propose?

primary key

STUDENT

STUDENT-NUMBER	STUDENT-NAME	TOTAL-CREDITS	GPA	ADVISOR-NUMBER	ADVISOR-NAME
1035	Linda	47	3.647	49	Smith
3397	Sam	29	3.000	49	Smith
4070	Kelly	14	2.214	23	Jones

primary key

COURSE

COURSE-NUMBER	COURSE-DESC	NUM-CREDITS
BUS105	Introduction to Business	2
CHM112	General Chemistry I	4
CSC151	Computer Science I	4
ENG101	English Composition	3
MKT212	Marketing Management	3

primary key based on combination of two fields

GRADE

STUDENT-NUMBER	COURSE-NUMBER	GRADE
1035	CSC151	B
1035	MKT212	A
1035	ENG101	B
1035	CHM112	A
1035	BUS105	A
3397	ENG101	A
3397	MKT212	C
3397	CSC151	B
4070	CSC151	B
4070	CHM112	C
4070	ENG101	C
4070	BUS105	C

FIGURE 9-31 STUDENT, COURSE, and GRADE tables in 2NF. Notice that all fields are functionally dependent on the entire primary key of their respective tables.

STUDENT

STUDENT-NUMBER	STUDENT-NAME	TOTAL-CREDITS	GPA	ADVISOR-NUMBER
1035	Linda	47	3.647	49
3397	Sam	29	3.000	49
4070	Kelly	14	2.214	23

in 3NF, no nonkey field is dependent on another nonkey field

ADVISOR

ADVISOR-NUMBER	ADVISOR-NAME
23	Jones
49	Smith

COURSE

COURSE-NUMBER	COURSE-DESC	NUM-CREDITS
BUS105	Introduction to Business	2
CHM112	General Chemistry I	4
CSC151	Computer Science I	4
ENG101	English Composition	3
MKT212	Marketing Management	3

GRADE

STUDENT-NUMBER	COURSE-NUMBER	GRADE
1035	CSC151	B
1035	MKT212	A
1035	ENG101	B
1035	CHM112	A
1035	BUS105	A
3397	ENG101	A
3397	MKT212	C
3397	CSC151	B
4070	CSC151	B
4070	CHM112	C
4070	ENG101	C
4070	BUS105	C

FIGURE 9-32 STUDENT, ADVISOR, COURSE, and GRADE tables in 3NF. When the STUDENT table is transformed from 2NF to 3NF, the result is two tables: STUDENT and ADVISOR.

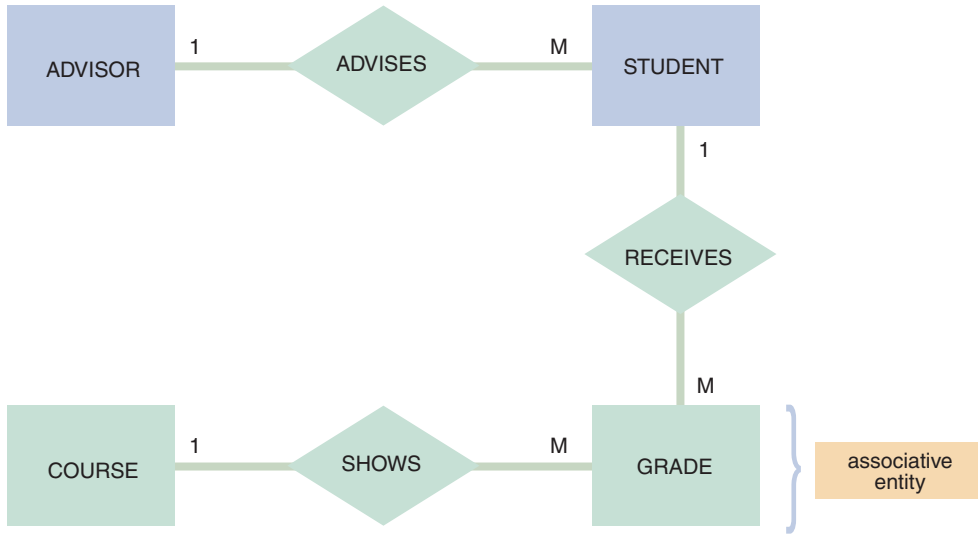


FIGURE 9-33 The entity-relationship diagram for STUDENT, ADVISOR, and COURSE after normalization. The GRADE entity was identified during the normalization process. GRADE is an associative entity that links the STUDENT and COURSE tables.

Using Codes During Data Design

A **code** is a set of letters or numbers that represents a data item. Codes can be used to simplify output, input, and data formats. During the data design process, you review existing codes and develop new ones that will be used to store and access data efficiently.

Overview of Codes

Because codes often are used to represent data, you encounter them constantly in your everyday life. Student numbers, for example, are unique codes to identify students in a school registration system. Three students with the name John Turner might be enrolled at your school, but only one is student number 268960.

Your ZIP code is another common example. A ZIP code contains multiple items of information compressed into nine digits. The first digit identifies one of ten geographical areas of the United States. The combination of the next three digits identifies a major city or major distribution point. The fifth digit identifies an individual post office, an area within a city, or a specific delivery unit. The last four digits identify a post office box or a specific street address.

For example, consider the ZIP code 27906-2624 shown in Figure 9-34. The first digit, 2, indicates a broad geographical area in the eastern United States. The digits, 790, indicate Elizabeth City, North Carolina. The fifth digit, 6, identifies the post office that services the College of the Albemarle. The last four digits, 2624, identify the post office box for the college.

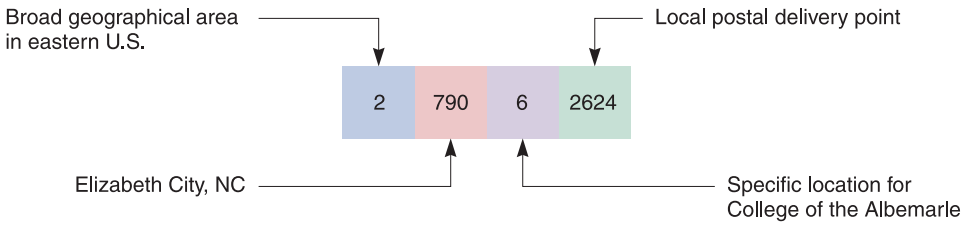


FIGURE 9-34 A ZIP code is an example of a significant digit code that uses subgroups to provide information.

As you can imagine, codes serve many useful purposes. Because codes often are shorter than the data they represent, they save storage space and costs, reduce data transmission time, and decrease data entry time. For example, ZIP codes are used to classify and sort mail efficiently. Codes also can be used to reveal or conceal information. The last two digits of a seven-digit part number, for example, can represent the supplier number. The coded wholesale price on a retail price tag is known to salespeople but generally not to customers.

Finally, codes can reduce data input errors in situations when the coded data is easier to remember and enter than the original source data, when only certain valid codes are allowed, and when something within the code itself can provide immediate verification that the entry is correct.

Types of Codes

Companies use many different coding methods. Because information system users must work with coded data, the codes should be easy to learn and apply. If you plan to create new codes or change existing ones, you first should obtain comments and feedback from users. The following section describes eight common coding methods.

1. **Sequence codes** are numbers or letters assigned in a specific order. Sequence codes contain no additional information other than an indication of order of entry into the system. For example, a human resource system issues consecutive employee numbers to identify employees. Because the codes are assigned in the order in which employees are hired, you can use the code to see that employee number 584 was hired after employee number 433. The code, however, does not indicate the starting date of either person's employment.
2. **Block sequence codes** use blocks of numbers for different classifications. College course numbers usually are assigned using a block sequence code. 100-level courses, such as Chemistry 110 and Mathematics 125, are freshman-level courses, whereas course numbers in the 200s indicate sophomore-level courses. Within a particular block, the sequence of numbers can have some additional meaning, such as when English 151 is the prerequisite for English 152.
3. **Alphabetic codes** use alphabet letters to distinguish one item from another based on a category, an abbreviation, or an easy-to-remember value, called a mnemonic code. Many classification codes fit more than one of the following definitions:
    a. **Category codes** identify a group of related items. For example, a local department store uses a two-character category code to identify the department in which a product is sold: GN for gardening supplies, HW for hardware, and EL for electronics.
    b. **Abbreviation codes** are alphabetic abbreviations. For example, standard state codes include NY for New York, ME for Maine, and MN for Minnesota. Some abbreviation codes are called **mnemonic codes** because they use a specific combination of letters that are easy to remember. Many three-character airport codes such as those pictured in Figure 9-35 on the next page are mnemonic codes: BOS represents Boston, SEA represents Seattle, and ANC represents Anchorage. Some airport codes are not mnemonic, such as ORD, which designates Chicago O'Hare International Airport, or HPN, which identifies the White Plains, New York Airport.

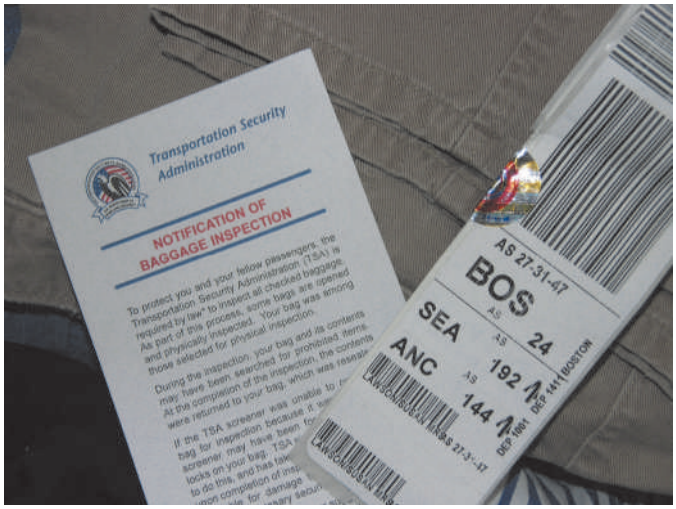


FIGURE 9-35 Airline baggage tags include three-letter codes that identify airports, and machine-readable bar codes that contain information about the passenger, the flight number, and other pertinent information.

4. **Significant digit codes** distinguish items by using a series of subgroups of digits. ZIP codes, for example, are significant digit codes. Other such codes include inventory location codes that consist of a two-digit warehouse code, followed by a one-digit floor number code, a two-digit section code, a one-digit aisle number, and a two-digit bin number code. Figure 9-36 illustrates the inventory location code 11205327. What looks like a large eight-digit number is actually five separate numbers, each of which has significance.
5. **Derivation codes** combine data from different item attributes, or characteristics, to build the code. Most magazine subscription codes are derivation codes. One popular magazine's subscriber code consists of the subscriber's five-digit ZIP code, followed by the first, third, and fourth letters of the subscriber's last name, the last two digits of the subscriber's house number, and the first, third, and fourth letters of the subscriber's street name. The magazine's subscriber code for one particular subscriber is shown in Figure 9-37.

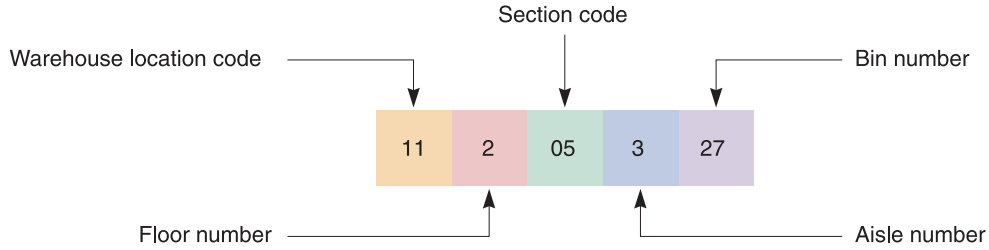


FIGURE 9-36 Sample of a code that uses significant digits to pinpoint the location of an inventory item.

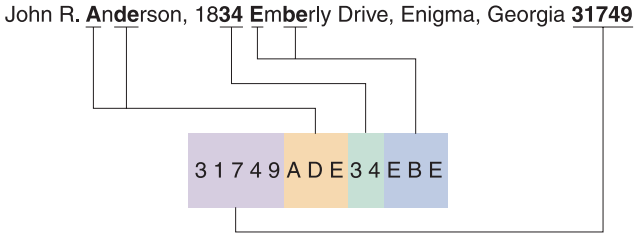


FIGURE 9-37 A magazine subscriber code is derived from various parts of the name and address.

6. **Cipher codes** use a keyword to encode a number. A retail store, for example, might use a 10-letter word, such as CAMPGROUND, to code wholesale prices, where the letter C represents 1, A represents 2, and so on. Thus, the code, GRAND, indicates that the store paid $562.90 for the item.
7. **Action codes** indicate what action is to be taken with an associated item. For example, a student records program might prompt a user to enter or click an action code such as *D* (to display a record), *A* (to add a record), and *X* (to exit the program).

Developing a Code

Devising a code with too many features makes it difficult to remember, decipher, and verify. Keep the following suggestions in mind when developing a code:

1. Keep codes concise. Do not create codes that are longer than necessary. For example, if you need a code to identify each of 250 customers, you will not need a six-digit code.

2. Allow for expansion. A coding scheme must allow for reasonable growth in the number of assigned codes. If the company currently has eight warehouses, you should not use a one-digit code for the warehouse number. If three more warehouses are added, the code must be increased to two digits or changed to a character code in order to identify each location. The rule also applies to using a single letter as a character code; you might need more than 26 codes in the future. Of course, you can add more characters, which is just what the airline industry has done. Airlines now use six-character locator codes, which allow more than 300 million combinations.
3. Keep codes stable. Changes in codes can cause consistency problems and require data updates. During the changeover period, you will have to change all the stored occurrences of a particular code and all documents containing the old code, as users switch to the new code. Usually, both the old and new codes are used for an interim period, and special procedures are required to handle the two codes. For example, when area codes change, you can use either area code for a certain time period.
4. Make codes unique. Codes used for identification purposes must be unique to have meaning. If the code HW can indicate hardware or houseware, the code is not very useful.
5. Use sortable codes. If products with three-digit codes in the 100s or the 300s are of one type, while products with codes in the 200s are a different type, a simple sort will not group all the products of one type together. In addition, be careful that single-digit character codes will sort properly with double-digit codes — in some cases you must add a leading zero (01, 02, 03, and so on) to ensure that codes sort correctly.
6. Avoid confusing codes. Do not code some part numbers with two letters, a hyphen, and one digit, and others with one letter, a hyphen, and two digits. Avoid allowing both letters and numbers to occupy the same positions within a code because some of those are easily confused. It is easy to confuse the number zero (0) and the uppercase letter O, or the number one (1) with the lowercase letter L (l) or uppercase letter I. For example, the five-character code 5Z081 easily can be misread as 5ZO8I, or 52081, or even totally incorrectly as S2OBI.
7. Make codes meaningful. Codes must be easy to remember, useful for users, convenient to use, and easy to encode and interpret. Using SW as a code for the southwest sales region, for example, has far more meaning than the code 14. Using ENG as the code for the English department is easier to interpret and remember than either XVA or 132.
8. Use a code for a single purpose. Do not use a single code to classify two or more unrelated attributes. For example, if you use a single code to identify the combination of an employee's department *and* the employee's insurance plan type, users will have difficulty identifying all the subscribers of a particular plan, or all the workers in a particular department, or both. A separate code for each separate characteristic makes much more sense.
9. Keep codes consistent. For example, if the payroll system already is using two-digit codes for departments, do not create a new, different coding scheme for the personnel system. If the two systems already are using different coding schemes, you should try to convince the users to adopt a consistent coding scheme.

CASE IN POINT 9.3: DOTCOM TOOLS

DotCom Tools operates a small business that specializes in hard-to-find woodworking tools. The firm advertises in various woodworking magazines, and currently accepts mail and telephone orders. DotCom is planning a Web site that will be the firm's primary sales channel. The site will feature an online catalog, powerful search capabilities, and links to woodworking information and resources.

DotCom has asked you, an IT consultant, whether a set of codes would be advantageous and if so, what codes you would suggest. Provide at least two choices for a customer code and at least two choices for a product code. Be sure to describe your choices and provide some specific examples. Also include an explanation of why you selected these particular codes and what advantages they might offer.

DATABASE DESIGN: ONE STEP AT A TIME

After normalizing your table designs, you are ready to create a physical database. The following steps will help you construct a database that is user-friendly, efficient, maintainable, and meets IT industry design standards:

Step 1: Create an initial ERD. Start by reviewing your DFDs and class diagrams to identify all system entities. Also consider any data stores shown on the DFDs to see whether they might represent entities. For example, a data store called JOB APPLICANTS will need a table to store the information and attributes of each applicant.

Step 2: Next, create an ERD. Carefully analyze each relationship to determine if it is 1:1, 1:M, or M:N.

Step 3: During modeling, you created a data dictionary. Now, review all the data elements to be sure that each one is assigned to an appropriate entity.

Step 4: Review the 3NF designs for all tables, and verify that all primary, secondary, and foreign keys are identified properly. If you added any associative entities during the design process, now is the time to update your ERD.

Step 5: Double-check all data dictionary entries. Make sure that the entries for data stores, records, and data elements are documented completely and correctly. Also be sure that all codes that were developed or identified during the data design process are documented in the data dictionary.

After creating your final ERD and normalized table designs, you can transform them into a database. You might work with applications such as Microsoft Access, Oracle, IBM's DB2, or other DBMS development tools — or the design might be coded in-house, using one of the popular programming languages.

In the next section, you will work with examples of relational design and apply skills that you learned earlier in the chapter. At this point, it makes no difference whether you used traditional modeling or object-oriented methods to create your model. Either way, you must transform the logical model into a physical data structure that will support user needs and business requirements.

Database Models

Whether traditional or object-oriented development methods are used, the relational database model is universally accepted as the best way to organize, store, and access large amounts of data. Relational databases can run on many platforms, including personal computers, and are well suited to client/server computing because they are so powerful and flexible. Other models exist, but they are less common, and generally are found on older, mainframe-based systems. At the other end of the spectrum, researchers are exploring **neural architectures** that resemble human brain functions, and some observers believe that this research could change the way we think about database design.

A Real-World Business Example

Imagine a company that provides on-site service for electronic equipment, including parts and labor. Figure 9-38 shows the overall database design that such a firm might use. If you study the figure carefully, you will see examples of many concepts described earlier. The database consists of seven separate tables, all joined by common fields so they form an integral data structure.

Figure 9-39 on the next page shows even more detail, including sample data, primary keys, and common fields. Notice that the entities include customers, technicians, service calls, and parts. Other tables store data about labor and parts that are used on specific service calls. Also notice that all tables use a single field as a primary key, except the SERVICE-LABOR-DETAIL and SERVICE-PARTS-DETAIL tables, where the primary key requires a combination of two fields to uniquely identify each record.

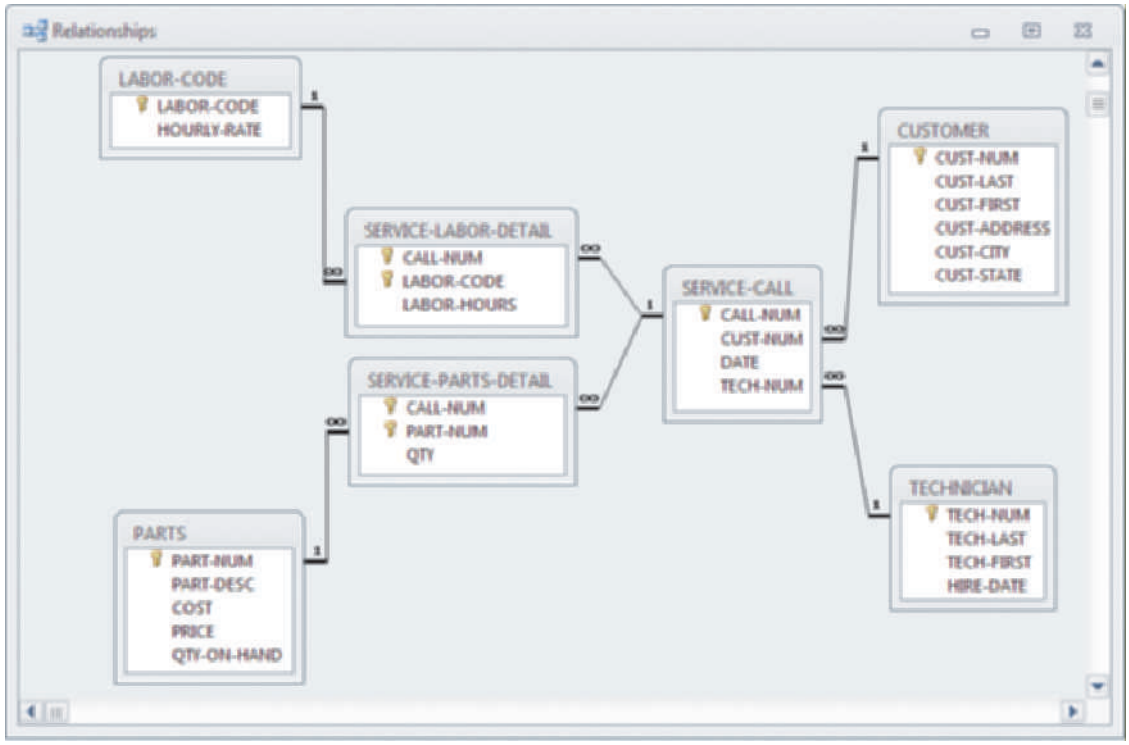


FIGURE 9-38 A relational database design for a computer service company uses common fields to link the tables and form an overall data structure. Notice the one-to-many notation symbols, and the primary keys, which are shown in bold.

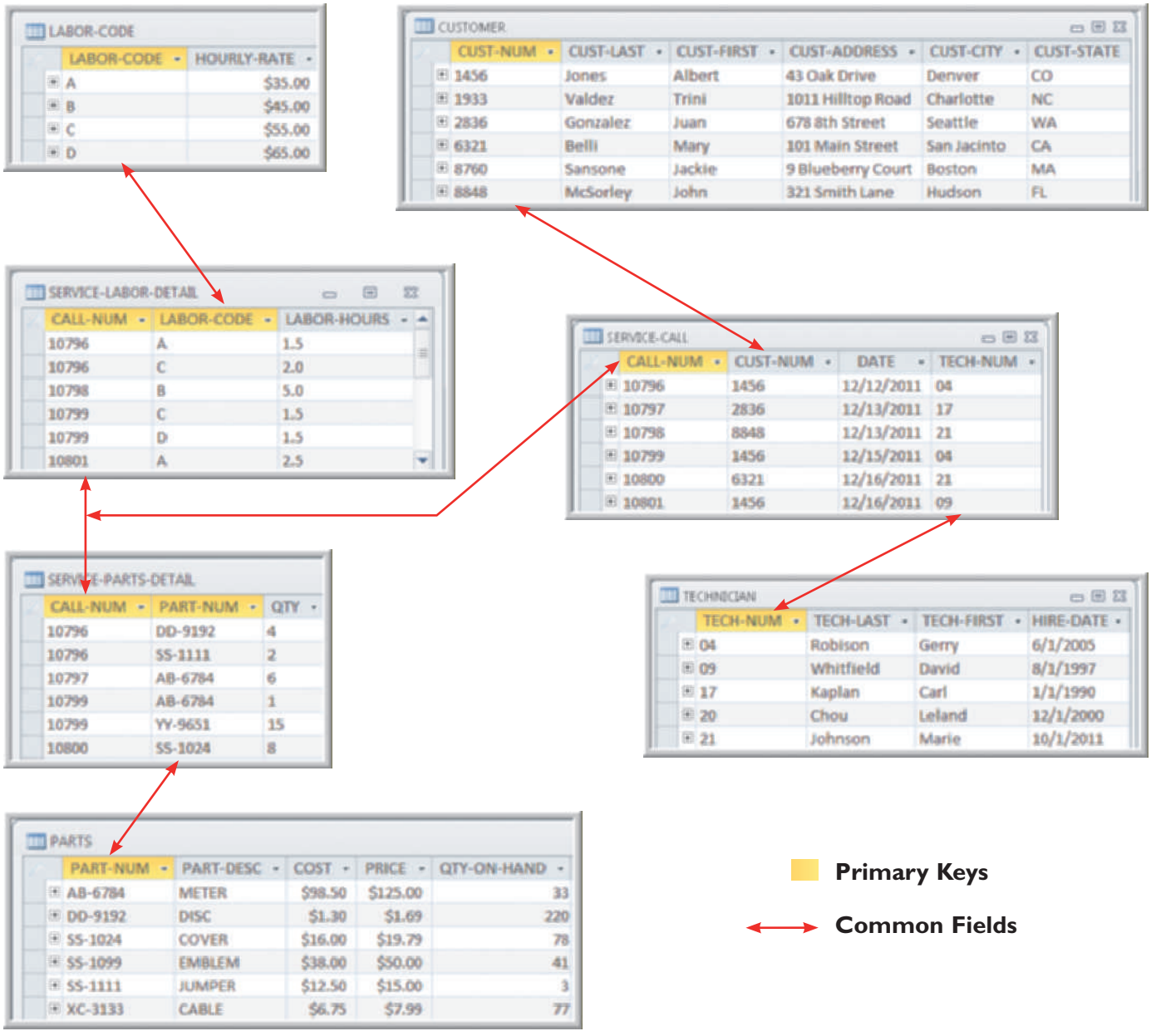


FIGURE 9-39 Sample data, primary keys, and common fields for the database shown in Figure 9-38 on the previous page. The design is in 3NF. Notice that all nonkey fields are functionally dependent on a primary key, the whole key, and nothing but the key.

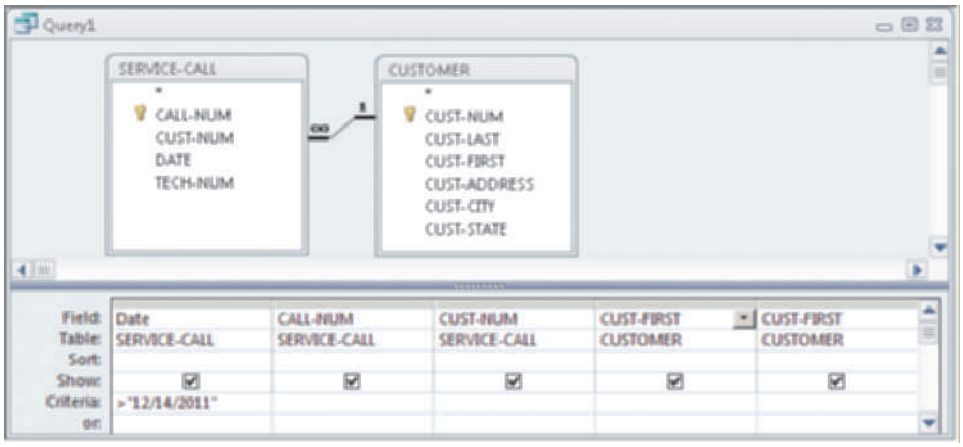
Working with a Relational Database

To understand the power and flexibility of a relational database, try the following exercise. Suppose you work in IT, and the sales team needs answers to three specific questions. The data might be stored physically in seven tables. You know how to create Microsoft Access queries, but to learn more about database logic and design, you decide to study Figure 9-39, and then pretend to tell the database how to find the answers, step by step. The questions are shown in Figures 9-40, 9-41, and 9-42 on pages 429 and 430.

Question 1: Did any customers receive service after 12/14/2011? If so, who were they?

Steps: To identify service dates after 12/14/2011, go to the SERVICE-CALL table and look in the DATE field for dates later than ***12/14/2011***. Three service call records meet that condition: ***10799***, ***10800***, and ***10801***. Each record has a CUST-NUM field, which shows that two of the calls were made to customer ***1456***, and one call was made to customer ***6321***. To obtain the customer names, go to the CUSTOMER table, look up the customer numbers, and identify the two customers as Albert Jones and Mary Belli.

Access Query:



Query Results:

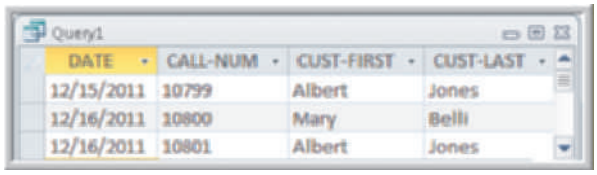
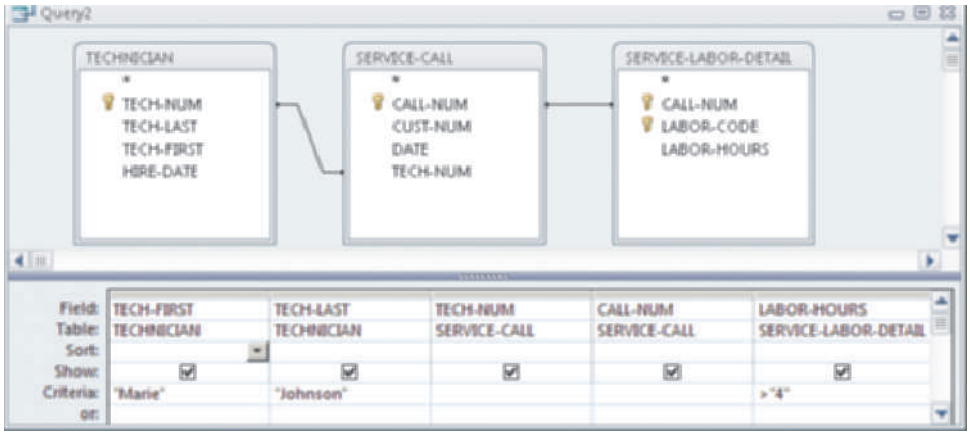


FIGURE 9-40 Question 1

Question 2: Did technician Marie Johnson put in more than six hours of labor on any service calls? If so, which ones?

Steps: To locate Marie Johnson, find her name in the TECHNICIAN table and locate her technician number, which is ***21***. Then, go to the SERVICE-CALL table and seek matching values of ***21*** in the TECH-NUM field. Marie's two service calls are shown as call numbers ***10798*** and ***10800***. Finally, go to the SERVICE-LABOR-DETAIL table and use those two values to see if either service call showed more than six hours of labor. In the example, only service call ***10798*** meets the conditions.

Access Query:



Query Results:

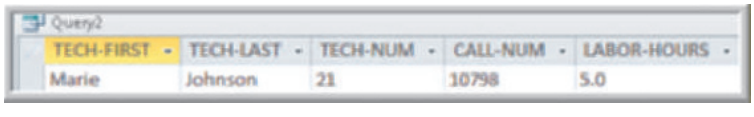
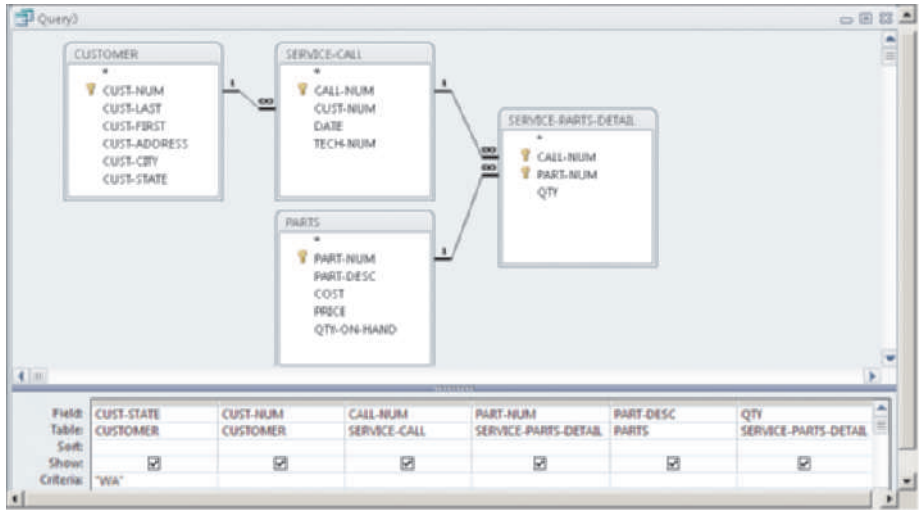


FIGURE 9-41 Question 2

Question 3: Were any parts used on service calls in Washington? If so, what were the part numbers, descriptions and quantities?

Steps: To identify Washington customers, go to the CUSTOMER table and locate all records with the value of ***WA*** in the CUST-STATE field. This shows a record for Juan Gonzalez, customer number ***2836***. Next, to see whether he received service, look for the value ***2836*** in the SERVICE-CALL table. This produces one record: service call ***10797***. Then, to find out what, if any, parts were used, go to the SERVICE-PARTS-DETAIL table, and look for service record ***10797***. It appears part ***AB-6784*** was used on service call ***10797***. Finally, to get a description for part ***AB-6784***, go to the PARTS table, look for part number ***AB-6784***, and retrieve the description. The final result indicates that six meters with part number ***AB-6784*** were sold to Washington customers.

Access Query:



Query Results:

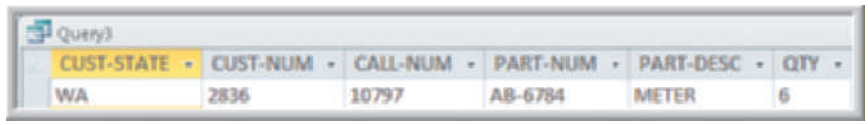


FIGURE 9-42 Question 3

Data Storage and Access

Data storage and access involve strategic business tools, such as data warehousing and data mining software, as well as logical and physical storage issues, selection of data storage formats, and special considerations regarding storage of date fields.

Strategic Tools for Data Storage and Access

ON THE WEB

To learn more about data warehousing, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Data Warehousing link.

Companies use data warehousing and data mining as strategic tools to help manage the huge quantities of data they need for business operations and decisions. A large number of software vendors compete for business in this fast-growing IT sector.

DATA WAREHOUSING Large firms maintain many databases, which might or might not be linked together into an overall structure. To provide rapid access to this information, companies use software packages that organize and store data in special configurations called data warehouses. A **data warehouse** is an integrated collection of data that can include seemingly unrelated information, no matter where it is stored in the company. Because it can link various information systems and databases, a data warehouse provides an enterprise-wide view to support management analysis and decision making.

A data warehouse allows users to specify certain **dimensions**, or characteristics. By selecting values for each characteristic, a user can obtain multidimensional information from the stored data. For example, in a typical company, most data is generated by transaction-based systems, such as order processing systems, inventory systems, and payroll systems. If a user wants to identify the customer on sales order 34071, he or she can retrieve the data easily from the order processing system by entering an order number.

On the other hand, suppose that a user wants to see November, 2011 sales results for Sally Brown, the sales rep assigned to Jo-Mar Industries. The data is stored in two different systems: the sales information system and the human resources information system, as shown in Figure 9-43. Without a data warehouse, it would be difficult for a user to extract data that spans several information systems and time frames. Rather than accessing separate systems, a data warehouse stores transaction data in a format that allows users to retrieve and analyze the data easily.

While a data warehouse typically spans the entire enterprise, many firms prefer to use a **data mart**, which is designed to serve the needs of a specific department, such as sales, marketing, or finance. Each data mart includes only the data that users in that department require to perform their jobs. There are pros and cons to both approaches, and the best solution usually depends on the specific situation.

As the article in Figure 9-44 on the next page points out, storing large quantities of data is like building a house — it doesn't just happen. A well-constructed data warehouse needs an architecture that includes detailed planning and specifications.

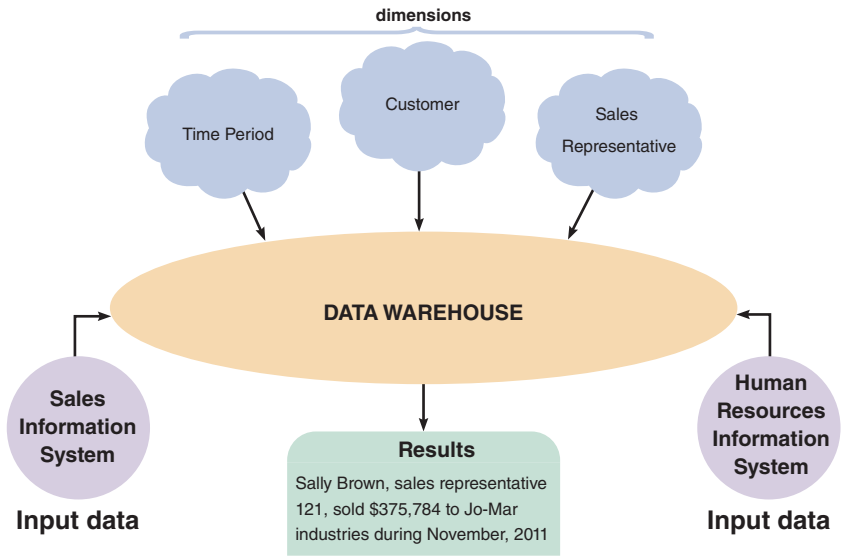


FIGURE 9-43 A data warehouse stores data from several systems. By selecting data dimensions, a user can retrieve specific information without having to know how or where the data is stored.

DATA MINING **Data mining** software looks for meaningful data patterns and relationships. For example, data mining software could help a consumer products firm identify potential customers based on their prior purchases. Information about customer behavior is valuable, but data mining also raises serious ethical and privacy issues, such as the example in the Question of Ethics feature on page 436. Figure 9-45 on the next page shows North Carolina State University's Ethics in Computing site, which offers many articles and research papers on this subject.

The enormous growth in e-commerce has focused attention on data mining as a marketing tool. In an article in *New Architect*, a Web-based magazine, Dan R. Greening noted that Web hosts typically possess a lot of information about visitors, but most of it is of little value. His article mentions that smart marketers and business analysts are using data mining techniques, which he describes as "machine learning algorithms that find buried patterns in databases, and report or act on those findings." He concludes by saying that "The great advantage of Web marketing is that you can measure visitor interactions more effectively than in brick-and-mortar stores or direct mail. Data mining works best when you have clear, measurable goals." Some of the goals he suggests are:

- Increase the number of pages viewed per session
- Increase the number of referred customers

ON THE WEB

To learn more about data mining, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Data Mining link.

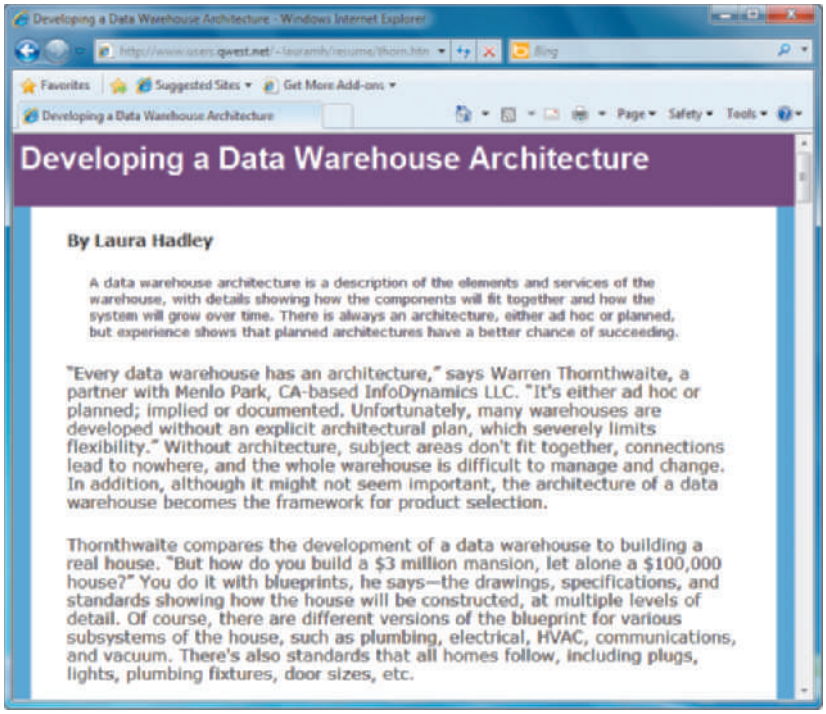


FIGURE 9-44 As the author points out, without an overall architecture, a data warehouse would be difficult to manage and change.

- Reduce **clicks to close**, which means average page views to accomplish a purchase or obtain desired information
- Increase checkouts per visit
- Increase average profit per checkout

This type of data gathering is sometimes called **clickstream storage**. Armed with this information, a skillful Web designer could build a profile of typical new customers, returning customers, and customers who browse but do not buy. Although this information would be very valuable to the retailer, clickstream storage could raise serious legal and privacy issues if an unscrupulous firm sought to link a customer's Web behavior to a specific name or e-mail address, and then sell or otherwise misuse the information.

Because it can detect patterns and trends in large amounts of data, data mining is a valuable tool for managers. The popular Wikipedia site contains an interesting example of data mining. According to the Wikipedia article, a hypothetical chain of supermarkets performed a detailed analysis of purchases, and found that beer and diapers were often purchased together. Without attempting to explain this correlation, the obvious tactic for a retailer would be to display these items in the same area of the store. Wikipedia states that this data mining technique is often called **market basket analysis**.

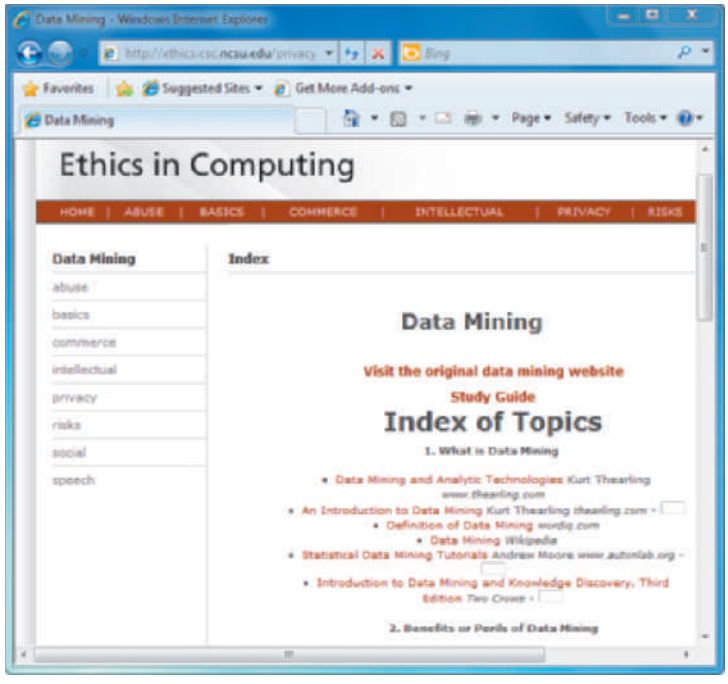


FIGURE 9-45 The Ethics in Computing site at North Carolina State University offers many articles about ethical and privacy issues associated with data mining.

Logical and Physical Storage

It is important to understand the difference between logical storage and physical storage. **Logical storage** refers to data that a user can view, understand, and access, regardless of how or where that information actually is organized or stored. In contrast, **physical storage** is strictly hardware-related, because it involves the process of reading and writing binary data to physical media such as a hard drive, CD-ROM, or network-based storage device. For example, portions of a document might be stored in different physical locations on a hard drive, but the user sees the document as a single logical entity on the computer screen.

LOGICAL STORAGE Logical storage consists of alphabetic and numeric **characters**, such as the letter *A* or the number *9*. As you learned earlier in this chapter, a set of related characters forms a field, which describes a single characteristic, or attribute, of a person, place, thing, or event. A field also is called a **data element** or a **data item**.

When designing fields, you should provide space for the largest values that can be anticipated, without allocating unnecessarily large storage capacities that will not be used. For example, suppose you are designing a customer order entry system for a firm with 800 customers. It would be a mistake to limit the customer number field to three, or even four characters. Instead, you might consider a five-character field with leading zeros that could store customer numbers from *00001* to *99999*.

You also might consider a mix of alphabetic and numeric characters, which many people find easier to view and use. Alphabetic characters expand the storage capacity, because there are 26 possible values for each character position. Most airlines now use six alphabetic characters as a record locator, which has over 300 million possible values.

A **logical record** is a set of field values that describes a single person, place, thing, or event. For example, a logical customer record contains specific field values for a single customer, including the customer number, name, address, telephone number, credit limit, and so on. Application programs see a logical record as a group of related fields, regardless of how or where the data is stored physically.

The term *record* usually refers to a logical record. Whenever an application program issues a read or write command, the operating system supplies one logical record to the program or accepts one logical record from the program. The physical data might be stored on one or more servers, in the same building or thousands of miles away, but all the application program sees is the logical record — the physical storage location is irrelevant.

PHYSICAL STORAGE Physical storage involves a **physical record**, or **block**, which is the smallest data unit that can be handled by the operating system. The system reads or writes one physical record at a time. When the system *reads* a physical record, it loads the data from storage into a **buffer**, which is a segment of computer memory.

The physical storage location can be local, remote, or Web-based. Similarly, when the system *writes* a physical record, all data in the memory buffer is saved physically to a storage location. A physical record can contain more than one logical record, depending on the **blocking factor**. For example, a blocking factor of two means that each physical record will consist of two logical records. Some database programs, such as Microsoft Access, automatically write a physical record each time that a logical record is created or updated.

Data Coding and Storage

Computers represent data as **bits**, or **binary digits**, that have only two possible values: 1 (which indicates an electrical signal) and 0 (which indicates the absence of a signal). A computer understands a group of bits as a digital code that can be transmitted, received, and stored. Computers use various data coding and storage schemes, such as EBCDIC, ASCII, and binary. A more recent coding standard called Unicode also is popular. Also, the storage of dates raises some design issues that must be considered.

EBCDIC, ASCII, AND BINARY EBCDIC (pronounced EB-see-dik), which stands for Extended Binary Coded Decimal Interchange Code, is a coding method used on mainframe computers and high-capacity servers. **ASCII** (pronounced ASK-ee), which stands for American Standard Code for Information Interchange, is a coding method used on most personal computers. EBCDIC and ASCII both require eight bits, or one **byte**, for each character. For example, the name Ann requires three bytes of storage, the number *12,345* requires five bytes of storage, and the number *1,234,567,890* requires ten bytes of storage.

Compared with character-based formats, a **binary storage format** offers a more efficient storage method because it represents numbers as actual binary values, rather than as coded numeric digits. For example, an **integer format** uses only 16 bits, or two bytes, to represent the number *12,345* in binary form. A **long integer format** uses 32 bits, or four bytes, to represent the number *1,234,567,890* in binary form.

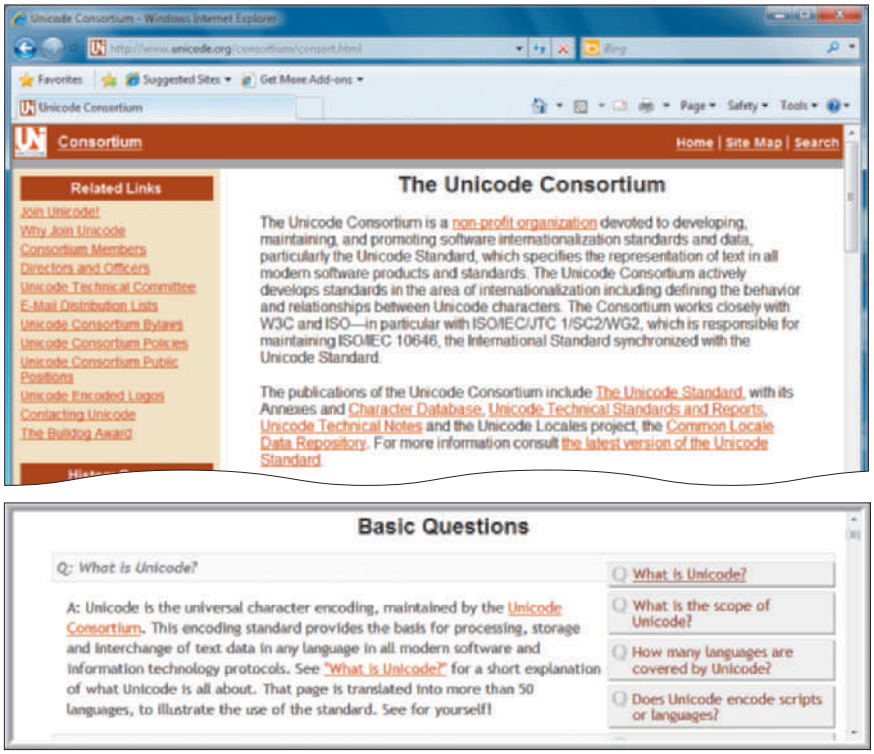


FIGURE 9-46 Unicode is an international coding format that represents characters as integers, using 16 bits per character. The Unicode Consortium maintains standards and support for Unicode.

UNICODE Unicode is a more recent coding standard that uses two bytes per character, rather than one. This expanded scheme enables Unicode to represent more than 65,000 unique, multilingual characters. Why is this important? Consider the challenge of running a multinational information system, or developing a program that will be sold in Asia, Europe, and North America. Because it supports virtually all languages, Unicode has become a global standard.

Traditionally, domestic software firms developed a product in English, then translated the program into one or more languages. This process was expensive, slow, and error-prone. In contrast, Unicode creates translatable content right from the start. Today, most popular operating systems support Unicode, and the Unicode Consortium maintains standards and support, as shown in Figure 9-46. Although Unicode has many advantages, it also has some disadvantages. In fact, because of its size and certain typographical issues, Microsoft recommends that you use its Arial Unicode font " … only when you can't use multiple fonts tuned for different writing systems."

STORING DATES What is the best way to store dates? The answer depends on how the dates will be displayed and whether they will be used in calculations.

At the beginning of the 21st century, many firms that used only two digits to represent the year were faced with a major problem called the **Y2K issue**. Based on that experience, most date formats now are based on the model established by the **International Organization for Standardization (ISO)**, which requires a format of four digits for the year, two for the month, and two for the day (YYYYMMDD). A date stored in that format can be sorted easily and used in comparisons. If a date in ISO form is larger than another date in the same form, then the first date is later. For example, 20120927 (September 27, 2012) is later than 20110713 (July 13, 2011).

But, what if dates must be used in calculations? For example, if a manufacturing order placed on June 23 takes three weeks to complete, when will the order be ready? If a payment due on August 13 is not paid until April 27 of the following year, exactly how late is the payment and how much interest is owed? In these situations, it is easier to use absolute dates.

An **absolute date** is the total number of days from some specific base date. To calculate the number of days between two absolute dates, you subtract one date from the other. For example, if the base date is January 1, 1900, then September 27, 2012 has an absolute date value of 41179. Similarly, July 13, 2011 has an absolute date of 40737. If you subtract the earlier date value from the later one, the result is 442 days. You can use a spreadsheet to determine and display absolute dates easily, as shown in Figure 9-47.

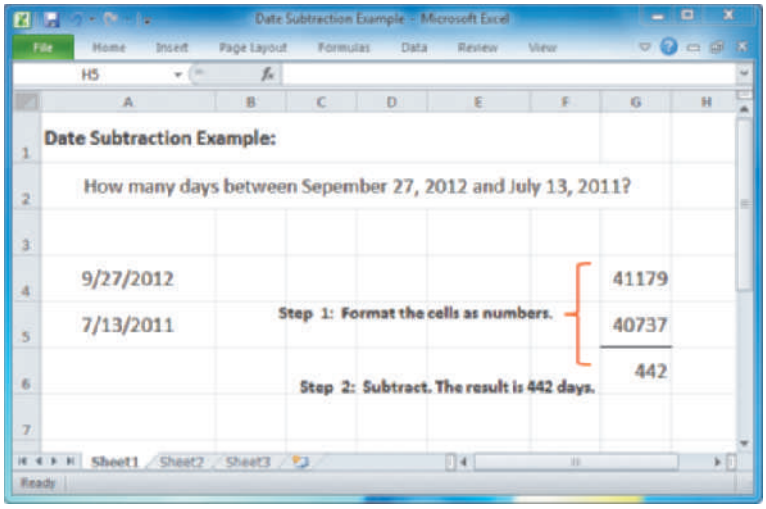


FIGURE 9-47 Microsoft Excel uses absolute dates in calculations. For example, January 1, 1900, has a numeric value of 1. In the example shown, September 27, 2012 is displayed as 41179, and July 13, 2011 is displayed as 40737. The difference between the dates is 442 days.

Data Control

Just as it is important to secure the physical part of the system, as shown in Figure 9-48, file and database control must include all measures necessary to ensure that data storage is correct, complete, and secure. File and database control also is related to input and output techniques discussed earlier.

A well-designed DBMS must provide built-in control and security features, including subschemas, passwords, encryption, audit trail files, and backup and recovery procedures to maintain data. Your main responsibility is to ensure that the DBMS features are used properly.

Earlier in this chapter, you learned that a subschema can be used to provide a limited view of the database to a specific user, or level of users. Limiting access to files and databases is the most common way of protecting stored data. Users must furnish a proper **user ID** and **password** to access a file or database. Different privileges, also called **permissions,** can be associated with different users, so some employees can be limited to read-only access, while other users might be allowed to update or delete data. For highly sensitive data, additional access codes can be established that restrict specific records or fields within records. Stored data also can be encrypted to prevent unauthorized access. **Encryption** is the process of converting readable data into unreadable characters to prevent unauthorized access to the data.

FIGURE 9-48 In addition to network monitoring, system security includes access codes, data encryption, passwords, and audit trails.

All system files and databases must be backed up regularly and a series of **backup** copies must be retained for a specified period of time. In the event of a file catastrophe, **recovery procedures** can be used to restore the file or database to its current state at the time of the last backup. **Audit log files**, which record details of all accesses and changes to the file or database, can be used to recover changes made since the last backup. You also can include **audit fields**, which are special fields within data records to provide additional control or security information. Typical audit fields include the date the record was created or modified, the name of the user who performed the action, and the number of times the record has been accessed.

CASE IN POINT 9.4: SoccerMom

SoccerMom Company sells a patented seat that spectators can take to youth soccer games. The seat folds so it is small enough to fit in the glove box of most vehicles. The company operates a factory in Kansas and also contracts its manufacturing projects to small firms in Canada and Mexico.

An unusual problem has occurred for this small multinational company: People are getting confused about dates in internal memos, purchase orders, and e-mail. Towson Hopkins handles all IT functions for SoccerMom. When he designed the company's database, he was not aware that the format for dates in Canada and Mexico was different from the format used in the United States. For example, in Canada and Mexico, the notation 7/1/11 indicates January 7, 2011, whereas in the United States the same notation indicates July 1, 2011. Although it seems like a small point, the date confusion has resulted in several order cancellations.

Towson has asked for your advice. You could suggest writing a simple program to convert the dates automatically or designing a switchboard command that would allow users to select a date format as data is entered. You realize, however, that SoccerMom might want to do business in other countries in the future. What would be the best course of action? Should SoccerMom adapt to the standard of each country, or should it maintain a single international format? What are the arguments for each option?

A QUESTION OF ETHICS

Olivia is the database manager at Tip Top Toys, a relatively small division of Worldwide Enterprises. Worldwide has nine other divisions, which include insurance, health care products, and financial planning services, to name a few.

Riccardo, corporate marketing director for Worldwide, has requested Tip Top's customer shopping data to target people who might be likely to purchase items or services from other Worldwide divisions. Olivia is not totally comfortable with this, and pointed out Tip Top's Web privacy policy, which states that "Tip Top Toys, a division of Worldwide Enterprises, will not share personal data with other companies without a customer's consent."

Riccardo replied that the statement only applies to outside companies – not other Worldwide divisions. He said he checked with the corporate legal department, and they agreed. Emily responded "Even if it is legally OK, it's not the *right* thing to do. Many people take our statement to mean that their data does not leave Tip Top. At the very least, we should give customers a choice, and share the data only with their consent."

Do you agree with Olivia? Why or why not?

Chapter Summary

In this chapter, you continued your study of the systems design phase of the SDLC. You learned that files and tables contain data about people, places, things, or events that affect the information system. File-oriented systems, also called file processing systems, manage data stored in separate files, including master files, table files, transaction files, work files, security files, and history files.

A database consists of linked tables that form an overall data structure. A database management system (DBMS) is a collection of tools, features, and interfaces that enable users to add, update, manage, access, and analyze data in a database.

DBMS designs are more powerful and flexible than traditional file-oriented systems. A database environment offers scalability, support for organization-wide access, economy of scale, data sharing among user groups, balancing of conflicting user requirements, enforcement of standards, controlled redundancy, effective security, flexibility, better programmer productivity, and data independence. Large-scale databases are complex and require extensive security and backup/recovery features.

DBMS components include interfaces for users, database administrators, and related systems; a data manipulation language; a schema; and a physical data repository. Other data management techniques include data warehousing, which stores data in an easily accessible form for user access, and data mining, which looks for meaningful patterns and relationships among data. Data mining also includes clickstream storage, which records how users interact with a site, and market basket analysis, which can identify product relationships and consumer buying patterns.

In an information system, an entity is a person, place, thing, or event for which data is collected and maintained. A field, or attribute, is a single characteristic of an entity. A record, or tuple, is a set of related fields that describes one instance of an entity. Records are grouped into files (in a file-oriented system) and tables (in a database environment).

A primary key is the field or field combination that uniquely and minimally identifies a specific record; a candidate key is any field that could serve as a primary key. A foreign key is a field or field combination that must match the primary key of another file or table. A secondary key is a field or field combination used as the basis for sorting or retrieving records.

An entity-relationship diagram (ERD) is a graphic representation of all system entities and the relationships among them. The ERD is based on entities and data stores in DFDs prepared during the systems analysis phase. The three basic relationships represented in an ERD are one-to-one (1:1), one-to-many (1:M), and many-to-many (M:N). In a M:N relationship, the two entities are linked by an associative entity.

The relationship between two entities also is referred to as cardinality. A common form of cardinality notation is called crow's foot notation, which uses various symbols to describe the characteristics of the relationship.

Normalization is a process for avoiding problems in data design. A first normal form (1NF) record has no repeating groups. A record is in second normal form (2NF) if it is in 1NF and all nonkey fields depend on the entire primary key. A record is in third normal form (3NF) if it is in 2NF and if no field depends on a nonkey field.

Data design tasks include creating an initial ERD; assigning data elements to an entity; normalizing all table designs; and completing the data dictionary entries for files, records, and data elements. Files and database tables should be sized to estimate the amount of storage space they will require.

You learned that a code is a set of letters or numbers used to represent data in a system. By using codes, you can speed up data entry, reduce data storage space, and reduce transmission time. Codes also can be used to reveal or to conceal information. The main types of codes are sequence codes, block sequence codes, classification codes, alphabetic codes (including category codes, abbreviation codes, and mnemonic codes), significant digit codes, derivation codes, cipher codes, and action codes.

Logical storage is information seen through a user's eyes, regardless of how or where that information actually is organized or stored. Physical storage is hardware-related and involves reading and writing blocks of binary data to physical media. A logical record is a related set of field values that describes a single person, place, thing, or event. A physical record consists of one or more logical records, depending on the blocking factor. Data storage formats include EBCDIC, ASCII, binary, and Unicode. Dates can be stored in several formats, including ISO and absolute format.

File and database control measures include limiting access to the data, data encryption, backup/recovery procedures, audit-trail files, and internal audit fields.

Key Terms and Phrases

1:1 *406*
1:M *406*
abbreviation codes *423*
absolute date *435*
action codes *424*
alphabetic codes *423*
ASCII *433*
associative entity *407*
attribute *402*
audit fields *436*
audit log files *436*
backup *436*
binary digit *433*
binary storage format *434*
bit *433*
block *433*
block sequence codes *423*
blocking factor *433*
buffer *433*
byte *433*
candidate key *402*
cardinality *408*
cardinality notation *408*
category codes *423*
characters *432*
cipher codes *424*
clicks to close *432*
clickstream storage *432*
client/server *395*
clients *400*
code *422*
combination key *402*
common field *402*
composite key *402*
concatenated key *402*
crow's foot notation *408*
data element *432*
data integrity *394*
data item *432*
data manipulation language (DML) *398*
data mart *431*
data mining *431*
data redundancy *394*
data structure *392*
data warehouse *430*
database administrator (DBA) *395*
database management system (DBMS) *392*, *395*
derivation codes *424*
dimensions *431*
EBCDIC *433*
economy of scale *395*
encryption *435*
entity *401*
entity-relationship diagram (ERD) *406*
extranet *400*
field *402*
file *392*
file-oriented system *392*
first normal form (1NF) *412*
foreign key *404*
functionally dependent *413*
history file *394*
HTML (Hypertext Markup Language) *399*
integer format *434*
International Organization for Standardization (ISO) *434*
intranet *399*
JDBC (Java database connectivity) *398*
key fields *402*
logical record *433*
logical storage *432*
long integer format *434*
M:N *407*
many-to-many relationship *407*
market basket analysis *432*
master file *394*
middleware *400*
mnemonic codes *423*
multivalued key *403*
neural architectures *427*
nonkey field *402*
normalization *411*
ODBC (open database connectivity) *398*
one-to-many relationship *406*
one-to-one relationship *406*
orphan *404*
password *435*
permissions *435*
physical record *433*
physical storage *432*
primary key *402*
protocols *400*
query by example (QBE) *396*
query language *396*
record *402*
recovery procedures *436*
referential integrity *404*
relational database *392*
relational model *392*
repeating group *411*
scalability *395*
schema *398*
second normal form (2NF) *413*
secondary key *404*
security file *394*
sequence codes *422*
servers *400*
significant digit codes *424*
SQL (Structured Query Language) *397*
standard notation format *411*
subschema *398*
table *392*
table design *411*
table file *394*
tags *399*
third normal form (3NF) *416*
transaction file *394*
tuple *402*
Unicode *434*
Unified Modeling Language (UML) *408*
unnormalized *412*
user ID *435*
Web browser *399*
Web page *399*
Web server *399*
Web site *399*
Web-centric *400*
work file *394*
Y2K issue *434*

Learn It Online

Instructions: To complete the Learn It Online exercises, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the resources for this chapter, and click the link for the exercise you want to complete.

1 Chapter Reinforcement

TF, MC, and SA

Click one of the Chapter Reinforcement links for Multiple Choice, True/False, or Short Answer. Answer each question and submit to your instructor.

2 Flash Cards

Click the Flash Cards link and read the instructions. Type 20 (or a number specified by your instructor) in the Number of playing cards text box, type your name in the Enter your Name text box, and then click the Flip Card button. When the flash card is displayed, read the question and then click the ANSWER box arrow to select an answer. Flip through the Flash Cards. If your score is 15 (75%) correct or greater, click Print on the File menu to print your results. If your score is less than 15 (75%) correct, then redo this exercise by clicking the Replay button.

3 Practice Test

Click the Practice Test link. Answer each question, enter your first and last name at the bottom of the page, and then click the Grade Test button. When the graded practice test is displayed on your screen, click Print on the File menu to print a hard copy. Continue to take practice tests until you score 80% or better.

4 Who Wants To Be a Computer Genius?

Click the Computer Genius link. Read the instructions, enter your first and last name at the bottom of the page, and then click the Play button. When your score is displayed, click the PRINT RESULTS link to print a hard copy.

5 Wheel of Terms

Click the Wheel of Terms link. Read the instructions, and then enter your first and last name and your school name. Click the PLAY button. When your score is displayed on the screen, right-click the score and then click Print on the shortcut menu to print a hard copy.

6 Crossword Puzzle Challenge

Click the Crossword Puzzle Challenge link. Read the instructions, and then click the Continue button. Work the crossword puzzle. When you are finished, click the Submit button. When the crossword puzzle is redisplayed, submit it to your instructor.

SCR Associates Case Simulation Session 9: Data Design

Overview

The SCR Associates case study is a Web-based simulation that allows you to practice your skills in a real-world environment. The case study transports you to SCR's intranet, where you complete 12 work sessions, each aligning with a chapter. As you work on the case, you will receive e-mail and voice mail messages, obtain information from SCR's online libraries, and perform various tasks.

How do I use the case?

- Review the SCR background material in Chapter 1.
- Read the Preview for this session and study the Task List.
- Visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the **SCR Case Simulation**, and locate the intranet link.
- Enter your name and the password **sad9e**. An opening screen will display the 12 sessions.
- Select this session. Check your e-mail and voice mail carefully, and then work on the tasks.

Preview: Session 9

Your supervisor, Jesse Baker, has asked you to begin working on data design tasks for the new information system, which will be implemented as a relational database. You will need to identify the entities, draw an ERD, design tables, and add sample data to each table.

Task List

1. *List all the entities that interact with the TIMS system. Start by reviewing the data library, previous e-mail messages, DFDs, and other documentation.*
2. *Draw an ERD that shows cardinality relationships among the entities. Send the diagram to Jesse.*
3. *For each entity, Jesse wants to see table designs in 3NF. Use standard notation format to show the primary key and the other fields in each table.*
4. *Jesse wants to use sample data to populate fields for at least three records in each table. Better get started on this right away.*

FIGURE 9-49 Task list: Session 9.

Chapter Exercises

Review Questions

1. Explain the main differences between a file processing system and a database system.
2. What is a DBMS? Briefly describe the components of a DBMS.
3. Describe a primary key, candidate key, secondary key, foreign key, and common field.
4. What are entity-relationship diagrams and how are they used? What symbol is used to represent an entity in an ERD? What symbol is used for a relationship? What is cardinality, and what symbols do you use in the crow's foot notation method?
5. What are data warehousing and data mining? Are the terms related?
6. What is the criterion for a table design to be in first normal form? How do you convert an unnormalized design to 1NF?
7. What are the criteria for a table design to be in second normal form? How do you convert a 1NF design to 2NF?
8. What are the criteria for a table design to be in third normal form? How do you convert a 2NF design to 3NF?
9. Explain the difference between a logical record and a physical record.
10. How would a specific date, such as September 1, 2011, be represented as an absolute date?

Discussion Topics

1. Are there ethical issues to consider when planning a database? For example, should sensitive personal data (such as medical information) be stored in the same DBMS that manages employee salary and benefits data? Why or why not?
2. Suggest three typical business situations where referential integrity avoids data problems.
3. Consider an automobile dealership with three locations. Data fields exist for stock number, vehicle identification number, make, model, year, color, and invoice cost. Identify the possible candidate keys, the likely primary key, a probable foreign key, and potential secondary keys.
4. In the example shown in Figures 9-25 and 9-26 on pages 416 and 417, the 2NF customer table was converted to two 3NF tables. Verify that the four potential problems identified for 2NF tables were eliminated in the 3NF design.

Projects

1. Search the Internet to find information about data storage formats. Also do research on international date formats. Determine whether the date format used in the United States is the most common format.
2. Visit the IT department at your school or at a local business and determine whether the organization uses file-oriented systems, DBMSs, or both. Write a brief memo with your conclusions.
3. Use Microsoft Access or similar database software to create a DBMS for the imaginary company called TopText Publishing, which is described in Case In Point 9.1 on page 410. Add several sample records to each table and report to the class on your progress.
4. Visit the bookstore at your school or a bookstore in your area. Interview the manager or store employees to learn how the operation works and what entities are involved in bookstore operations. Remember that an entity is a person, place, thing, or event that affects the information system. Draw an ERD, including cardinality that describes the bookstore operations.

Apply Your Knowledge

The Apply Your Knowledge section contains four mini-cases. Each case describes a situation, explains your role in the case, and asks you to respond to questions. You can answer the questions by applying knowledge you learned in the chapter.

1 Pick and Shovel Construction Company

Situation:

Pick and Shovel Construction Company is a multistate building contractor specializing in medium-priced town homes. C. T. Scott, the owner, is in your office for the third time today to see how the new relational database project is coming along. Unfortunately, someone mentioned to C. T. that the delay had something to do with achieving *normalization*.

"Why is all this normalization stuff so important?" he asks. "The old system worked OK most of the time, and now you are telling me that we need all these special rules. Why is this necessary?"

1. How should you respond to C. T.? Write him a brief memo with your views.
2. Assume that the Pick and Shovel's main entities are its customers, employees, projects, and equipment. A customer can hire the company for more than one project, and employees sometimes work on more than one project at a time. Equipment, however, is assigned only to one project. Draw an ERD showing those entities.
3. Add cardinality notation to your ERD.
4. Create 3NF table designs.

2 Puppy Palace

Situation:

Puppy Palace works with TV and movie producers who need dogs that can perform special tricks, such as headstands, somersaults, ladder climbs, and various dog-and-pony tricks. Puppy Palace has about 16 dogs and a list of 50 tricks from which to choose. Each dog can perform one or more tricks, and many tricks can be performed by more than one dog. When a dog learns a new trick, the trainer assigns a skill level. Some customers insist on using dogs that score a 10, which is the highest skill level. As an IT consultant, you have been asked to suggest 3NF table designs. You are fairly certain that a M:N relationship exists between dogs and tricks.

1. Draw an ERD for the Puppy Palace information system.
2. Indicate cardinality.
3. Identify all fields you plan to include in the dogs and tricks tables. For example, in the dogs table, you might want breed, size, age, name, and so on. In the tricks table, you might want the trick name and description. You will need to assign a primary key in each table. ***Hint***: Before you begin, review some database design samples in this chapter. You might spot a similar situation that requires an associative entity that you can use as a pattern. In addition, remember that numeric values work well in primary key fields.
4. Create 3NF table designs.

3 Mayville Public Library

Situation:

Mayville is a rural village with a population of 900. Until now, Mayville was served by a bookmobile from a larger town. The Mayville Village Council has authorized funds for a small public library, and you have volunteered to set up an information system for the library. Assume that the library will have multiple copies of certain books.

1. Draw an ERD for the Mayville library system.
2. Indicate cardinality.
3. Identify all fields you plan to include in the tables.
4. Create 3NF table designs.

4 Western Wear Outfitters

Situation:

Western Wear is a mail-order firm that offers an extensive selection of casual clothing for men and women. Western Wear plans to launch a new Web site, and the company wants to develop a new set of product codes. Currently, 650 different products exist, with the possibility of adding more in the future. Many products come in various sizes, styles, and colors. The marketing manager asked you to develop an individualized product code that can identify a specific item and its characteristics. Your initial reaction is that it can be done, but the code might be fairly complex. Back in your office, you give the matter some thought.

1. Design a code scheme that will meet the marketing manager's stated requirements.
2. Write a brief memo to the marketing manager suggesting at least one alternative to the code she proposed, and state your reasons.
3. Suggest a code scheme that will identify each Western Wear customer.
4. Suggest a code scheme that will identify each specific order.

Case Studies

Case studies allow you to practice specific skills learned in the chapter. Each chapter contains several case studies that continue throughout the textbook, and a chapter capstone case.

New Century Health Clinic

New Century Health Clinic offers preventive medicine and traditional medical care. In your role as an IT consultant, you will help New Century develop a new information system.

Background

After completing the user interface, input, and output design for the new information system at New Century, you will consider data design issues. Begin by studying the DFDs and object-oriented diagrams you prepared previously and the rest of the documentation from the systems analysis phase. Perform the following tasks:

Assignments

1. Create an initial entity-relationship diagram for the New Century Health Clinic system.
2. Normalize your table designs.
3. If you identified any new entities during normalization, create a final entity-relationship diagram for the system.
4. Write a memo for your documentation file that contains your recommendation about whether a file processing or a database environment should be used. Attach copies of your ERD(s) and normalized designs.

PERSONAL TRAINER, INC.

Personal Trainer, Inc., owns and operates fitness centers in a dozen Midwestern cities. The centers have done well, and the company is planning an international expansion by opening a new "supercenter" in the Toronto area. Personal Trainer's president, Cassia Umi, hired an IT consultant, Susan Park, to help develop an information system for the new facility. During the project, Susan will work closely with Gray Lewis, who will manage the new operation.

Background

After evaluating various development strategies, Susan prepared a system requirements document and submitted her recommendations to Cassia Umi, Personal Trainer's president. During her presentation, Susan discussed several development strategies, including in-house development and outsourcing. She did not feel that a commercial software package would meet Personal Trainer's needs.

Based on her research, Susan felt it would be premature to select a development strategy at this time. Instead, she recommended to Cassia that an in-house team should develop a design prototype, using a relational database as a model. Susan said that the prototype would have two main objectives: It would represent a user-approved model of the new system, and it would identify all systems entities and the relationships among them. Susan explained that it would be better to design the basic system first, and then address other issues, including Web enhancements and implementation options. She proposed a three-step plan: data design, user interface design, and application architecture. She explained that systems analysts refer to this as the systems design phase of a development project.

Cassia agreed with Susan's recommendation, and asked her to go forward with the plan.

Assignments

1. Review the Personal Trainer fact-finding summary in Chapter 4 and draw an ERD with cardinality notation. Assume that system entities include members, activities and services, and fitness instructors.
2. Design tables in 3NF. As you create the database, include various codes for at least three of the fields.
3. Use sample data to populate the fields for at least three records in each table.
4. Recommend a date format for the new system. Should Personal Trainer adopt a single international standard, or should the format be determined by the country in which the center is located? Write a message to Susan with your recommendation.

Fastflight Airlines

FastFlight Airlines is a small air carrier operating in three northeastern states. FastFlight is computerizing its passenger reservation system. The data items must include reservation number, flight number, flight date, origin, destination, departure time, arrival time, passenger name, and seat number. For example, flight number 303 leaves Augusta, Maine, daily at 9:23 a.m. and arrives in Nashua, New Hampshire, at 10:17 a.m. A typical reservation code might be AXQTBC, for passenger Lisa Lane, in seat 4A, on flight 303 on 11/12/2011.

Assignments

1. Create an ERD for the reservations system.
2. Create 3NF table designs for the system.
3. For each of the entities identified, design tables and identify the possible candidate keys, the primary key, a probable foreign key, and potential secondary keys.
4. Use sample data to populate the fields for three records.

CHAPTER CAPSTONE CASE: SoftWear, Limited

SoftWear, Limited (SWL), is a continuing case study that illustrates the knowledge and skills described in each chapter. In this case study, the student acts as a member of the SWL systems development team and performs various tasks.

Background

Work continued on the ESIP system. Rick said that the system would be a client/server design that could support SWL's current and future business requirements. He also said that Ann Hon wanted to use the ESIP system as a prototype for developing other SWL systems in the future. Ann said that the new design would have to be powerful, flexible, and scalable. With that in mind, the team decided that a DBMS strategy was the best solution for SWL's future information systems requirements.

ERDs and Normalization

Rick asked Tom and Becky to draw an entity-relationship diagram with normalized designs. Tom and Becky used the Visible Analyst, a CASE tool, to produce the diagram shown in Figure 9-50. Rick noticed that only two entities were shown: EMPLOYEE and DEDUCTION. Rick suggested that the ESIP-OPTION and HUMAN RESOURCES entities should be added. Tom and Becky agreed. The second version of their ERD is shown in Figure 9-51 on the next page.

With the ERD completed, Tom turned to the design of the EMPLOYEE table. He suggested the following design:

EMPLOYEE (SSN, EMPLOYEE-NAME, HIRE-DATE)

"The table obviously is in 1NF, because it has no repeating groups," Tom said. "It's also in 2NF, because it has a single field as the primary key. And I'm sure it's in 3NF, because the

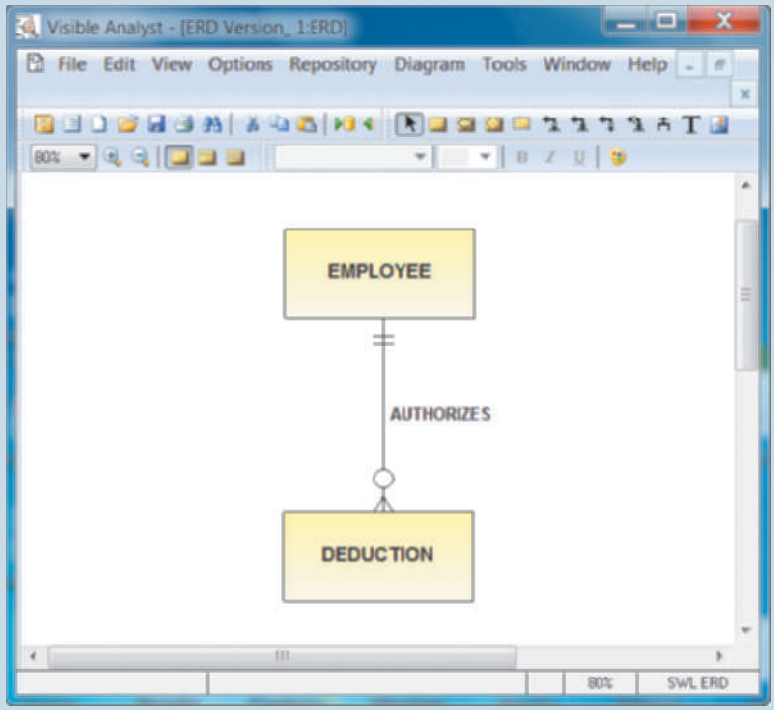


FIGURE 9-50 The initial ERD shows two entities: EMPLOYEE and DEDUCTION. Notice that crow's foot notation indicates that one and only one employee can authorize anywhere from zero to many deductions.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

employee name and the hire date both depend on the Social Security number." Everyone agreed that this was the correct design. Tom and Becky turned their attention to designing the ESIP-OPTION table and later suggested the following design:

ESIP-OPTION (OPTION-CODE, OPTION-NAME, DESCRIPTION, DEDUCTION-CYCLE, APPLICATION-CYCLE, MIN-SERVICE, MIN-DEDUCTION, MAX-DEDUCTION)

The design appeared to meet the test for 3NF. Although seven fields existed in addition to the primary key, each field seemed to depend on the entire primary key. Finally, Becky proposed the following design for the DEDUCTION record:

DEDUCTION (SSN, ESIP-OPTION, DATE, EMPLOYEE-NAME, AMOUNT)

This time, Rick felt that the table was in 1NF, but not in 2NF because the EMPLOYEE-NAME field was dependent on only a part of the key rather than the entire key. Becky agreed with him and suggested that the EMPLOYEE-NAME field could be removed and accessed using the EMPLOYEE table. The SSN field could be used as a foreign key to match values in the EMPLOYEE table's primary key. To put the table into 2NF, she revised the DEDUCTION table design as follows:

DEDUCTION (SSN, ESIP-OPTION, DATE, AMOUNT)

With that change, everyone agreed that the table also was in 3NF because the AMOUNT field depended on the entire primary key. The next step was to work on a system design to interface with the payroll system and provide support for SWL's long-term information technology goals.

While Rick, Tom, and Becky were working on ERDs and normalization, Pacific Software delivered the payroll package that SWL ordered. Tom was assigned to work on installing and configuring the package and training users on the new payroll system.

Meanwhile, Rick felt that SWL should get more information about client/server design. With Ann Hon's approval, he contacted several IT consulting firms that advertised their client/server design expertise on the Internet. Rick and Becky met with three firms and recommended that SWL work with True Blue Systems, a consulting group with a local office in Raleigh, not far from SWL's headquarters. In addition to the design for the ESIP system, Rick suggested that the agenda should include the possibility that employees could access their ESIP accounts from home via the Internet.

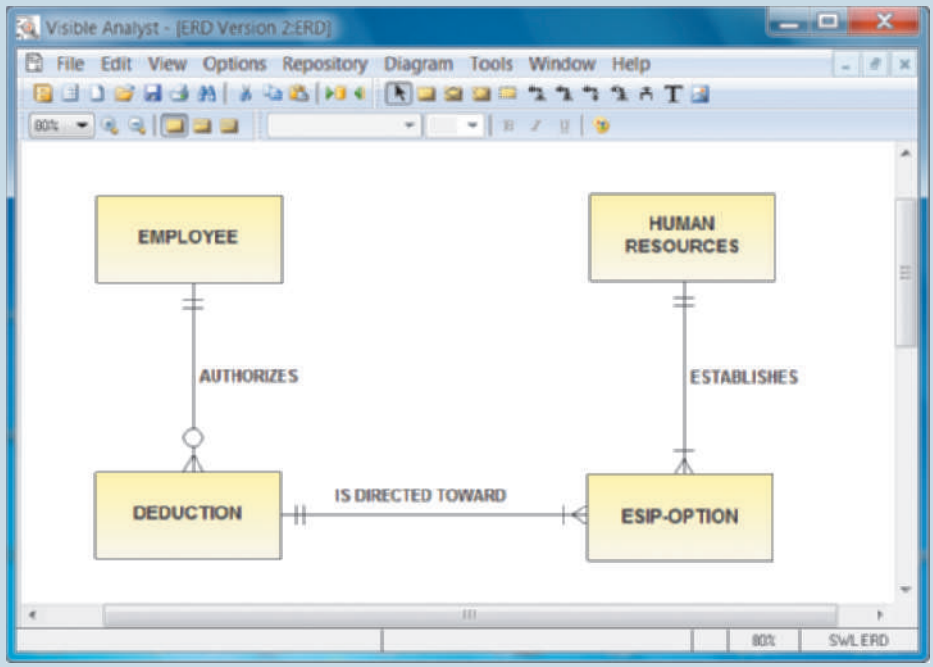


FIGURE 9-51 Second version of the ERD. Now, the DEDUCTION entity has relationships to two other entities: EMPLOYEE and ESIP-OPTION.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

SWL Team Tasks

1. Rick asked you to help him put together a brief progress update for Michael Jeremy and several other top managers. Specifically, Rick wants you to explain the concept of normalization without using a lot of technical jargon. Rick wants you to summarize the concept using plain English and simple examples.
2. At SWL, each employee is assigned to a specific department. Employees from several departments often are assigned to special project teams, however, when a new product is launched or for major marketing events. Carla wants to develop a project management system to track the projects, employees assigned, and accumulated project hours. She believes that employees and projects are in a M:N relationship. She showed you an initial design where all data is stored in a single table:

   PROJECT DATA (PROJECT-NUMBER, PROJECT-NAME, START-DATE, PROJECT-STATUS, (EMPLOYEE-NUMBER, EMPLOYEE-NAME, JOB-TITLE, DEPT-NUMBER, DEPT-NAME, PROJECT-HOURS))

   How would you describe Carla's design?
3. Carla wants you to create an ERD, including cardinality, for the project management system. She says that you probably will need to add an associative entity.
4. After you create the ERD in the previous step, design a table for each entity, in third normal form.

Manage the SWL Project

You have been asked to manage SWL's new information system project. One of your most important activities will be to identify project tasks and determine when they will be performed. Before you begin, you should review the SWL case in this chapter. Then list and analyze the tasks, as follows:

LIST THE TASKS Start by listing and numbering at least 10 tasks that the SWL team needs to perform to fulfill the objectives of this chapter. Your list can include SWL Team Tasks and any other tasks that are described in this chapter. For example, Task 3 might be to Identify all entities, and Task 6 might be to Create an initial ERD.

ANALYZE THE TASKS Now study the tasks to determine the order in which they should be performed. First identify all concurrent tasks, which are not dependent on other tasks. In the example shown in Figure 9-52 on the next page, Tasks 1, 2, 3, 4, and 5 are concurrent tasks, and could begin at the same time if resources were available.

Other tasks are called dependent tasks, because they cannot be performed until one or more earlier tasks have been completed. For each dependent task, you must identify specific tasks that need to be completed before this task can begin. For example, you would want to identify all the entities before you could create an initial ERD, so Task 6 cannot begin until Task 3 is completed, as Figure 9-52 shows.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

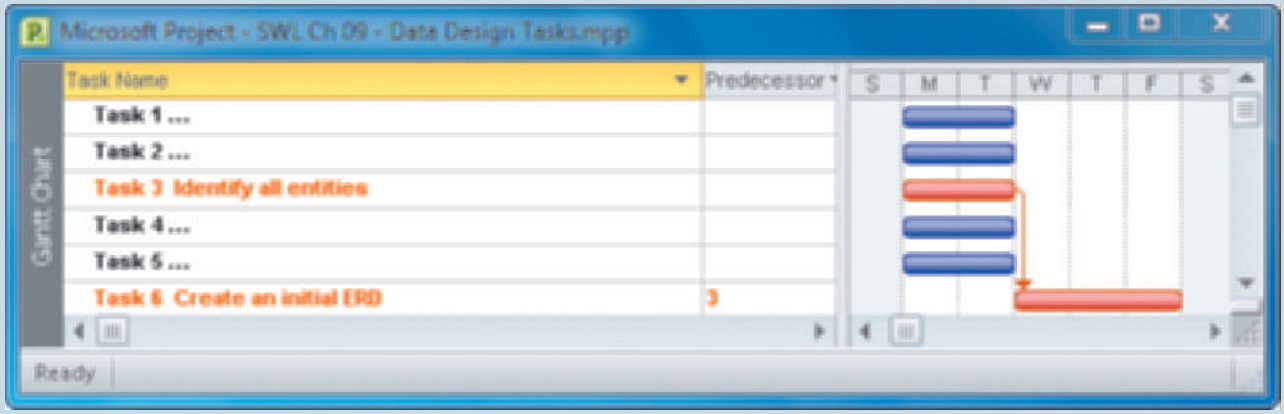


FIGURE 9-52 Tasks 1, 2, 3, 4, and 5 are concurrent tasks that could be performed at the same time. Task 6 is a dependent task that cannot be performed until Task 3 has been completed.

Chapter 3 describes project management tools, techniques, and software. To learn more, you can use the Features section on your Student Study Tool CD-ROM, or visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and locate the project management resources library for this book. On the Web, Microsoft offers demo versions, training, and tips for using Project 2010. You also can visit the OpenWorkbench.org site to learn more about this free, open-source software.

Ready for a Challenge?

In additional to technical skills, IT professionals need critical thinking skills such as perception, organization, analysis, problem-solving, and decision-making. The Ready for a Challenge feature can help you learn, practice, and apply critical thinking skills that you can take to the workplace.

This week, the IT team at Game Technology is working on database design and codes. Your job is to develop a recommendation for keeping track of individual games and their authors. You will be expected to create an ERD and table designs. Here is what you know so far:

- Every game has a product ID, name, version number, category code, and category description.
- Every author has an author ID, name, mailing address, and e-mail address.
- An individual author can develop one or more games.
- A game also can be developed by several authors. In this case, each author receives a specific percentage of the royalty that the game earns.

This information must be stored in the database. Your plan is to develop an overall design, list all the individual data items, and then include them as fields in the tables you create. Your design must be in the form of an ERD that shows all the entities, including any associative entities, and their relationships. Also, all table designs must be in third normal form (3NF).

The IT team also discussed data codes. You read somewhere that six-letter codes can have over 300 million unique combinations, but you don't know the exact number. Your friend, a math major, said that you could figure out the answer on your own, and she gave you a hint by saying it was 26 to the sixth power.

Practice Tasks

A. Draw an ERD showing all entities and their relationships, and create table designs that include all necessary fields.
B. Calculate the exact number of unique combinations that are possible using a six-letter code.

After you complete the Practice Tasks, to check your work and view sample answers, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the resources for this chapter, and locate Ready for a Challenge?.

The Challenge

Your recommended design ran into a snag. The team leader said it wasn't bad, but it did not go far enough. The specific comment was that it was not in third normal form.

Also, the team finally decided on a code for the customer ID field, which is a primary key. The code will be a combination code, with four letters followed by four digits. The letters would be the first four letters of the customer's last name, which might make it easier for customers to remember.

Challenge Tasks

A. Review the sample ERD and table designs, and convert them to third normal form (3NF).
B. Calculate the exact number of possible combinations with a code composed of four letters followed by four numbers. Does this seem like a good choice? Why or why not? What would you suggest?

CHAPTER 10 System Architecture

Chapter 10 is the final chapter in the systems design phase of the SDLC. This chapter describes system architecture, which translates the logical design of an information system into a physical blueprint, or architecture. As you plan the system architecture, you will learn about servers, clients, processing methods, networks, and related issues.

OBJECTIVES

When you finish this chapter, you will be able to:

- Provide a checklist of issues to consider when selecting a system architecture
- Describe servers, server-based processing, clients, and client-based processing
- Explain client/server architecture, including tiers, cost-benefit issues, and performance
- Compare in-house e-commerce development with packaged solutions
- Discuss the potential impact of cloud computing and Web 2.0
- Explain the difference between online and batch processing
- Define network topology, including hierarchical, bus, ring, and star models
- Explain network protocols and licensing issues
- Describe wireless networking, including wireless standards, topologies, and trends
- Describe the system design specification

INTRODUCTION

At this point in the SDLC, your objective is to determine an overall architecture to implement the information system. You learned in Chapter 1 that an information system requires hardware, software, data, procedures, and people to accomplish a specific set of functions. An effective system combines those elements into an architecture, or design, that is flexible, cost-effective, technically sound, and able to support the information needs of the business. This chapter covers a wide range of topics that support the overall system design, just as a plan for a new home would include a foundation plan, building methods, wiring and plumbing diagrams, traffic flows, and costs.

System architecture translates the logical design of an information system into a physical structure that includes hardware, software, network support, processing methods, and security. The end product of the systems design phase is the system design specification. If this document is approved, the next step is systems implementation.

Chapter Introduction Case: Mountain View College Bookstore

Background: Wendy Lee, manager of college services at Mountain View College, wants a new information system that will improve efficiency and customer service at the three college bookstores.

In this part of the case, Tina Allen (systems analyst) and David Conroe (student intern) are talking about system architecture issues.

Participants: Tina and David
Location: Mountain View College cafeteria, Monday afternoon, January 9, 2012
Project status: The team has completed data design tasks and user interface, output, and input design work. The last step in the systems design phase is to consider a system architecture for the bookstore system.
Discussion topics: System architecture checklist, client/server architecture, processing methods, network issues, and system management tools

Tina: Hi, David. Did you enjoy the holiday break?

David: *I sure did. Now I'm ready to get back to work.*

Tina: Good. As the last step in the systems design phase of the SDLC, we need to study the physical structure, or architecture, of the bookstore system. Our checklist includes enterprise resource planning, total cost of ownership, scalability, Web integration, legacy systems, processing methods, and security issues that could affect the system design.

David: *Where do we start?*

Tina: Well, the bookstore interfaces with many publishers and vendors, so we'll consider supply chain management, which is part of enterprise resource planning, or ERP for short.

David: *What happens after we finish the checklist?*

Tina: Then we'll define a client/server architecture. As I see it, the bookstore client workstations will share the processing with a server in the IT department. Also, we may need to look at middleware software to connect the new system with existing legacy systems, such as the college accounting system.

David: *Anything else?*

Tina: Yes. We need to select a network plan, or topology, so we'll know how to plan the physical cabling and connections — or possibly use wireless technology. When we're done, we'll submit a system design specification for approval.

David: *Sounds good to me.*

Tina: Good. Here's a task list to get us started:

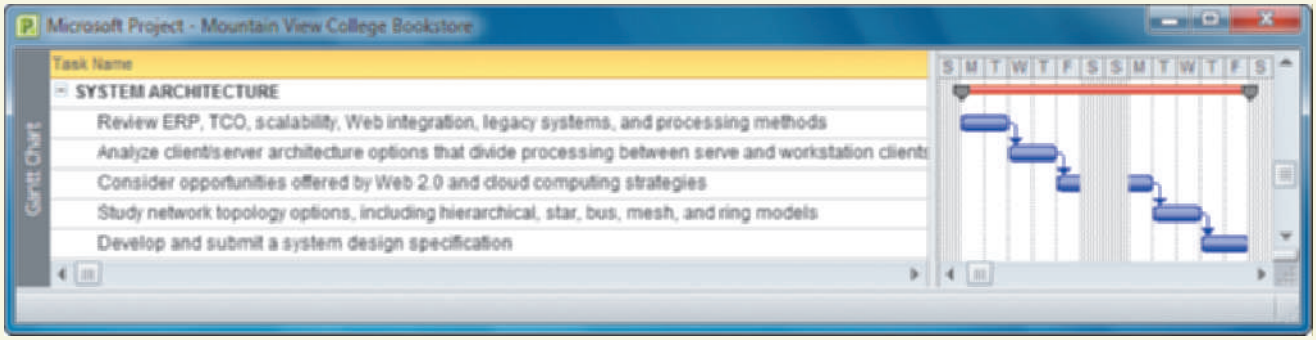


FIGURE 10-1 Typical system architecture tasks.

System Architecture Checklist

Just as an architect begins a project with a list of the owner's requirements, a systems analyst must approach system architecture with an overall checklist. Before making a decision, the analyst must consider seven specific issues that will affect the architecture choice:

- Enterprise resource planning (ERP)
- Initial and total cost of ownership (TCO)
- Scalability
- Web integration
- Legacy system interface requirements
- Processing options
- Security issues

Enterprise Resource Planning (ERP)

Many companies use **enterprise resource planning** (**ERP**) software, which was described in Chapter 1. The objective of ERP is to establish a company-wide strategy for using IT resources. ERP defines a specific architecture, including standards for data, processing, network, and user interface design. A main advantage of ERP is that it describes a specific hardware and software **environment**, also called a **platform**, that ensures connectivity and easy integration of future systems, including in-house software and commercial packages.

In the article in *CIO* magazine shown in Figure 10-2, Thomas Wailgum remarks that ERP is like watching the TV series *Lost*, because both have subplots, crashes, and interesting characters. In an earlier article, Mr. Wailgum pointed out that ERP isn't really about resources or planning — it is about the *enterprise*, and the concept of integrating departmental systems into an overall application, with a central database that can be shared and used effectively.

FIGURE 10-2 The article by Thomas Wailgum makes an interesting comparison between ERP and a popular TV series.

Many companies are extending internal ERP systems to their suppliers and customers, using a concept called **supply chain management** (**SCM**). For example, in a totally integrated supply chain system, a customer order could cause a manufacturing system to schedule a work order, which in turn triggers a call for more parts from one or more suppliers. In a dynamic, highly competitive economy, SCM can help companies achieve faster response, better customer service, and lower operating costs. Most supply chain management systems depend on RFID technology for real-time input data. As you learned in Chapter 1, RFID allows companies to track incoming material, current production, and finished inventory by using small devices that respond to radio frequency signals.

Microsoft offers an ERP solution called Microsoft Dynamics, as shown in Figure 10-3. The company claims that the software can integrate financial management, customer relationship management (CRM),

FIGURE 10-3 Microsoft Dynamics is an ERP strategy that can integrate separate systems and improve productivity throughout the organization.

supply chain management (SCM), project management, human resources, and business intelligence reporting.

CASE IN POINT 10.1: ABC SYSTEMS

You are a systems analyst at ABC Systems, a fast-growing IT consulting firm that provides a wide range of services to companies that want to establish e-commerce operations. During the last 18 months, ABC acquired two smaller firms and set up a new division that specializes in supply chain management. Aligning ABC's internal systems was quite a challenge, and top management was not especially happy with the integration cost or the timetable. To avoid future problems, you have decided to suggest an ERP strategy, and you plan to present your views at the staff meeting tomorrow. ABC's management team is very informal and prefers a loose, flexible style of management. How will you persuade them that ERP is the way to go?

Initial Cost and TCO

You learned earlier about the importance of considering economic feasibility and TCO during systems planning and analysis. As Figure 10-4 on the next page suggests, TCO includes tangible purchases, fees, and contracts called *hard* costs. However, additional *soft* costs of management, support, training, and downtime are just as important, but more difficult to measure. As the pie chart shows, the combination of user-related and operational costs exceeds hardware and software acquisition costs.

A TCO analysis should include the the following questions.

- If in-house development was selected as the best alternative initially, is it still the best choice? Is the necessary technical expertise available, and does the original cost estimate appear realistic?

- If a specific package was chosen initially, is it still the best choice? Are newer versions or competitive products available? Have any changes occurred in pricing or support?
- Have any new types of outsourcing become available?
- Have any economic, governmental, or regulatory events occurred that could affect the proposed project?
- Have any significant technical developments occurred that could affect the proposed project?
- Have any major assumptions changed since the company made the build versus buy decision?
- Are there any merger or acquisition issues to consider, whereby the company might require compatibility with a specific environment?
- Have any new trends occurred in the marketplace? Are new products or technologies on the verge of being introduced?
- Have you updated the original TCO estimate? If so, are there any significant differences?

The answers to these questions might affect the initial cost and TCO for the proposed system. You should review system requirements and alternatives now, before proceeding to design the system architecture.

Scalability

A network is composed of individual nodes. A **node** represents a physical device, wired or wireless, that can send, receive, or manage network data. For example, nodes can be servers, workstations, shared printers, mass storage devices, wireless access points, or mobile computers.

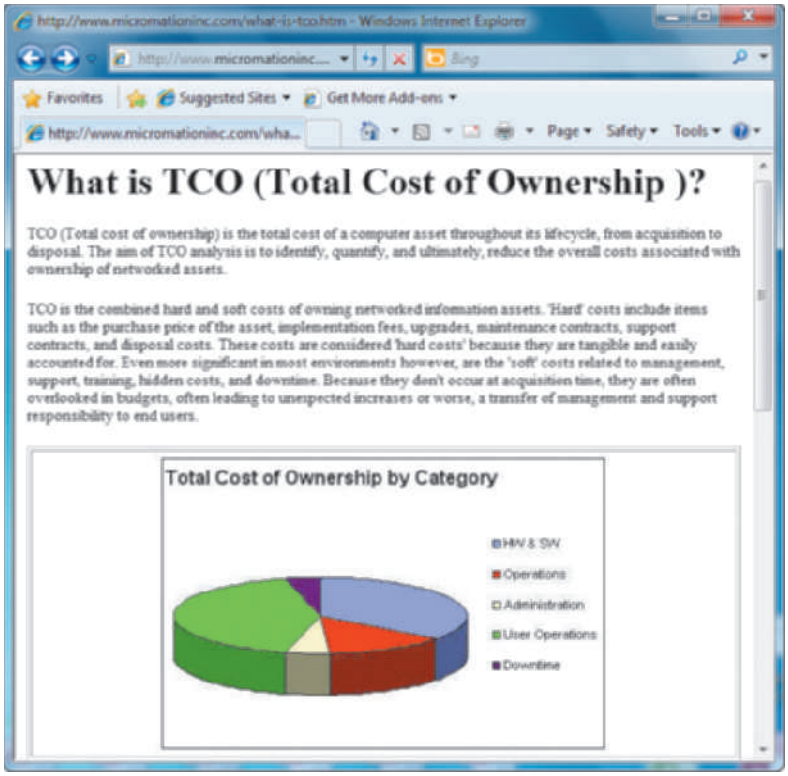


FIGURE 10-4 The Micromation site suggests that *soft* costs are very significant, but are more difficult to measure.

Scalability, also called **extensibility**, refers to a system's ability to expand, change, or downsize easily to meet the changing needs of a business enterprise. Scalability is especially important in implementing systems that are volume-related, such as transaction processing systems. A scalable system is necessary to support a dynamic, growing business. For example, a scalable network could handle anywhere from a few dozen nodes to thousands of nodes; a scalable DBMS could support the acquisition of a new sales division. When investing large amounts of money in a project, management is especially concerned about scalability issues that could affect the system's life expectancy.

Web Integration

An information system includes **applications**, which are programs that handle the input, manage the processing logic, and provide the required output. The systems analyst must know if a new application will be part of an e-commerce strategy and the

degree of integration with other Web-based components. As you learned earlier, a **Web-centric** architecture follows Internet design protocols and enables a company to integrate the new application into its e-commerce strategy. Even where e-commerce is not involved, a Web-centric application can run on the Internet or a company intranet or extranet. A Web-based application avoids many of the connectivity and compatibility problems that typically arise when different hardware environments are involved. In a Web-based environment, a firm's external business partners can use standard Web browsers to import and export data.

Figure 10-5 shows IBM's WebSphere software. WebSphere offers Java-based ERP solutions as well as software development tools that customers can use to build their own Web-centric applications.

ON THE WEB

To learn more about TCO, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the TCO link.

Legacy System Interface Requirements

A new system might have to interface with one or more **legacy systems**, which are older systems that use outdated technology, but still are functional. For example, a new marketing information system might need to report sales data to a server-based accounting system and obtain product cost data from a legacy manufacturing system.

Interfacing a new system with a legacy system involves analysis of data formats and compatibility. In some cases, a company will need to convert legacy file data, which can be an expensive and time-consuming process. Middleware, which is discussed later in this chapter, might be needed to pass data between new systems and legacy systems. Finally, to select the best architecture, the analyst must know if the new application eventually will replace the legacy system.

ON THE WEB

To learn more about legacy systems, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Legacy Systems link.

FIGURE 10-5 WebSphere offers ERP solutions and development tools.

Processing Options

In planning the architecture, designers also must consider how the system will process data — online or in batches. For example, a high-capacity transaction processing system, such as an order entry system, requires more network, processing, and data storage resources than a monthly billing system that handles data in batches. Also, if the system must operate online, 24 hours a day and seven days a week (24/7), provision must be made for backup and speedy recovery in the event of system failure.

The characteristics of online and batch processing methods are described later in this chapter, with examples of each type.

Security Issues

From the password protection shown in Figure 10-6 to complex intrusion detection systems, security threats and defenses are a major concern to a systems analyst. As the physical design is translated into specific hardware and software, the analyst must consider security issues and determine how the company will address them. Security is especially important when data or processing is performed at remote locations, rather than at a centralized facility. In mission-critical systems, security issues will have a major impact on system architecture and design.

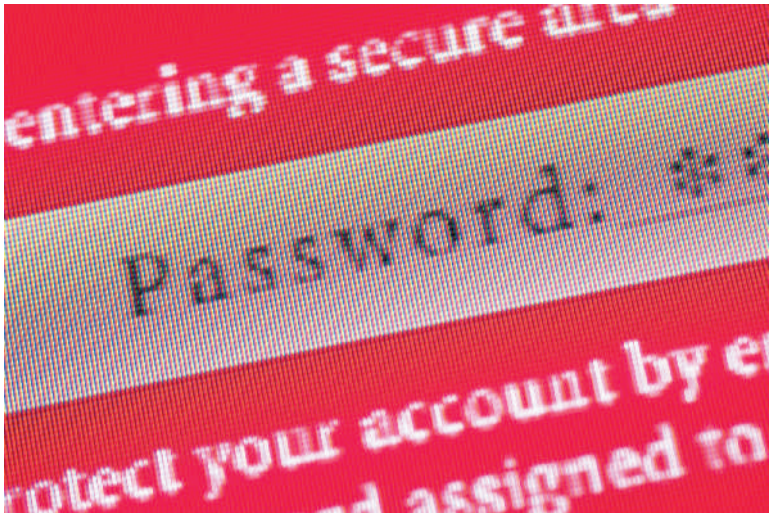


FIGURE 10-6 User IDs and passwords are important elements of system security.

Web-based systems introduce additional security concerns, as critical data must be protected in the Internet environment. Also, firms that use e-commerce applications must assure customers that their personal data is safe and secure. System security concepts and strategies are discussed in detail in Chapter 12, Managing Systems Support and Security.

Planning the Architecture

Every information system involves three main functions: data storage and access methods, application programs to handle the processing logic, and an interface that allows users to interact with the system. Depending on the architecture, the three functions are performed on a server, on a client, or are divided between the server and the client. As you plan the system design, you must determine where the functions will be carried out and the advantages and disadvantages of each design approach. This section discusses server and client characteristics and how each design alternative handles system functions.

Servers

A **server** is a computer that supplies data, processing services, or other support to one or more computers, called **clients**. A system design where the server performs *all* the processing sometimes is described as **mainframe architecture**. Although the actual server does not have to be a mainframe, the term mainframe architecture typically describes a multiuser environment where the server is significantly more powerful than the clients. A systems

analyst should know the history of mainframe architecture to understand the server's role in modern system design.

MAINFRAME HISTORY In the 1960s, mainframe architecture was the *only* system design available. In addition to centralized data processing, early systems performed all data input and output at a central location, often called a **data processing center**. Physical data was delivered or transmitted in some manner to the data processing center, where it was entered into the system. Users in the organization had no input or output capability, except for printed reports that were distributed by a corporate IT department.

SERVER-BASED PROCESSING As network technology advanced and became affordable, companies installed terminals at remote locations, so that users could enter and access data from anywhere in the organization, regardless of where the centralized computer was located. A **terminal** included a keyboard and display screen to handle input and output, but lacked independent processing capability. In a centralized design, as shown in Figure 10-7, the remote user's keystrokes are transmitted from his or her terminal to the mainframe, which responds by sending screen output back to the user's screen.

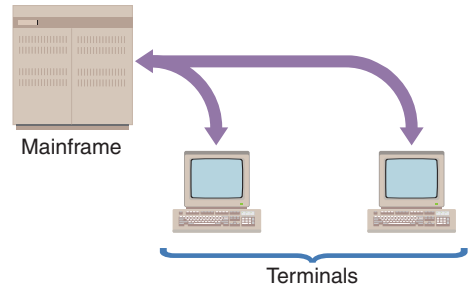


FIGURE 10-7 In a centralized design, the remote user's keystrokes are transmitted to the mainframe, which responds by sending screen output back to the user's screen.

A main advantage of server-based processing is that various types of terminals can communicate with the mainframe, and the design is not tied to a specific hardware platform. A disadvantage is that server-based processing typically uses character-based terminals that provide a limited interface for users. In a server-based system, all data storage, access, and application programs are located on the mainframe.

Today, mainframe architecture still is used in industries that require large amounts of data processing that can be done in batches at a central location. For example, a credit card company might run monthly statements in a batch, or a bank might use mainframe servers to update customer balances each night. In a blend of old and new technology, an Internet-based retail operation might use centralized data management at a customer service center to support and manage its online sales activity, as shown in Figure 10-8.

As server technology evolved, terminals also changed dramatically. Instead of simple input–output devices, a company might use a mix of PCs, handheld computers, and other specialized hardware that allows users to interact with a centralized server. In most companies, workstations that use powerful GUIs have replaced character-based terminals.

Clients

As PC technology exploded in the 1980s and 1990s, powerful microcomputers quickly appeared on corporate desktops. Users found that they could run their own word processing, spreadsheet, and database applications, without assistance from the IT group, in a mode called stand-alone computing. Before long, companies linked the stand-alone computers into networks that enabled the user clients to exchange data and perform local processing.

FIGURE 10-8 Internet-based retail operations such as Amazon.com use customer service centers to support online sales activity.

STAND-ALONE COMPUTING When an individual user works in **stand-alone** mode, the workstation performs all the functions of a server by storing, accessing, and processing data, as well as providing a user interface. Although stand-alone PCs improved employee productivity and allowed users to perform tasks that previously required IT department assistance, stand-alone computing was inefficient and expensive. Even worse, maintaining data on individual workstations raised major concerns about data security, integrity, and consistency. Without a central storage location, it was impossible to protect and back up valuable business data, and companies were exposed to enormous risks. In some cases, users who were frustrated by a lack of support and services from the IT department created and managed their own databases. In addition to security concerns, this led to data inconsistency and unreliability.

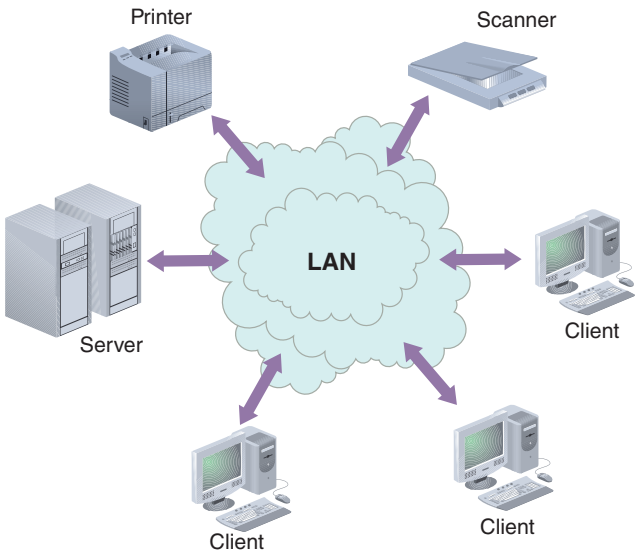


FIGURE 10-9 A LAN allows sharing of data and hardware, such as printers and scanners.

LOCAL AND WIDE AREA NETWORKS As technology became available, companies resolved the problems of stand-alone computing by joining clients into a **local area network** (**LAN**) that allows sharing of data and hardware resources, as shown in Figure 10-9. One or more LANs, in turn, can connect to a centralized server. Further advances in technology made it possible to create powerful networks that could use satellite links, high-speed fiber-optic lines, or the Internet to share data.

A **wide area network** (**WAN**) spans long distances and can connect LANs that are continents apart, as shown in Figure 10-10. When a user accesses data on a LAN or WAN, the network is **transparent** because a user sees the data as if it were stored on his or her own workstation. Company-wide systems that connect one or more LANs or WANs are called **distributed systems**. The capabilities of a distributed system depend on the power and capacity of the underlying data communication network. Compared to mainframe architecture, distributed systems increase concerns about data security and integrity because many individual clients require access to perform processing.

CLIENT-BASED PROCESSING In a typical LAN, clients share data stored on a local server that supports a group of users or a department. As LANs became popular, the most common LAN configuration was a file server design, as shown in Figure 10-11. In a **file server** design, also called a **file sharing architecture**, an individual LAN client has a copy of the application program installed locally, while the data is stored on a central file server. The client requests a copy of the data file and the server responds by transmitting the entire data file to the client. After performing

ON THE WEB

To learn more about local and wide area networks, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Local and Wide Area Networks link.

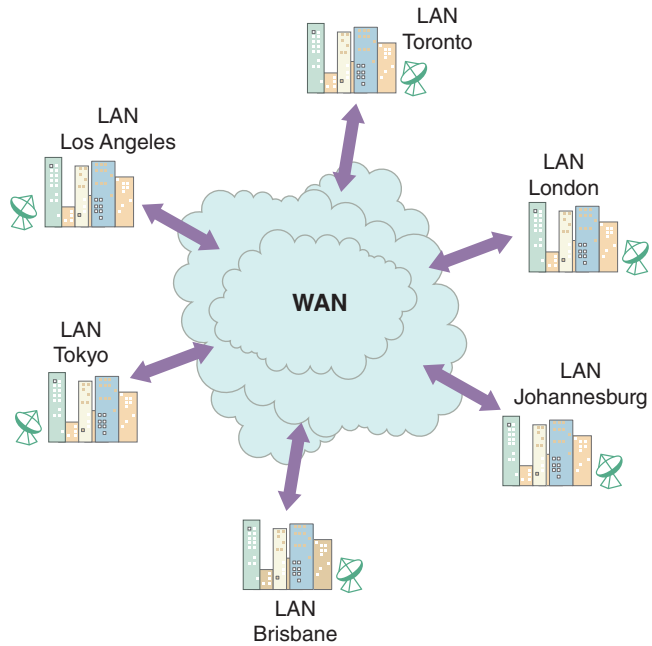


FIGURE 10-10 A WAN can connect many LANs and link users who are continents apart.

the processing locally, the client returns the data file to the central file server where it is stored. File sharing designs are efficient only if the number of networked users is low and the transmitted data file sizes are relatively small. Because the entire data file is sent to each requesting client, a file server design requires significant network resources.

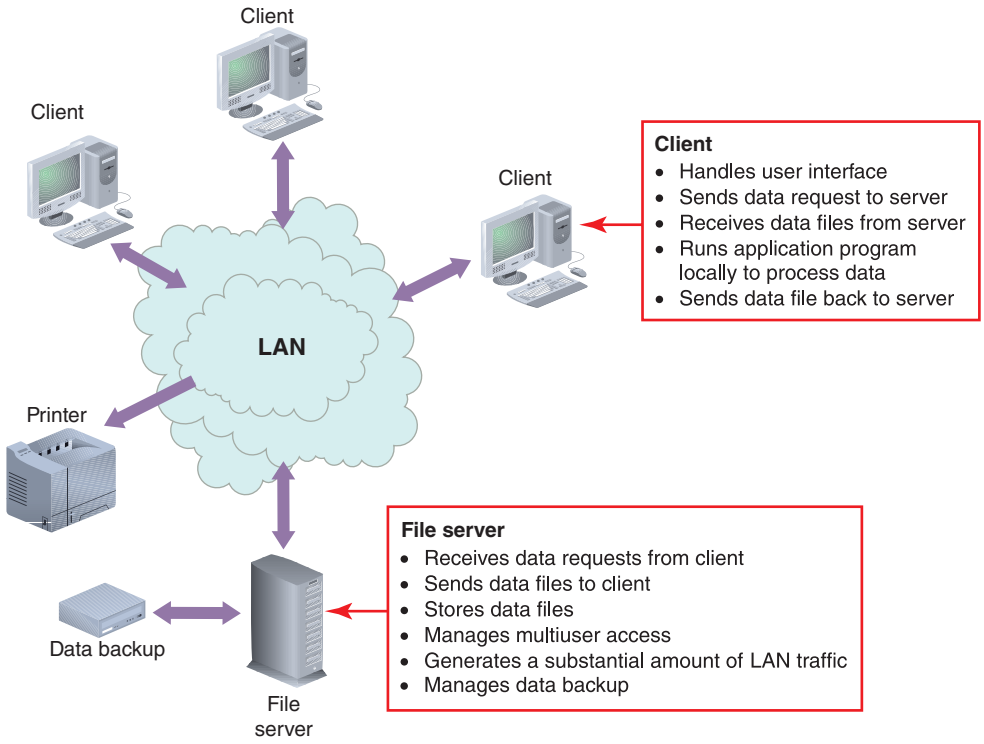


FIGURE 10-11 Example of a LAN file server design. The server stores and manages the data, while the clients run the application program and perform all the processing.

Client/Server Architecture

Today's interconnected world requires an information architecture that spans the entire enterprise. Whether you are dealing with a departmental network or a multinational corporation, as a systems analyst you will work with a distributed computing strategy called client/server architecture.

Overview

Although no standard definition exists, the term **client/server architecture** generally refers to systems that divide processing between one or more networked clients and a central server. In a typical client/server system, the client handles the entire user interface, including data entry, data query, and screen presentation logic. The server stores the data and provides data access and database management functions. Application logic is divided in some manner between the server and the clients. In a client/server interaction, the client submits a request for information from the server, which carries out the operation and responds to the client. As shown in Figure 10-12 on the next page, the data file is not transferred from the server to the client — only the request and the result are transmitted across the network. To fulfill a request from a client, the server might

ON THE WEB

To learn more about client/server architecture, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Client/Server Architecture link.

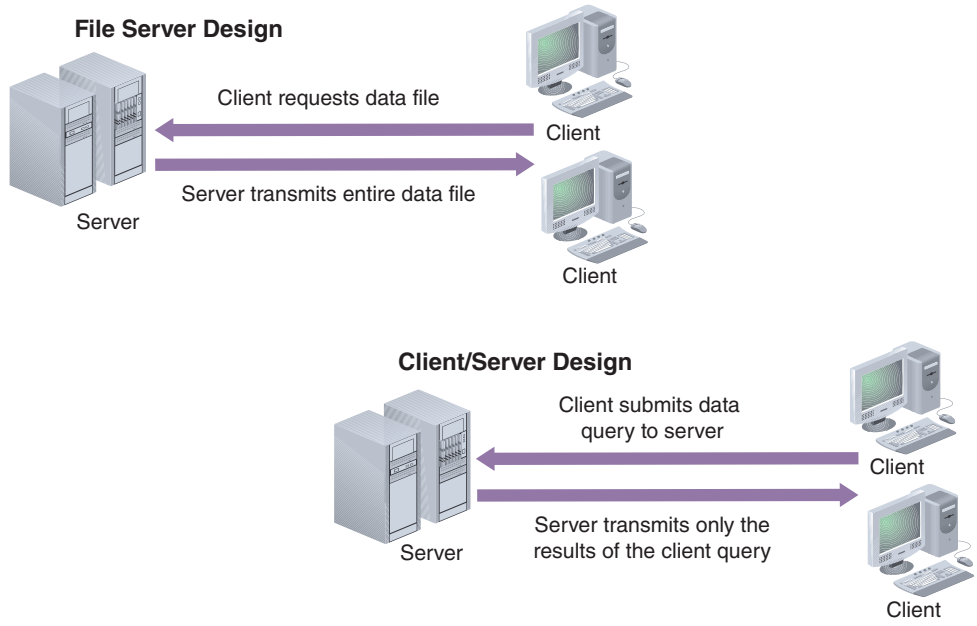


FIGURE 10-12 A file server design compared to a client/server design.

contact other servers for data or processing support, but that process is transparent to the client. The analogy can be made to a restaurant where the customer gives an order to a server, who relays the request to a cook, who actually prepares the meal.

Figure 10-13 lists some major differences between client/server and traditional mainframe systems. Many early client/server systems did not produce expected savings because few clear standards existed, and development costs often were higher than anticipated. Implementation was expensive because clients needed powerful hardware and software to handle shared processing tasks. In addition, many companies had an installed base of data, called **legacy data**, which was difficult to access and transport to a client/server environment.

As large-scale networks grew more powerful, client/server systems became more cost-effective. Many companies invested in client/server systems to achieve a unique combination of computing power, flexibility, and support for changing business operations. Today, client/server architecture is the dominant form of systems design, using Internet protocols and network models such as the ones described on pages 477–480. As businesses form new alliances with customers and suppliers, the client/server concept continues to expand to include clients and servers outside the organization.

Comparison of Client/Server and Mainframe Systems

Characteristics	Client/Server	Mainframe
Basic architecture	Very flexible	Very rigid
Application development	Flexible Fast Object-oriented	Highly structured Slow Traditional
User environment	PC-based GUI Empowers the user Improves productivity	Uses terminals Text interface Constrains the user Limited options
Security and control features	Decentralized Difficult to control	Centralized Easier to control
Processing options	Can be shared and configured in any form desired	Cannot be modified
Data storage options	Can be distributed to place data closer to users	All data is stored centrally
Hardware/software integration	Very flexible Multivendor model	Very rigid Single proprietary vendor

FIGURE 10-13 Comparison of the characteristics of client/server and mainframe systems.

Client/Server Design Styles

Client/server designs can take many forms, depending on the type of server and the relationship between the server and the clients. Figure 10-14 shows the client/server interaction for a database server, a transaction server, an object server, and a Web server. Notice that in each case, the processing is divided between the server and the clients. The nature of the communication depends on the type of server: A database server processes individual SQL commands, a transaction server handles a set of SQL commands, an object server exchanges object messages with clients, and a Web server sends and receives Internet-based communications.

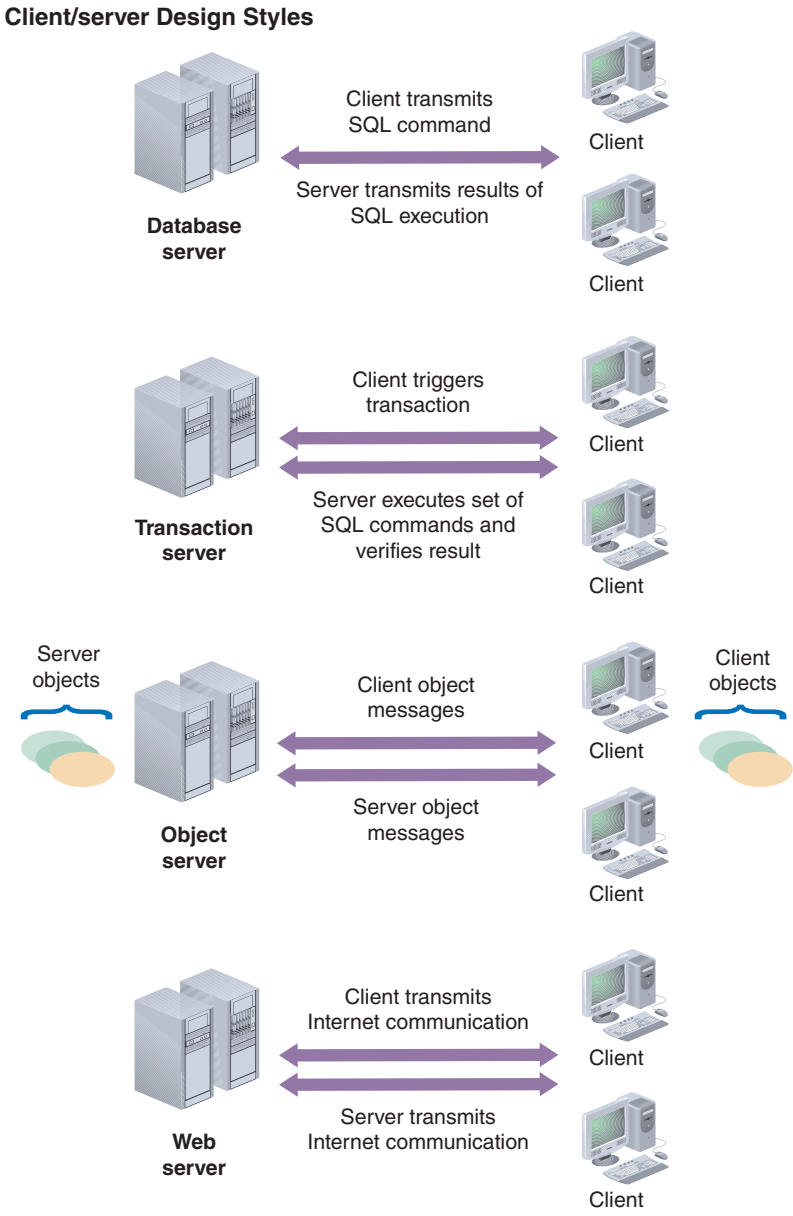


FIGURE 10-14 Client/server interaction for a database server, a transaction server, an object server, and a Web server.

Fat and Thin Clients

Client/server designs can be based on fat or thin clients. A **fat client**, also called a **thick client**, design locates all or most of the application processing logic at the client. A **thin client** design locates all or most of the processing logic at the server. What are the advantages and disadvantages of each design? Most IT experts agree that thin client designs provide better performance, because program code resides on the server, near the data. In contrast, a fat client handles more of the processing and must access and update the data more often. Compared with maintaining a central server, fat client TCO also is higher, because of initial hardware and software requirements and the ongoing expense of supporting and updating remote client computers. A fat client design, however, is simpler and less expensive to develop, because the architecture resembles traditional file server designs where all processing is performed at the client. Figure 10-15 compares the characteristics of fat and thin clients.

Characteristic	Fat Client	Thin Client
Network traffic	Higher, because the fat client must communicate more often with the server to access data and update processing results	Lower, because most interaction between code and data takes place at the server
Performance	Slower, because more network traffic is required	Faster, because less network traffic is required
Initial cost	Higher, because more powerful hardware is required	Lower, because workstation hardware requirements are not as stringent
Maintenance cost	Higher, because more program code resides on the client	Lower, because most program code resides on the central server
Ease of development	Easier, because systems resemble traditional file-server designs where all processing was performed at the client	More difficult, because developers must optimize the division of processing logic

FIGURE 10-15 Characteristics of fat and thin clients.

Client/Server Tiers

Early client/server designs were called two-tier designs. In a **two-tier** design, the user interface resides on the client, all data resides on the server, and the application logic can run either on the server or on the client, or be divided between the client and the server.

More recently, another form of client/server design, called a three-tier design, has become popular. In a **three-tier** design, the user interface runs on the client and the data is stored on the server, just as with a two-tier design. A three-tier design also has a middle layer between the client and server that processes the client requests and translates them into data access commands that can be understood and carried out by the server, as shown in Figure 10-16. You can think of the middle layer as an **application server**, because it provides the **application logic**, or **business logic**, required by the system. Three-tier designs also are called **n-tier** designs, to indicate that some designs use more than one intermediate layer.

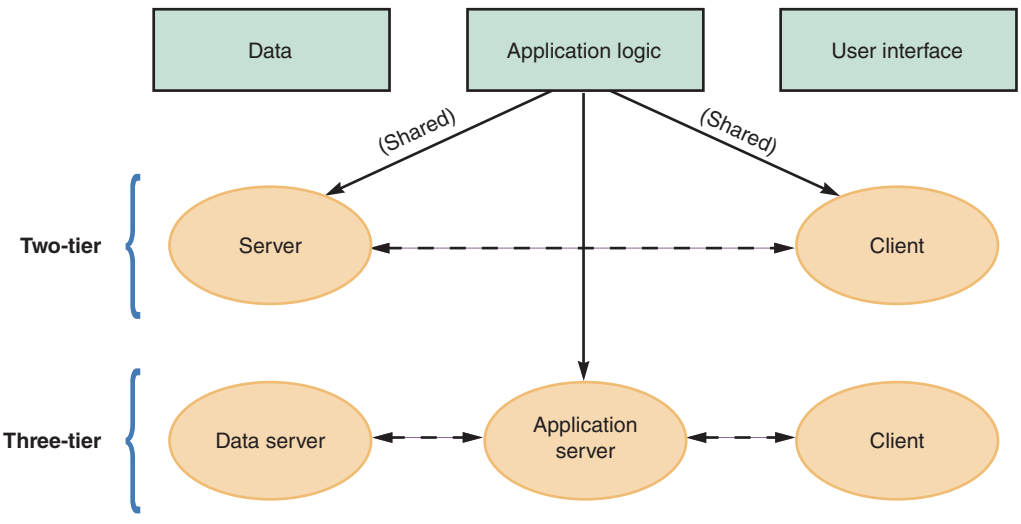


FIGURE 10-16 Characteristics of two-tier versus three-tier client/server design.

The advantage of the application logic layer is that a three-tier design enhances overall performance by reducing

the data server's workload. The separate application logic layer also relieves clients of complex processing tasks. Because it can run on a minicomputer that is much more powerful than the typical client workstations, the middle layer is more efficient and cost-effective in large-scale systems. Figure 10-17 shows where the data, the application logic, and the user interface are located on various architectures. In a client/server system, the tiers communicate using software called middleware, which is described in the following section.

Architecture		Data	Application Logic	User Interface
Central data processing center	Server	X	X	X
	Client			
Central server with remote terminals	Server	X	X	
	Client			X
Stand-alone client	Server			
	Client	X	X	X
Two-tier client/server	Server	X	X	
	Client		X	X
Three-tier client/server	Data server	X		
	Application server		X	
	Client			X

FIGURE 10-17 The location of the data, the application logic, and the user interface depend on the type of architecture.

Middleware

In an n-tier system, special software called **middleware** enables the tiers to communicate and pass data back and forth. Some IT professionals refer to middleware as the glue that holds clients and servers together. The broader definition shown in Figure 10-18 on the next page states that middleware is software that mediates between an application program and a network.

Middleware provides a transparent interface that enables system designers to integrate dissimilar software and hardware. For example, middleware can link a departmental database to a Web server, which can be accessed by client computers via the Internet or a company intranet. Middleware also can integrate legacy systems and Web-based applications. For example, when a user enters a customer number on a Web-based inquiry form, middleware accesses a legacy accounting system and returns the results.

ON THE WEB

To learn more about middleware, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Middleware link.

Cost-Benefit Issues

To support business requirements, information systems need to be scalable, powerful, and flexible. For most companies, client/server systems offer the best combination of features to meet those needs. Whether a business is expanding or downsizing, client/server systems enable the firm to scale the system in a rapidly changing environment. As the size of the business changes, it is easier to adjust the number of clients and the processing functions they perform than it is to alter the capability of a large-scale central server.

Client/server computing also allows companies to transfer applications from expensive mainframes to less-expensive client platforms. In addition, using common languages such as SQL, clients and servers can communicate across multiple platforms. That difference is important because many businesses have substantial investments in a variety of hardware and software environments.

Finally, compared to file server designs, client/server systems reduce network load and improve response times. For example, consider a user at a company headquarters who wants information about total sales figures. In a file server design, the system might need to transmit three separate sales transaction files from three regional offices in order to provide sales data that the client would process; in a client/server system, the server locates the data, performs the necessary processing, and responds immediately to the client's request. The data retrieval and processing functions are transparent to the client because they are done on the server, not the client.

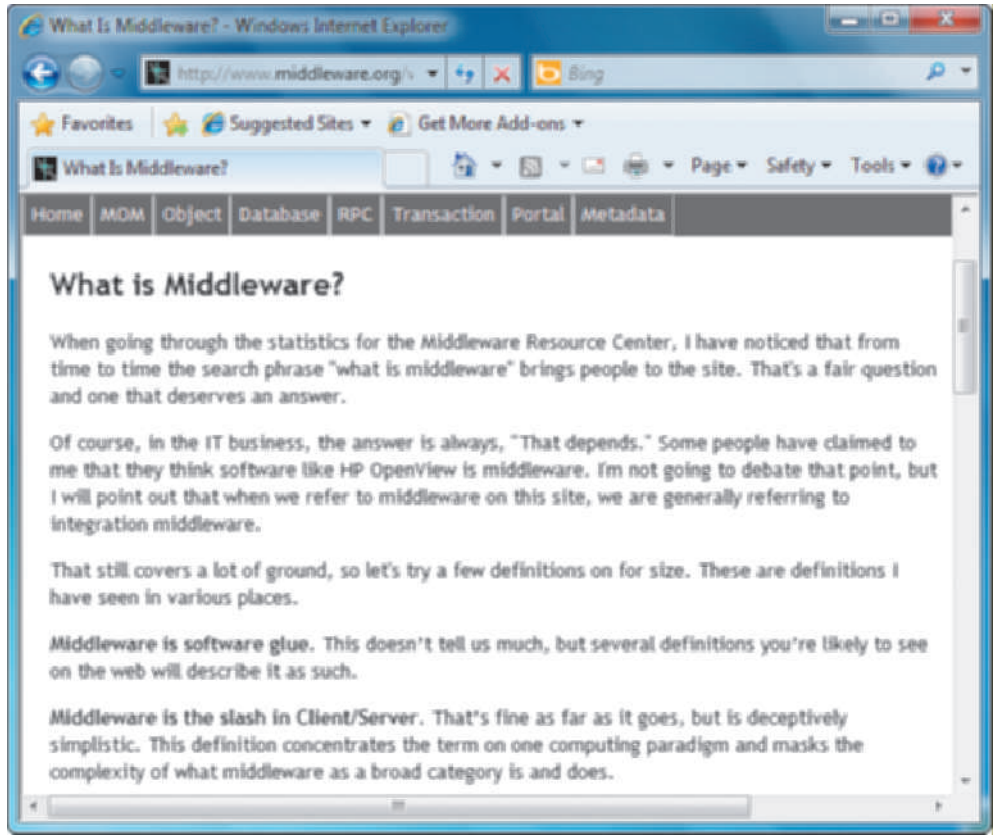


FIGURE 10-18 Middleware connects dissimilar applications and enables them to communicate and exchange data. Middleware also can integrate legacy systems and Web-based applications.

Client/Server Performance Issues

While it provides many important advantages over file-based systems, client/server architecture does involve performance issues that relate to the separation of server-based data and networked clients that must access the data.

Consider the difference between client/server design and a centralized environment, where a server-based program issues a command that is executed by the server's own CPU. Processing speed is enhanced because program instructions and data both travel on an internal system bus, which moves data more efficiently than an external network.

In contrast to the centralized system, a client/server design separates applications and data. Networked clients submit data requests to the server, which responds by sending data back to the clients. When the number of clients and the demand for services increases beyond a certain level, network capacity becomes a constraint, and system performance declines dramatically.

In the article shown in Figure 10-19, IBM states that the performance characteristics of a client/server system are not the same as a centralized processing environment. Client/server response times increase gradually as more requests are made, but then rise dramatically when the system nears its capacity. To deliver and maintain acceptable performance, system developers must anticipate the number of users, network traffic, server size and location, and design a client/server architecture that can support current and future business needs.

What is the answer to enhancing client/server performance? According to IBM, client/server systems must be designed so the client contacts the server only when necessary and makes as few trips as possible.

Another issue that affects client/server performance is data storage. Just as processing can be done at various places, data can be stored in more than one location using a **distributed database management system** (DDBMS).

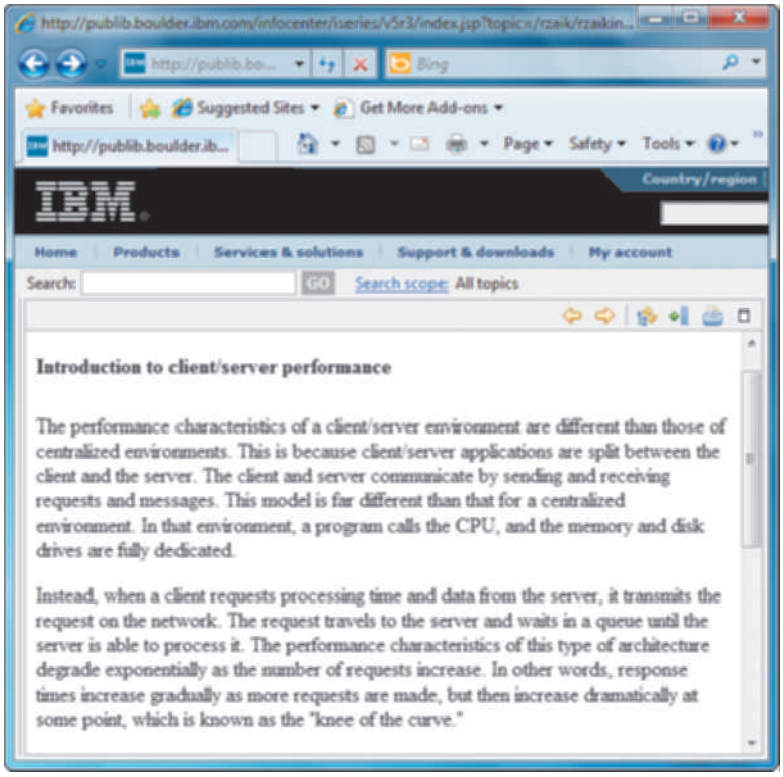


FIGURE 10-19 According to IBM, client/server response times increase gradually, and then rise dramatically when the system nears its capacity. That point is referred to as *the knee of the curve*.

Using a DDBMS offers several advantages: Data stored closer to users can reduce network traffic; the system is scalable, so new data sites can be added without reworking the system design; and with data stored in various locations, the system is less likely to experience a catastrophic failure. A potential disadvantage of distributed data storage involves data security. It can be more difficult to maintain controls and standards when data is stored in various locations. In addition, the architecture of a DDBMS is more complex and difficult to manage. From a system design standpoint, the challenge is that companies often want it both ways — they want the control that comes with centralization *and* the flexibility associated with decentralization.

Internet-Based Architecture

The Internet has had an enormous impact on system architecture. The Internet has become more than a communication channel — many IT observers see it as a fundamentally different environment for system development.

Recall that in a traditional client/server system, the client handles the user interface, as shown in Figure 10-16 on page 464, and the server (or servers in a multi-tier system) handles the data and application logic. In a sense, part of the system runs on the client, part on the server. In contrast, in an Internet-based architecture, in addition to data and application logic, the entire user interface is provided by the Web server in the form of HTML coded documents that are interpreted and displayed by the client's browser. Shifting the responsibility for the interface from the client to the server simplifies the process of data transmission and results in lower hardware costs and complexities.

The trend toward Internet-based e-business is reshaping the IT landscape as more firms use the Web to build efficient, reliable, and cost-effective solutions. When planning new systems, analysts can use available and emerging technology to meet their company's business requirements.

The advantages of Internet-based architecture are changing fundamental ideas about how computer systems should be designed, and many IT experts are shifting their focus to a total online environment. At the same time, large numbers of individual users are seeking Web-based collaboration and social networking services to accomplish tasks that used to be done in person, over the phone, or by more traditional Internet channels. As you learned in Chapter 7, cloud computing and Web 2.0 are important concepts that reflect this online shift.

The following sections examine Web-based architecture, including in-house development, packaged solutions, e-business service providers, corporate portals, cloud computing, and Web 2.0. It is important to be aware of these trends, as they may predict where the IT industry is headed.

Developing E-Commerce Solutions In-House

In Chapter 7, you learned how to analyze advantages and disadvantages of in-house development versus purchasing a software package. The same basic principles apply to system design.

If you decide to proceed with an in-house solution, you must have an overall plan to help achieve your goals. How should you begin? Figure 10-20 offers guidelines for companies developing e-commerce strategies. An in-house solution usually requires a greater initial investment, but provides more flexibility for a company that must adapt quickly in a dynamic e-commerce environment. By working in-house, a company has more freedom to integrate with customers and suppliers and is less dependent on vendor-specific solutions.

Guidelines for In-house E-commerce Site Development
Analyze the company's business needs and develop a clear statement of your goals. Consider the experience of other companies with similar projects.
Obtain input from users who understand the business and technology issues involved in the project. Plan for future growth, but aim for ease of use.
Determine whether the IT staff has the necessary skills and experience to implement the project. Consider training, additional resources, and the use of consultants if necessary.
Consider integration requirements for existing legacy systems or enterprise resource planning. Select a physical infrastructure carefully, so it will support the application, now and later.
Develop the project in modular form so users can test and approve the functional elements as you go along.
Connect the application to existing in-house systems and verify interactivity.
Test every aspect of the site exhaustively. Consider a preliminary rollout to a pilot group to obtain feedback before a full launch.

FIGURE 10-20 Guidelines for companies developing e-commerce strategies.

For smaller companies, the decision about in-house Web development is even more critical, because this approach will require financial resources and management attention that many small companies might be unable or unwilling to commit. An in-house strategy, however, can provide valuable benefits, including the following:

- A unique Web site, with a look and feel consistent with the company's other marketing efforts
- Complete control over the organization of the site, the number of pages, and the size of the files
- A scalable structure to handle increases in sales and product offerings in the future
- More flexibility to modify and manage the site as the company changes
- The opportunity to integrate the firm's Web-based business systems with its other information systems, creating the potential for more savings and better customer service

Whether a firm uses an in-house or a packaged design, the decision about Web hosting is a separate issue. Although internal hosting has some advantages, such as greater control and security, the expense would be much greater, especially for a small- to medium-sized firm.

CASE IN POINT 10.2: SMALL POTATOES, INC.

Small Potatoes is a family-operated seed business that has grown rapidly. Small Potatoes specializes in supplying home gardeners with the finest seeds and gardening supplies. Until now, the firm has done all its business by placing ads in gardening and health magazines, and taking orders using a toll-free telephone number.

Now, the family has decided to establish a Web site and sell online, but there is some disagreement about the best way to proceed. Some say it would be better to develop the site on their own, and Betty Lou Jones, a recent computer science graduate, believes she can handle the task. Others, including Sam Jones, Betty's grandfather, feel it would be better to outsource the site and focus on the business itself. Suppose the family asked for your opinion. What would you say? What additional questions would you ask?

Packaged Solutions and E-Commerce Service Providers

If a small company is reluctant to take on the challenge and complexity of developing an Internet commerce site in-house, an alternative can be a packaged solution or an e-commerce service provider. This is true even for medium- to large-sized firms. Many vendors, including Microsoft and Intershop, offer turnkey systems for companies that want to get an e-business up and running quickly, as shown in Figure 10-21.

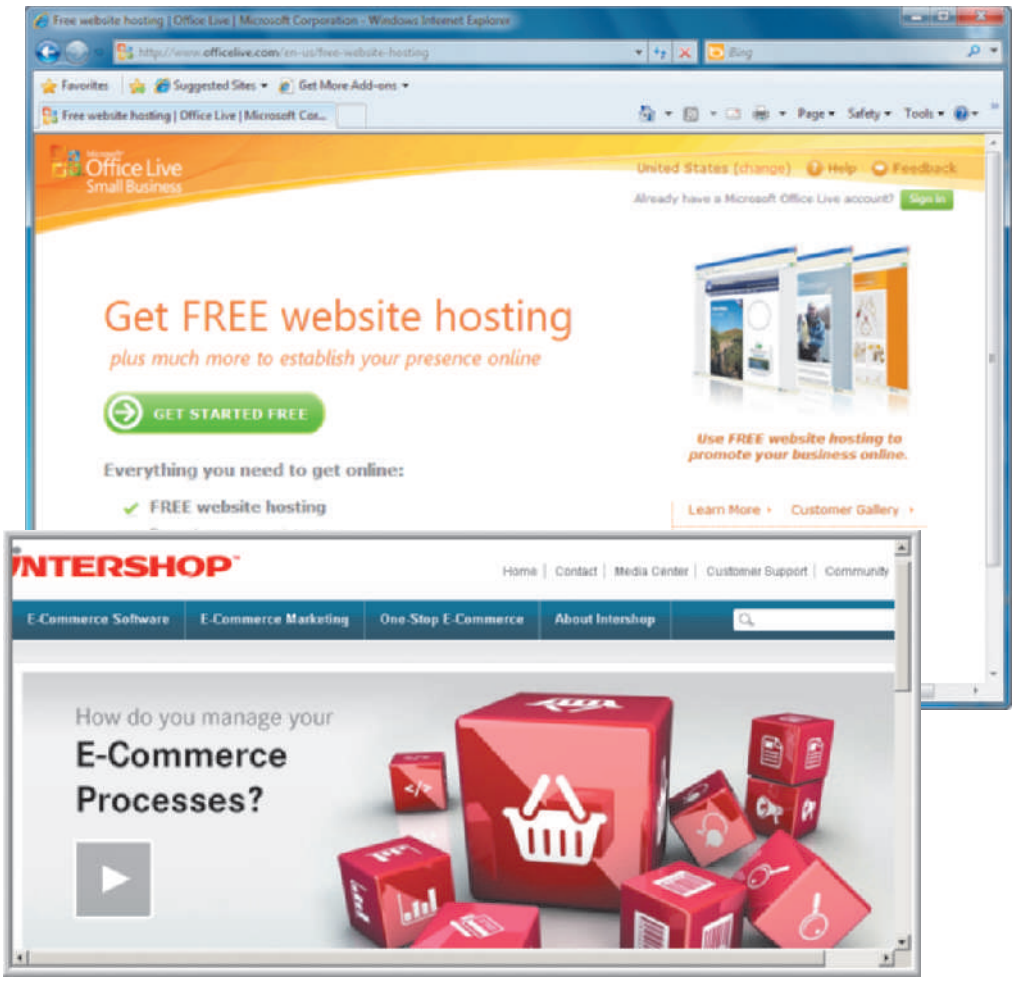


FIGURE 10-21 Microsoft and Intershop offer software solutions for companies that want to get an e-business up and running quickly.

For large-scale systems that must integrate with existing applications, packaged solutions might be less attractive. Another alternative is to use an application service provider (ASP). As explained in Chapter 7, an ASP provides applications, or access to applications, by charging a usage or subscription fee. Today, many ASPs offer full-scale Internet business services for companies that decide to outsource those functions.

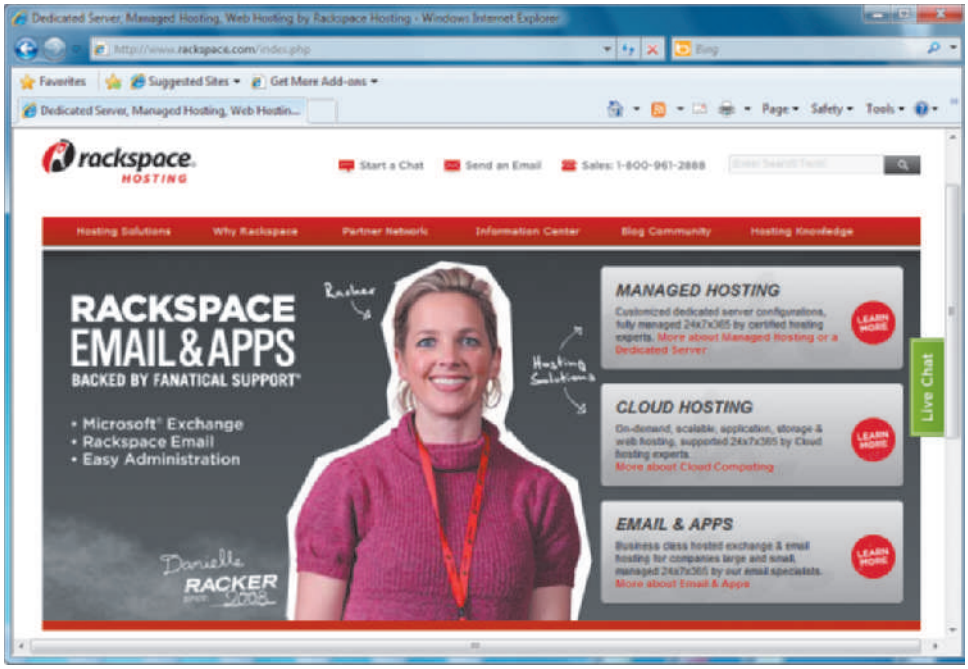


FIGURE 10-22 Rackspace offers managed hosting, cloud hosting, and e-mail services.

Another option is managed hosting, which also was discussed in Chapter 7. As shown in Figure 10-22, a solution provider such as Rackspace can host and maintain a corporate Web site. Rackspace states that its customers will "never have to implement, update, troubleshoot, patch, monitor, administer, backup data, or worry again."

A systems analyst confronts a bewildering array of products and strategies when implementing Internet-based systems. A good starting point might be to consider the experience of other companies in the same industry. Many firms, including Sybase, offer success stories and case studies of successful systems development, as shown in Figure 10-23. Although each situation is different, this type of research can provide valuable information about a vendor's products and services.

FIGURE 10-23 Success stories and case studies can provide valuable information about a vendor's products and services.

Corporate Portals

A **portal** is an entrance to a multifunction Web site. After entering a portal, a user can navigate to a destination using various tools and features provided by the portal designer. A corporate portal can provide access for customers, employees, suppliers, and the public. In a Web-based system, portal design provides an important link between the user and the system, and poor design can weaken system effectiveness and value. Figure 10-24 shows enterprise portals offered by HP and SAP. As partners, the firms use their global IT experience and skills to design and implement portal solutions.

Cloud Computing

Cloud computing refers to the cloud symbol that often is used to represent the Internet. The cloud computing concept envisions a *cloud* of remote computers that provide a total online

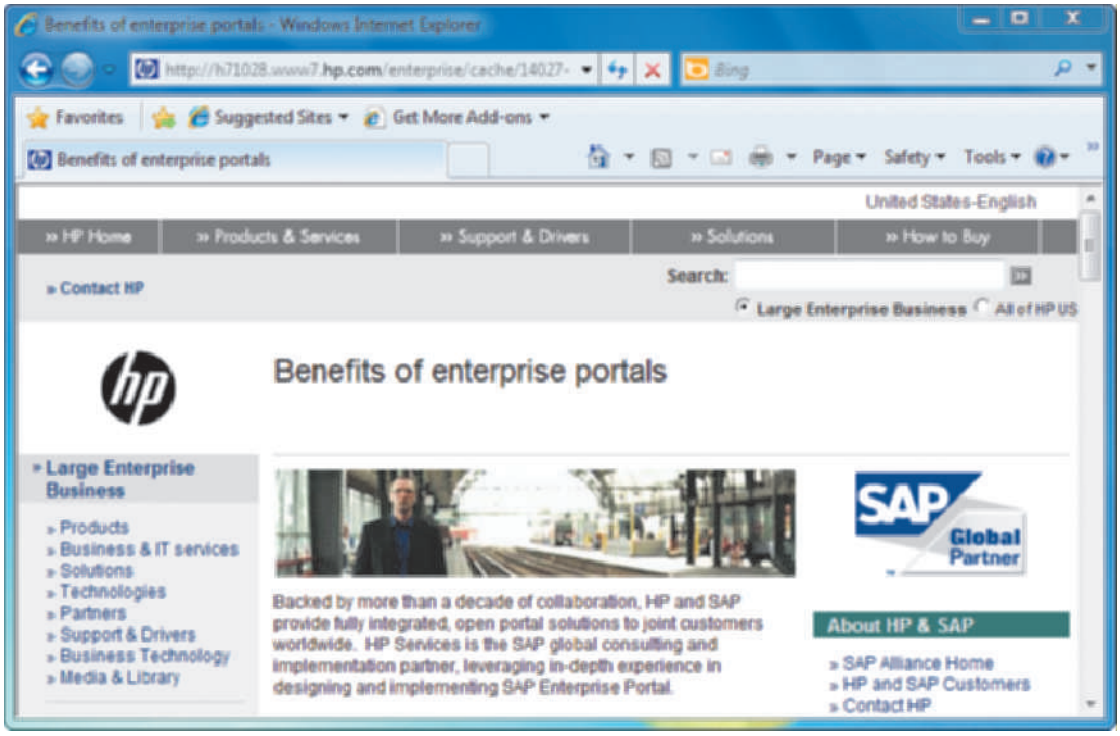


FIGURE 10-24 HP and SAP offer enterprise portal solutions to their customers worldwide.

software and data environment that is hosted by third parties. A user's computer does not perform processing or computing tasks — the cloud does. This concept is in contrast to today's computing model, which is based on networks that strategically distribute processing and data across the enterprise. In a sense, the cloud of computers acts as one giant computer that performs tasks for users. As shown in Figure 10-25, a user logs into a local computer and is connected to the cloud, which performs the computing work. Instead of requiring specific hardware and software on the user's computer, cloud computing spreads the workload to powerful remote systems that are part of the cloud. The user appears to be working on a local system, but all computing is actually performed in the cloud. No updates or maintenance are required of the user, and there are no compatibility issues.

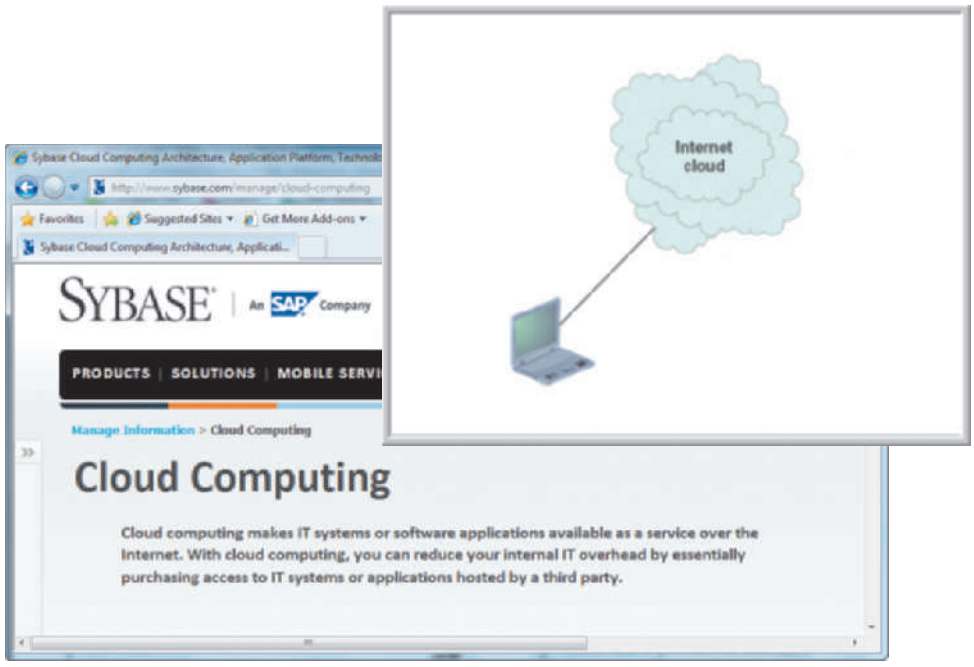


FIGURE 10-25 In cloud computing, users connect to the Internet cloud to access personal content and services through an online software and data environment.

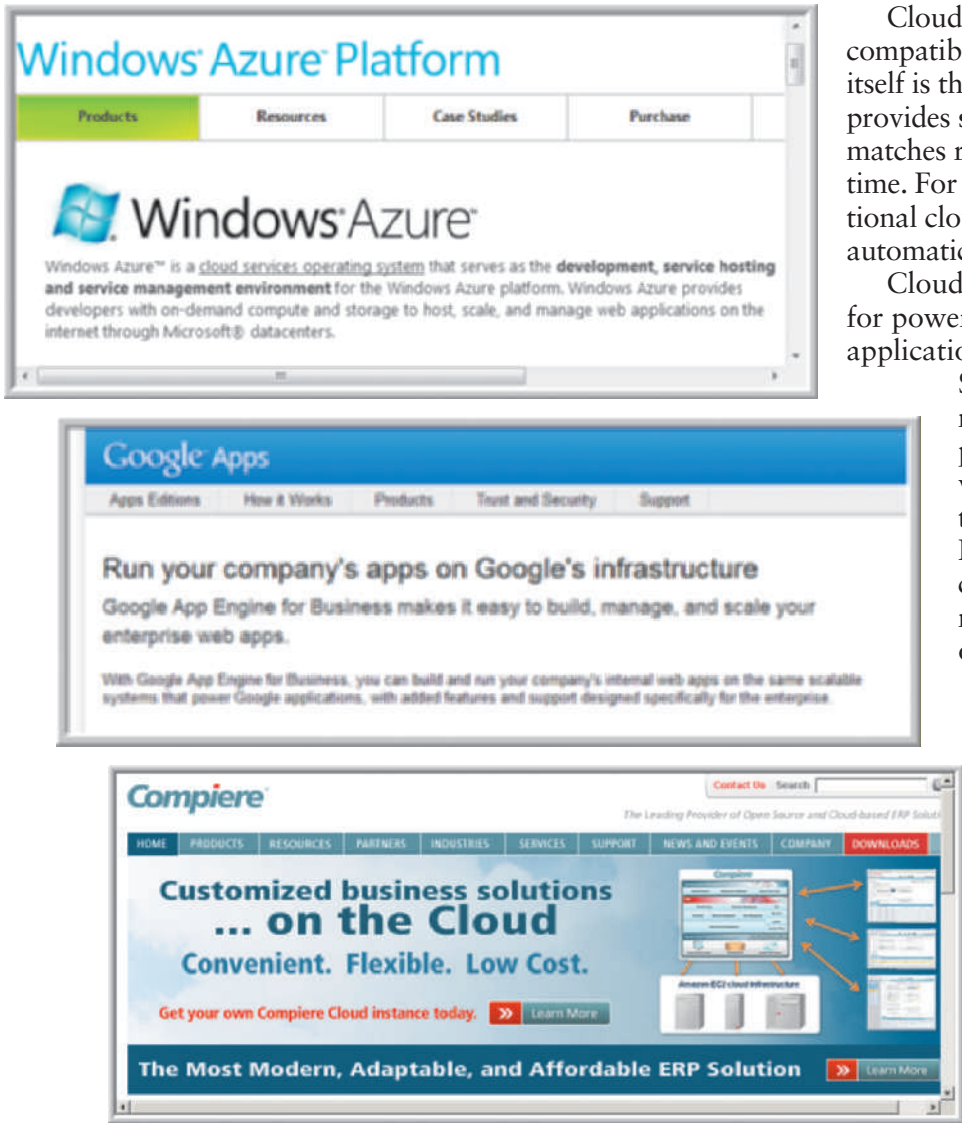


FIGURE 10-26 Three popular examples of cloud computing platforms.

Cloud computing effectively eliminates compatibility issues, because the Internet itself is the platform. This architecture also provides **scaling on demand**, which matches resources to needs at any given time. For example, during peak loads, additional cloud servers might come on line automatically to support the workload.

Cloud computing is an ideal platform for powerful Software as a Service (SaaS) applications. As you learned in Chapter 7, SaaS is a popular deployment method where software is not purchased but is paid for as a service, much like one pays for electricity or cable TV each month. In this architecture, updates and changes to services can be easily made by service providers without involving the users.

Even though cloud computing has many advantages, some concerns exist. First, cloud computing requires significantly more **bandwidth** (the amount of data that can be transferred in a fixed time period) than today's networks. Second, because cloud computing is Internet-based, if a user's Internet connection becomes unavailable, he or she will be unable to access any cloud-based services. In addition, there are security concerns associated with sending large amounts of data over the Internet, as well as concerns about storing it securely. Finally, there is the issue of control. Because a service provider hosts the resources and manages data storage and access, the provider has complete control of the system. Many firms are wary of handing over control of mission-critical data and systems to a third-party provider.

It remains to be seen whether cloud computing's advantages will outweigh its disadvantages. Technology advances continue to make cloud computing more feasible, desirable, and secure. As the IT industry moves toward Internet-based architectures, cloud computing's success will depend on how bandwidth, reliability, and security are addressed and how well cloud computing is received by users. As shown in Figure 10-26, examples of popular cloud computing platforms include Windows Azure, Amazon's Elastic Compute Cloud, and Compiere.

Web 2.0

The shift to Internet-based collaboration has been so powerful and compelling that it has been named **Web 2.0**. Web 2.0 is not a reference to a more technically advanced version of the current Web. Rather, Web 2.0 envisions a second generation of the Web that will enable people to collaborate, interact, and share information more dynamically.

Leading Web 2.0 author Tim O'Reilly has suggested that the strong interest in Web 2.0 is driven by the concept of the *Internet as a platform*. O'Reilly sees future Web 2.0 applications delivering software as a continuous service with no limitations on the number of users that can connect or how users can consume, modify, and exchange data.

Figure 10-27 shows examples of popular social networking sites, which are seeing explosive growth in the Web 2.0 environment. Another form of social collaboration is called a wiki. A **wiki** is a Web-based repository of information that anyone can access, contribute to, or modify. In a sense, a wiki represents the collective knowledge of a group of people. One of the best-known wikis is Wikipedia.org, but smaller-scale wikis are growing rapidly at businesses, schools, and other organizations that want to compile and share information.

One of the goals of Web 2.0 is to enhance creativity, interaction, and shared ideas. In this regard, the Web 2.0 concept resembles the agile development process and the open-source software movement. Web 2.0 communities and services are based on a body of data created by users. As users collaborate, new layers of information are added in an overall environment known as the **Internet operating system**. These layers can contain text, sound bytes, images, and video clips that are shared with the user community.

FIGURE 10-27 Facebook, MySpace, and Twitter are popular examples of Web 2.0 social networking.

Processing Methods

In selecting an architecture, the systems analyst must determine whether the system will be an online system, a batch processing system, or a combination of the two.

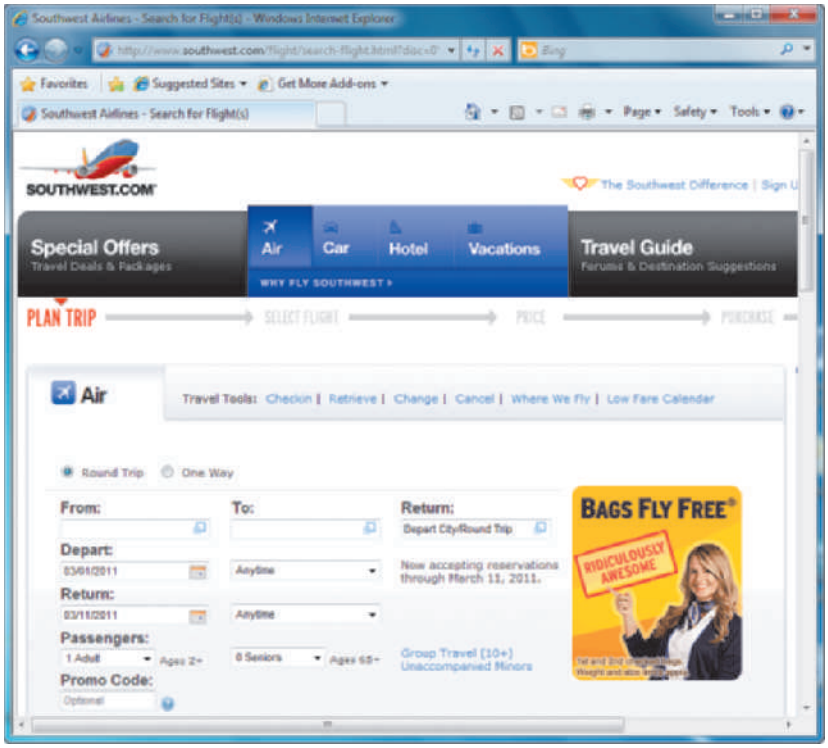


FIGURE 10-28 The Southwest Airlines reservation system is an example of Web-based online processing.

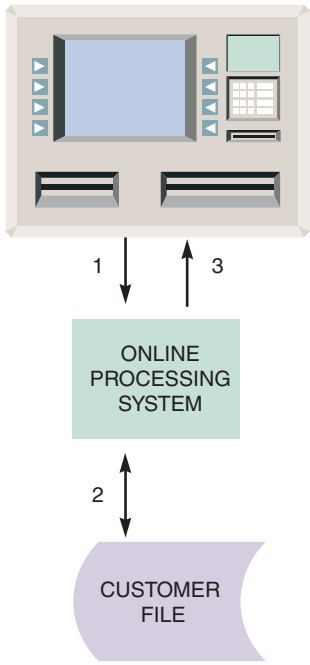
Online Processing

Early computer systems relied mainly on batch processing, but the vast majority of systems today use online processing. An **online system** handles transactions when and where they occur and provides output directly to users. Because it is interactive, online processing avoids delays and allows a constant dialog between the user and the system.

An airline reservations system is a familiar example of online processing. When an online customer views the screen shown in Figure 10-28, he or she can enter the origin, destination, travel dates, and travel times. The system searches a database and responds by displaying available flights, times, and prices. The customer can make a reservation, enter a name, address, credit card information, and other required data and the system creates the reservation, assigns a seat, and updates the flight database immediately.

Online processing also can be used with file-oriented systems. Figure 10-29 shows what happens when a customer uses an ATM to inquire about an account balance. After the ATM verifies the customer's card and password, the customer enters the request (Step 1). Then, the system accesses the account master file using the account number as the primary key and retrieves the customer's record (Step 2). The system verifies the account number and displays the balance (Step 3). Data is retrieved and the system transmits the current balance to the ATM, which prints it for the customer. Online processing systems have four typical characteristics:

1. The system processes transactions completely when and where they occur.
2. Users interact directly with the information system.
3. Users can access data randomly.
4. The information system must be available whenever necessary to support business functions.



ATM QUERY PROCESS

Step 1: Customer enters his or her account number and requests an account balance

Step 2: Retrieves current account balance

Step 3: Verifies bank account number and displays balance on ATM screen

FIGURE 10-29 When a customer requests a balance, the ATM system verifies the account number, submits the query, retrieves the current balance, and displays the balance on the ATM screen.

Batch Processing

In a **batch processing** system, data is collected and processed in groups, or batches. Although online processing is used for interactive business systems that require immediate data input and output, batch processing can handle other situations more efficiently. For example, batch processing typically is used for large amounts of data that must be processed on a routine schedule, such as paychecks or credit card transactions.

In batch processing, input transactions are grouped into a single file and processed together. For example, when a firm produces customer statements at the end of the month, a batch application might process many thousands of records in one run of the program. A batch processing system has several main characteristics: collect, group, and process transactions periodically; the IT operations group can run batch programs on a predetermined schedule, without user involvement, during regular business hours, at night, or on weekends; and batch programs require significantly fewer network resources than online systems.

CASE IN POINT 10.3: R/WAY TRUCKING COMPANY

You are the new IT manager at R/Way, a small but rapidly growing trucking company headquartered in Cleveland, Ohio. The company slogan is "Ship It R/Way — State-of-the-Art in Trucking and Customer Service." R/Way's information system currently consists of a file server and three workstations where freight clerks enter data, track shipments, and prepare freight bills. To perform their work, the clerks obtain data from the server and use database and spreadsheet programs stored on their PCs to process the data.

Unfortunately, your predecessor did not design a relational database. Instead, data is stored in several files, including one for shippers, one for customers, and one for shipments. The system worked well for several years, but cannot handle current volume or support online links for R/Way shippers and customers. The company president is willing to make changes, but he is reluctant to spend money on major IT improvements unless you can convince him that they are necessary.

What would you recommend and why?

Combined Online and Batch Processing

Even an online system can use batch processing to perform certain routine tasks. Online processing also can be used with file-oriented systems. Figure 10-30 shows a familiar **point-of-sale** (**POS**) terminal, and Figure 10-31 on the next page shows how a retail chain uses POS terminals to drive online and batch processing methods. Notice that the system uses online processing to handle data entry and inventory updates, while reports and accounting entries are performed in a batch.

FIGURE 10-30 Retail point-of-sale terminals provide customer sales support and transaction processing capability.

The retail store system illustrates both online processing and batch processing of data. During business hours, the salesperson enters a sale on a POS terminal, which is part of an information system that handles daily sales transactions and maintains the online inventory file. When the salesperson enters the transaction, online processing occurs. The system performs calculations, updates the inventory file, and produces output on the POS terminal in the

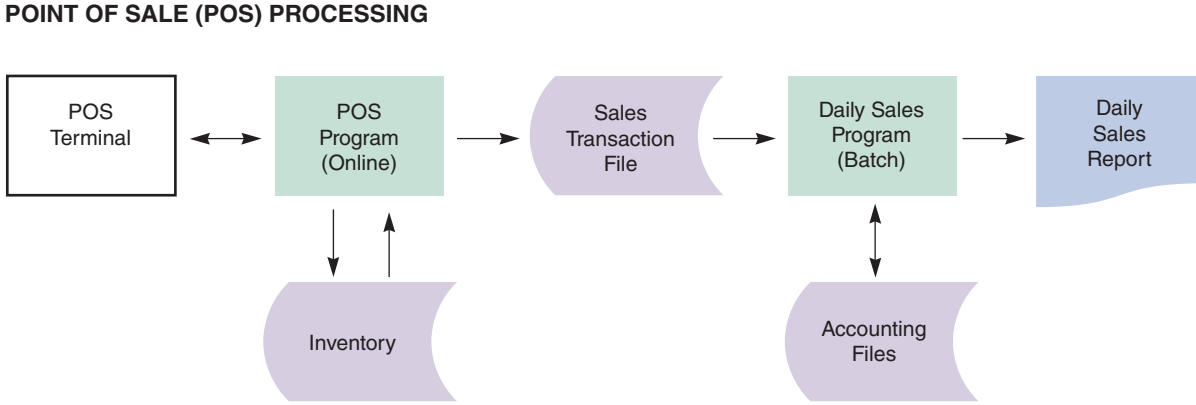


FIGURE 10-31 Many retailers use a combination of online and batch processing. When a salesperson enters the sale on the POS terminal, the online system retrieves data from the item file, updates the quantity in stock, and produces a sales transaction record. At the end of the day, a batch processing program produces a daily sales report and updates the accounting system.

form of a screen display and a printed receipt. At the same time, each sales transaction creates input data for day-end batch processing.

When the store closes, the system uses the sales transactions to produce the daily sales report and related accounting entries using batch processing. Performing the processing online before all sales transactions are completed does not make sense. In that situation, a batch method provides better routine transaction processing, while an online approach supports point-of-sale processing, which must be done as it occurs.

In the retail store example, both online and batch processing are integral parts of the information system. Online processing offers an inherent advantage because data is entered and validated as it occurs, so the stored data is available sooner and always is up to date. Online processing is more expensive, however, and the effect of computer system downtime or slowdown while transactions are processed causes far more disruption than in batch processing. In addition, backup and recovery for online processing are more difficult. In many situations, batch processing is cost-effective, less vulnerable to system disruption, and less intrusive to normal operations. Many information systems will continue to use a combination of online and batch processing for some time to come.

Network Models

A network allows the sharing of hardware, software, and data resources in order to reduce expenses and provide more capability to users. When planning a network design, you must consider network terms and concepts, including the OSI model, network modeling tools, network topology, network protocols, licensing issues, and wireless networks, which are covered in this section. Other important issues, such as network performance and security, are covered in Chapter 12, Managing Systems Support and Security.

ON THE WEB

To learn more about the OSI reference model, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the OSI Reference Model link.

The OSI Reference Model

Based on the discussion of system architecture earlier in this chapter, you already understand basic network terms such as client, server, LAN, WAN, file server design, client/server architecture, tiers, and middleware.

Before you study network topology, you should be familiar with the **OSI (Open Systems Interconnection) model**, which describes how data actually moves from an application on one computer to an application on another networked computer. The OSI model consists of seven layers. Each layer performs a specific function, as shown in Figure 10-32.

LAYER NUMBER	NAME	DESCRIPTION
7	Application layer	Provides network services requested by a local workstation
6	Presentation layer	Ensures that data is uniformly structured and formatted for network transmission
5	Session layer	Defines control structures that manage the communications link between computers
4	Transport layer	Provides reliable data flow and error recovery
3	Network layer	Defines network addresses and determines how data packets are routed over the network
2	Data link layer	Defines specific methods of transmitting data over the physical layer, such as defining the start and end of a data frame
1	Physical layer	Contains physical components that carry data, such as cabling and connectors

FIGURE 10-32 In the OSI model, data proceeds down through the layers on the transmitting computer, then up through the layers on the receiving computer. Along the way, data may pass through one or more network routers that control the path from one network address to another.

It is important to understand that OSI is a conceptual model and is not tied to any specific physical environment or hardware. However, the OSI model does provide design standards that assure seamless interchange and connectivity for network hardware and software.

Network Protocols

In all cases, the network must use a **protocol**, which is a set of standards that govern network data transmission. A popular network protocol is **Transmission Control Protocol/Internet Protocol (TCP/IP)**. Originally developed by the U.S. Department of Defense to permit interconnection of military computers, today TCP/IP is the backbone of the Internet. Other older network protocols include NetBIOS, which was popular for LANs, and IPX, which is a protocol used by Novell Corporation for older NetWare products.

TCP/IP actually consists of many individual protocols that control the handling of files, mail, and Internet addresses, among others. A familiar example of a TCP/IP protocol is the **File Transfer Protocol (FTP)**, which provides a reliable means of copying files from one computer to another over a TCP/IP network, such as the Internet or an intranet.

ON THE WEB

To learn more about network protocols, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Network Protocols link.

Network Topology

The way a network is configured is called the **network topology**. Topology can refer to a physical or a logical view of the network. For example, **physical topology** describes the actual network cabling and connections, while **logical topology** describes the way the components interact. It is important to understand the distinction, because a specific physical topology might be able to support more than one logical topology. For example, it is not uncommon to run cabling in a certain pattern because of physical installation and cost issues, but to use a different pattern for the logical topology.

The workstations in Figure 10-33 on the next page are arranged in a circular shape, but that might or might not reflect the network topology. The examples shown in Figures 10-34 to 10-38 on pages 478 to 481 represent a logical topology, as seen by network users, who do not know or care about the physical cabling pattern.

FIGURE 10-33 Although these workstations form a circle physically, the layout has no bearing on the network topology, which might be a bus, ring, star, or other logical design.

LAN and WAN networks typically are arranged in four patterns: hierarchical, bus, ring, and star. The concepts are the same regardless of the size of the network, but the physical implementation is different for a large-scale WAN that spans an entire business enterprise compared with a small LAN in a single department. The four topologies are described in the following sections.

HIERARCHICAL NETWORK In a **hierarchical network**, as shown in Figure 10-34, one or more powerful servers control the entire network. Departmental servers control lower levels of processing and network devices. An example of a hierarchical network might be a retail clothing chain, with a central computer that stores data about sales activity and inventory levels and local computers that handle store-level operations.

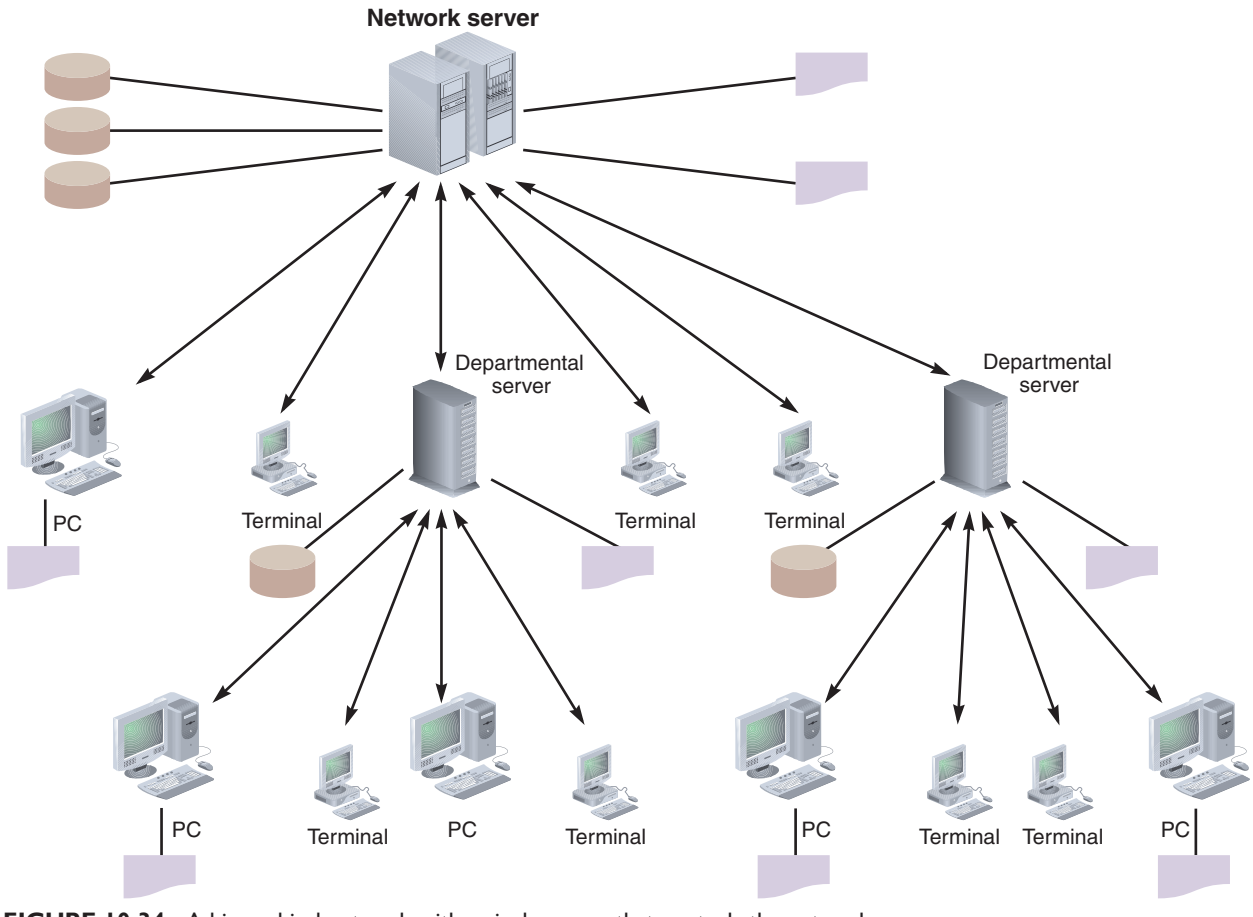


FIGURE 10-34 A hierarchical network with a single server that controls the network.

The stores transmit data to the central computer, which analyzes sales trends, determines optimum stock levels, and coordinates a supply chain management system. In this situation, a hierarchical network might be used, because it mirrors the actual operational flow in the organization.

One disadvantage of a hierarchical network is that if a business adds additional processing levels, the network becomes more complex and expensive to operate and maintain. Hierarchical networks were often used in traditional mainframe-based systems, but are much less common today.

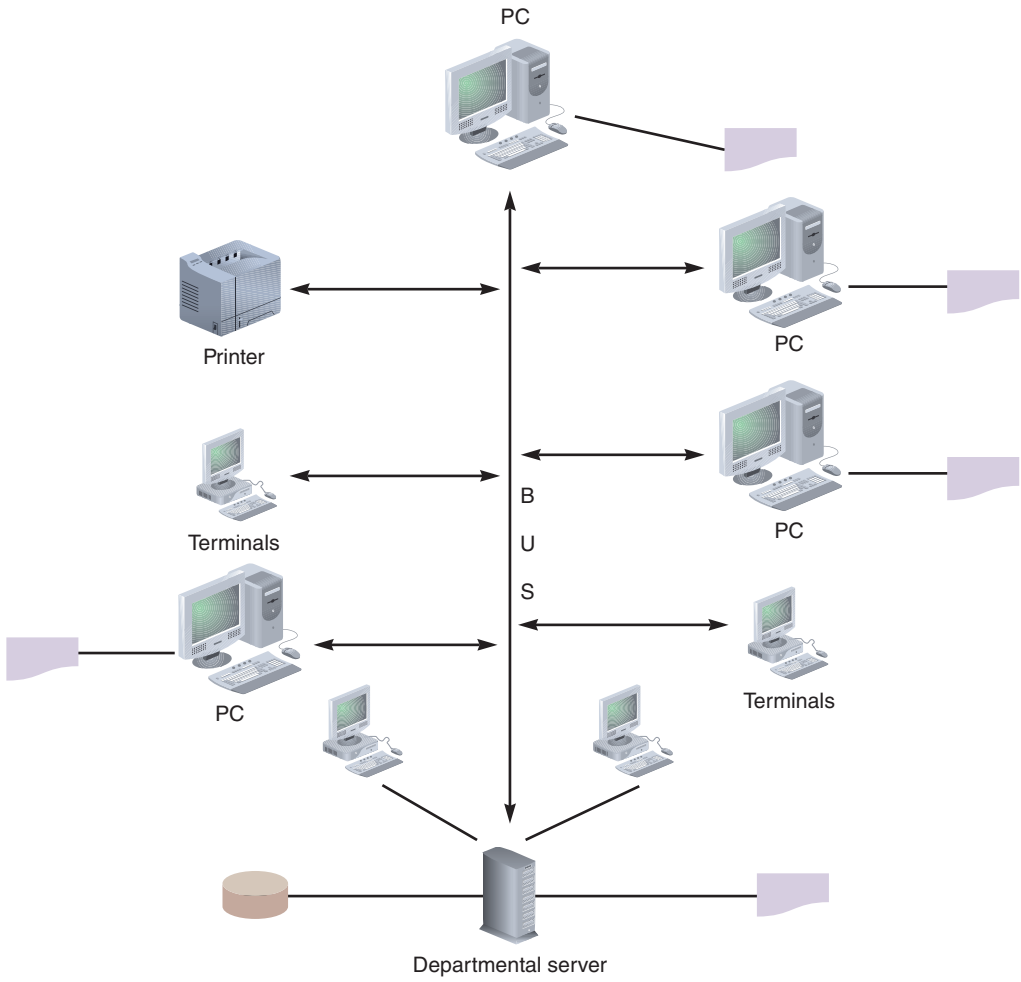


FIGURE 10-35 A bus network with all devices connected to a single communication path.

BUS NETWORK In a **bus network**, as shown in Figure 10-35, a single communication path connects the central server, departmental servers, workstations, and peripheral devices. Information is transmitted in either direction between networked devices, and all messages travel over the same central bus. Bus networks require less cabling than other topologies, because only a single cable is used. Devices can also be attached or detached from the network at any point without disturbing the rest of the network. In addition, a failure in one workstation on the network does not necessarily affect other workstations on the network.

One major disadvantage of a bus network is that if the central bus becomes damaged or defective, the entire network shuts down. Another disadvantage is that overall performance declines as more users and devices are added, because all message traffic must flow along the central bus. This does not occur in the treelike structure of a hierarchical network or the hub-and-spoke design of a star network, where network paths are more isolated and independent.

The bus network is one of the oldest LAN topologies, and is a simple way to connect multiple workstations. Before the proliferation of star networks, bus networks were very common. Today, the bus design is much less popular, but some firms have retained bus networks to avoid the expense of new wiring and hardware.

RING NETWORK A **ring network**, as shown in Figure 10-36 on the next page, resembles a circle where the data flows in only one direction from one device to the next. In function, a ring network can be thought of as a bus network with the ends connected. One disadvantage of a ring network is that if a network device (such as a PC or a

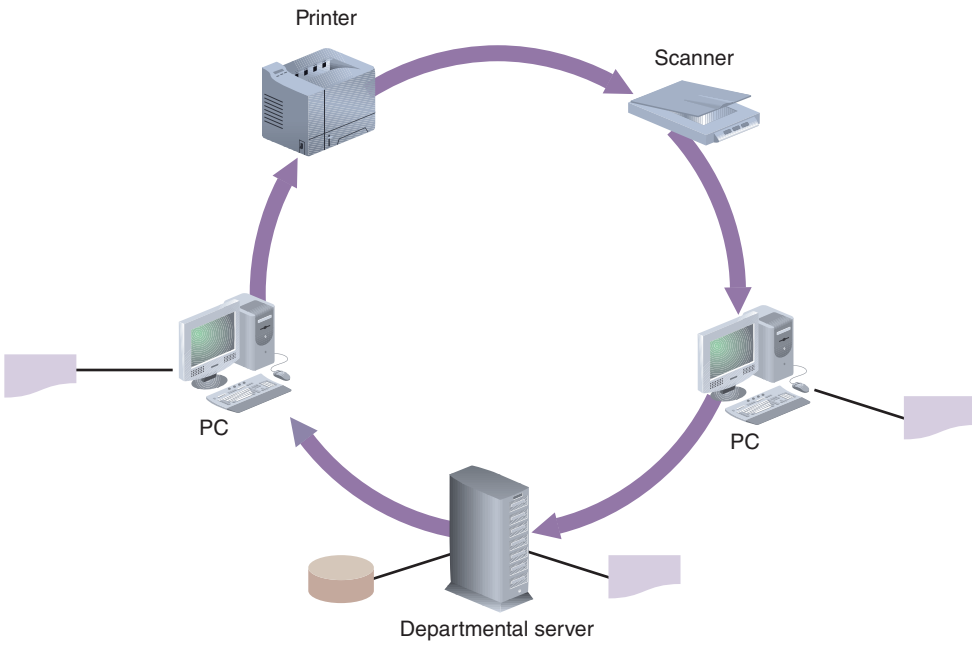


FIGURE 10-36 A ring network with a set of computers that send and receive data flowing in one direction.

server) fails, the devices downstream from the failed device cannot communicate with the network. Although ring networks are less common than other topologies, they sometimes are used to tie local processing sites together. For example, workstations and servers in the accounting, sales, and shipping departments might perform local processing and then use a ring network to exchange data with other divisions within the company.

It is interesting to note that in a ring network implementation, the physical wiring can resemble a star pattern, using a central device called a **Multistation Access Unit** (MAU). This unit internally wires the workstations into a logical ring, and manages the flow of data from one device to the next.

STAR NETWORK Because of its speed and versatility, the star network is by far the most popular LAN topology today. A **star network** has a central networking device called a **switch**, which manages the network and acts as a communications conduit for all network traffic. In the past, a device known as a **hub** was used to connect star networks, but a switch offers advanced technology and much better performance. A hub functions like a familiar multi-socket power strip, but with network devices such as servers, workstations, and printers plugged in rather than electrical appliances. The hub broadcasts network traffic, called **data frames**, to all connected devices. In contrast, a switch enhances network performance by sending traffic only to specific network devices that need to receive the data.

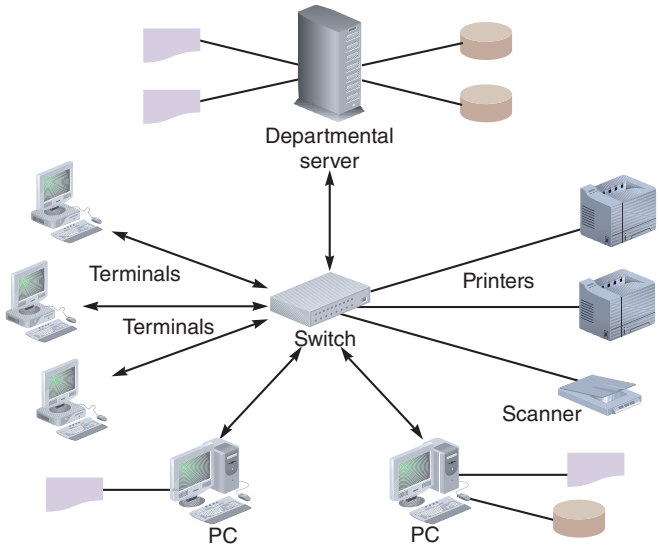


FIGURE 10-37 A typical star network with a switch, departmental server, and connected workstations.

A star configuration, as shown in Figure 10-37, provides a high degree of network control, because all traffic flows into and out of the switch. An inherent disadvantage of the star design is that the entire network is dependent on the switch. However, in most large star networks, backup switches are available immediately in case of hardware failure.

MESH NETWORK In the **mesh network** shown in Figure 10-38, each node connects to every other node. While this design is extremely reliable, it also is very expensive to install and maintain. A mesh network resembles the Internet in that a message can travel on more than one path. Originally developed for military applications, the primary advantage of a mesh network is redundancy, because multiple paths provide backup if communication problems arise or some nodes become inoperable.

Routers

Networks such as LANs or WANs can be interconnected using devices called routers. A **router** is a device that connects network segments, determines the most efficient data path, and guides the flow of data. Routers differ from switches in that they work at a higher OSI level (layer 3), dealing with IP packets, while switches handle data frames (layer 2).

Using a router, any network topology can connect to a larger, dissimilar network, such as the Internet. This connection is called a **gateway**. The example in Figure 10-39 shows a star topology, where the router links the network to the Internet. A device called a **proxy server** provides Internet connectivity for internal LAN users. The vast majority of business networks use routers to integrate the overall network architecture.

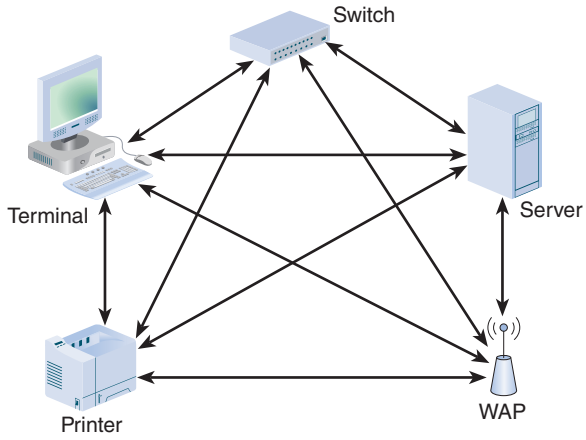


FIGURE 10-38 Mesh networks are used in situations where a high degree of redundancy is needed, such as military applications. The redundant design provides alternate data paths, but is expensive to install and maintain.

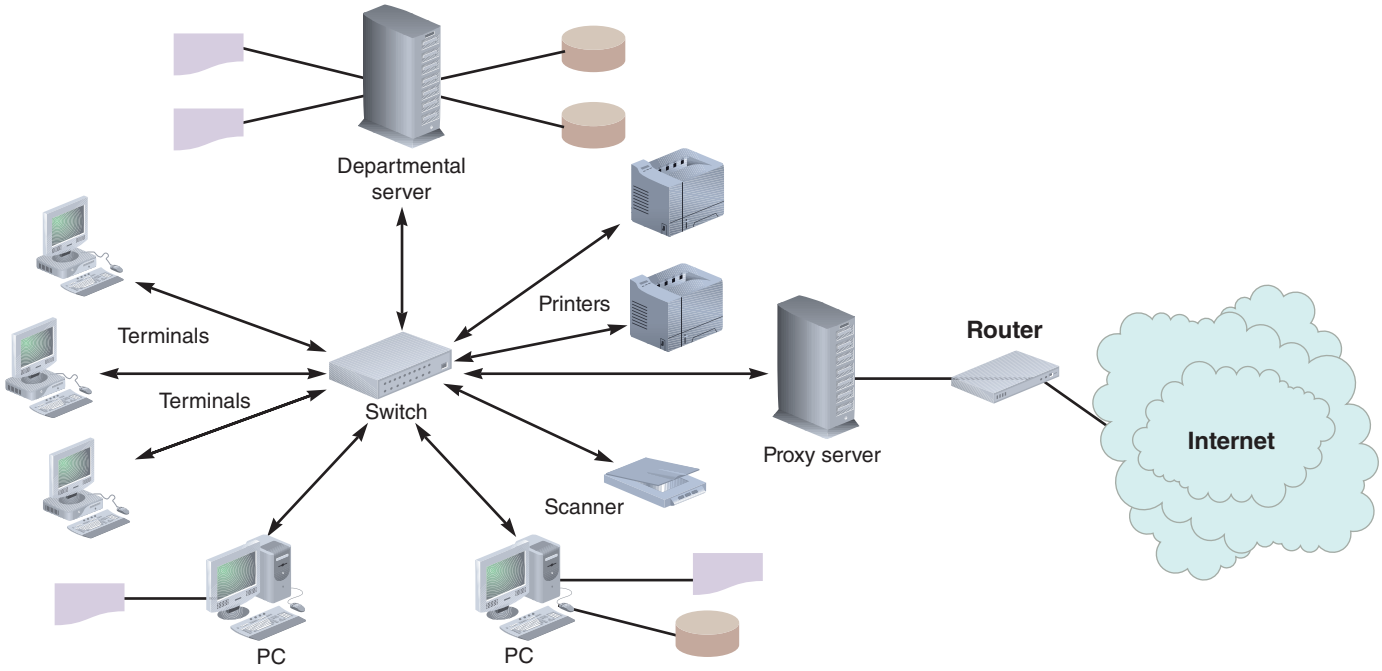


FIGURE 10-39 Routers can be used to connect LANs and WANs to other networks, such as the Internet.

Network Modeling Tools

The OSI's multilayer concept makes it easier to understand and work with the individual components. As you translate the OSI conceptual model into a physical version of the system, you can use software tools, such as Microsoft Visio, which is a multipurpose drawing tool, to represent the physical structure and network components. Visio offers a wide variety of drawing types, styles, templates, and shapes. For example, Visio supplies templates for basic network designs, plus manufacturer-specific symbols for firms such as Cisco, IBM, Bay Systems, and Hewlett-Packard, among others.

Visio is an example of a CASE tool that can help you plan, analyze, design, and implement an information system. Visio can be used to create a simple network model, either by using drag-and-drop shapes displayed on the left of the screen, or by using provided wizards to walk through a step-by-step network design process. Figure 10-40 on the next page shows a simple network model created using the drag-and-drop feature.

TOOLKIT TIME

The CASE Tools in Part B of the Systems Analyst's Toolkit can help you document business functions and processes, develop graphical models, and provide an overall framework for information system development. To learn more about these tools, turn to Part B of the four-part Toolkit that follows Chapter 12.

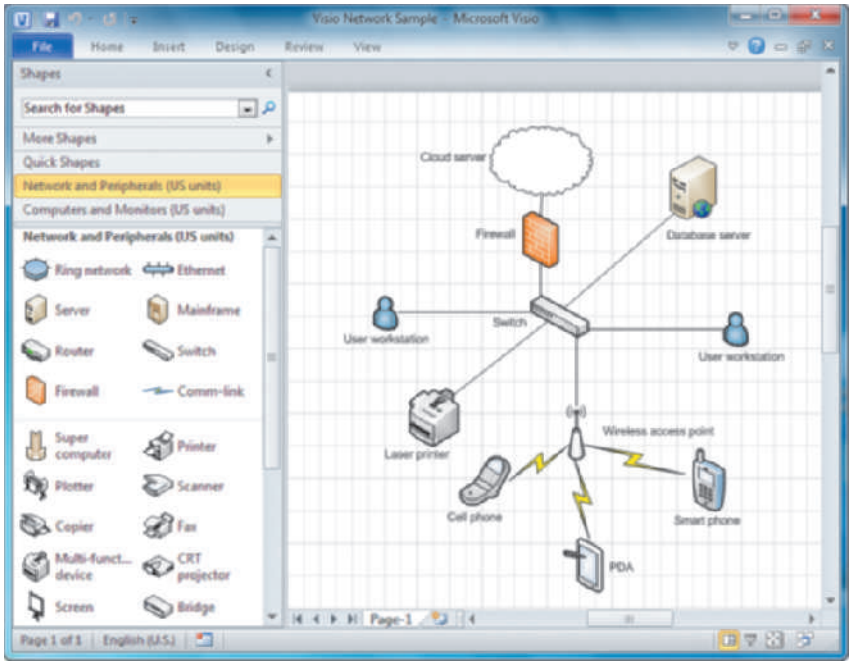


FIGURE 10-40 Microsoft Visio can be used to create a network drawing using drag-and-drop shapes displayed in the panel at the left of the screen.

Network Licensing Issues

When considering a network design, it is important to take into account software licensing restrictions. Various types of individual and site licenses are available from software vendors. Some vendors limit the number of users or the number of computers that can access the program simultaneously. You also must carefully investigate the capabilities of network software to ensure that it can handle the anticipated system traffic.

Wireless Networks

Although a LAN provides enormous flexibility, the initial cabling cost can be substantial, as well as the inevitable wiring changes that occur in a dynamic organization. Many companies find wireless technology to be an attractive alternative. A **wireless local area network**, or **WLAN**, is relatively inexpensive to install and is well-suited to workgroups and users who are not anchored to a specific desk or location. Most notebook computers are equipped with built-in wireless capability, and it is relatively simple to add this feature to existing desktop computers and workstations in order to set up a wireless network.

Like their wired counterparts, wireless networks have certain standards and topologies, which are discussed in the following sections.

Wireless Network Standards

Wireless networks are based on various standards and protocols that still are evolving. The most popular of these is called IEEE **802.11**, which is a family of standards developed by the **Institute of Electrical and Electronics Engineers** (**IEEE**) for wireless LANs.

Current wireless networks are based on variations of the original 802.11 standard. Several versions, or **amendments**, were intended to improve bandwidth, range, and security. The table in Figure 10-41 contains a brief comparison of the IEEE 802.11 amendments. Note that maximum speed is measured in **Mbps** (**megabits per second**).

STANDARD	ADOPTED	MAXIMUM SPEED	APPROXIMATE RANGE	COMPATIBILITY
802.11b	1999	11 Mbps	100–300 feet	Early 802.11 version
802.11a	1999	54 Mbps	50–100 feet	Incompatible with 802.11b and 802.11g
802.11g	2003	54 Mbps	50–100 feet	Compatible with 802.11b
802.11n	2009	200+ Mbps	150–300 feet	Compatible with all 802.11 standards

(Source: Wikipedia.org and IEEE.org)

FIGURE 10-41 This table shows various Wi-Fi standards and characteristics. Maximum speed is measured in Mbps (megabits per second).

Early IEEE 802.11 standards had limited transmission capacity and were not popular. Later versions, such as **802.11g**, offered increased bandwidth and were widely accepted by the IT industry. The more recent **802.11n** uses **multiple input/multiple output** (**MIMO**) technology to boost performance. MIMO relies on multiple data paths, also called **multipath design**, to increase bandwidth and range. An even newer MIMO version, called **802.11y**, is currently being tested. If wireless capacity continues to expand and security issues can be overcome, WLANs could replace wired networks in many situations. Wireless security is discussed in detail in Chapter 12, Managing Systems Support and Security.

Wireless Network Topologies

Like wired networks, wireless networks also can be arranged in different topologies. The three major network topologies available for IEEE 802.11 WLANs are the Basic Service Set, the Extended Service Set, and the Independent Service Set.

The **Basic Service Set** (**BSS**), also called the **infrastructure mode**, is shown is Figure 10-42. In this configuration, a central wireless device called an **access point** or **wireless access point** (**WAP**), is used to serve all wireless clients. The access point is similar to a hub in the LAN star topology, except it provides network services to wireless clients instead of wired clients. Because access points use a single communications medium, the air, they broadcast all traffic to all clients, just as a hub would do in a wired network. Typically, the access point itself is connected to a wired network, so wireless clients can access the wired network.

The second wireless topology is the **Extended Service Set** (**ESS**), as shown in Figure 10-43 on the next page. An Extended Service Set is made up of two or more Basic Service Set networks. Thus, using an ESS topology, wireless access can be expanded over a wide area. Each access point provides wireless services over a limited range. As a client moves away from one access point and closer to another, a process called **roaming** automatically allows the client to associate with the stronger access point, allowing for undisrupted service.

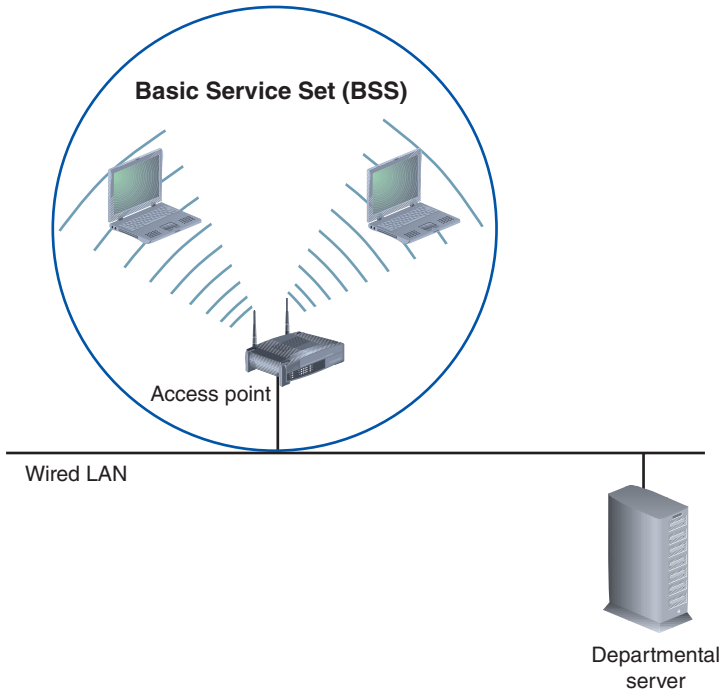


FIGURE 10-42 Basic Service Set (infrastructure mode).

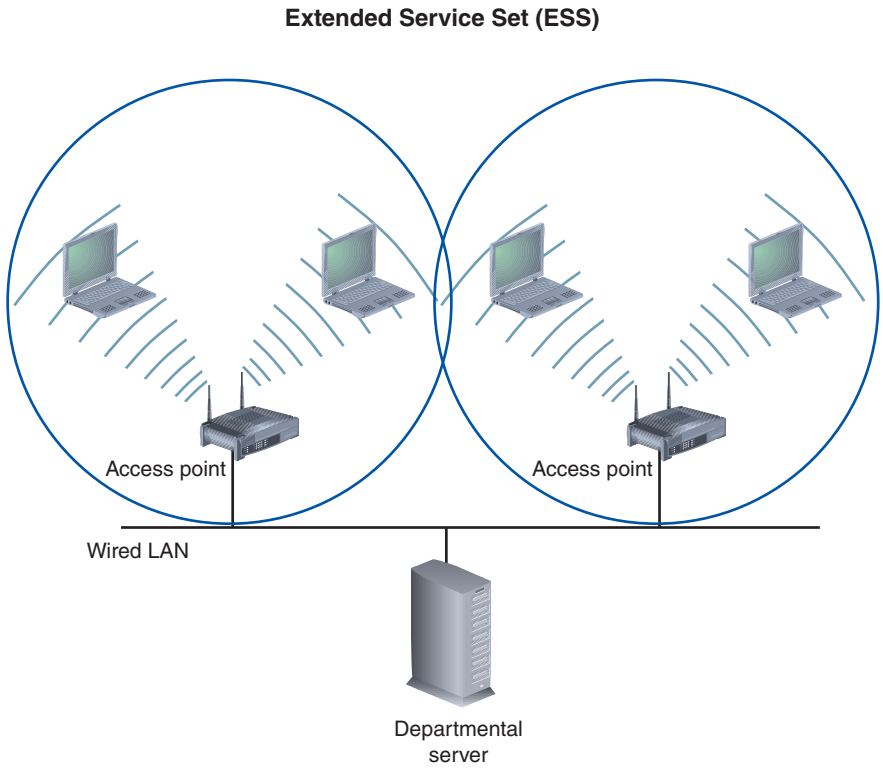


FIGURE 10-43 Extended Service Set.

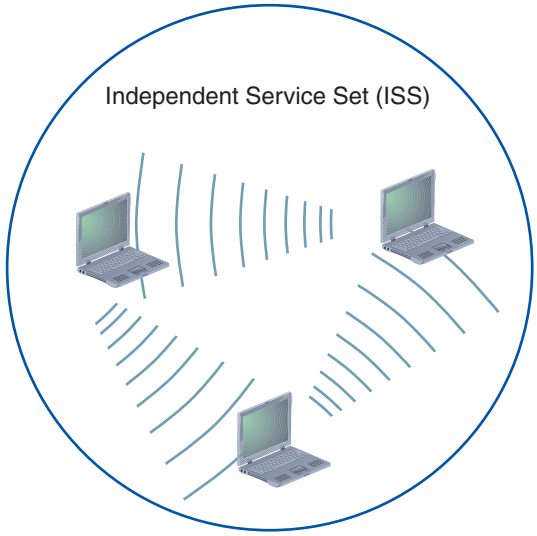


FIGURE 10-44 Independent Service Set (peer-to-peer mode).

The third wireless topology is the **Independent Service Set (ISS)**, as shown in Figure 10-44. In the ISS, also called **peer-to-peer mode**, no access point is used. Instead, wireless clients connect to each other directly. Most business WLANs use access points to provide wireless services, and do not utilize the Independent Service Set. However, ISS is well-suited to situations requiring quick data transfer among clients.

Wireless Trends

Wireless technology has brought explosive change to the IT industry, and will continue to affect businesses, individuals, and society. Even in the ever-changing world of IT, it would be difficult to find a more dynamic area than wireless technology.

With the growing popularity of 802.11, many firms offer networking products, services, and information. One of the most significant groups is the **Wi-Fi Alliance**, which maintains a Web site at *www.wi-fi.org*. According to the site, the Alliance is a nonprofit international association formed in 1999 to certify interoperability of wireless network products based on IEEE 802.11 specifications. Products that meet the requirements are certified as **Wi-Fi** (**wireless fidelity**) compatible. Currently the Wi-Fi Alliance has over 300 member companies from around the world, and over 4,200 products have received Wi-Fi certification. The stated goal of the Wi-Fi Alliance is to enhance the user experience through product interoperability.

Even though they have many advantages, wireless networks also have limitations and disadvantages. For example, because 802.11b and 802.11g devices use the 2.4 GHz band, these devices can pick up interference from appliances such as microwave ovens and cordless telephones that use the same band. More important, wireless networks pose major security concerns because wireless transmissions are much more susceptible to interception and intrusion than wired networks. These issues are discussed in detail in Chapter 12, Managing Systems Support and Security.

In addition to Wi-Fi, another form of wireless transmission called **Bluetooth** is very popular for short-distance wireless communication that does not require high power. Examples of Bluetooth devices include wireless keyboards, mice, printers, cell phone headsets, and digital cameras, among others. People with Bluetooth-equipped phones or PDAs can even beam information to each other and exchange digital notes.

Although the expansion of Wi-Fi has been dramatic, future technology promises even greater wireless speed, range, and compatibility. For example, in addition to 802.11 protocols for LANs, IEEE is working on **802.16** standards, which are broadband wireless communications protocols for **MANs** (**metropolitan area networks**). These specifications, which IEEE calls **WiMAX**, are expected to enable wireless multimedia applications with a range of up to 30 miles.

CASE IN POINT 10.4: SPIDER IT SERVICES

Spider IT Services specializes in custom network design and installation. Firms hire Spider to do an overall analysis of their network needs, including a detailed cost-benefit study. Recently, a problem arose. One of Spider's clients complained that the relatively new network was too slow and lacked sufficient capacity. Reviewing the case, Spider's top management realized that the rapidly growing client had simply outgrown the network much earlier than anticipated.

Could this problem have been avoided? Note that IBM, in the article shown in Figure 10-19 on page 467, commented that performance can "degrade exponentially" in certain kinds of network situations. Consider the IBM article and other material in this chapter, and offer your views.

SYSTEMS DESIGN COMPLETION

System architecture marks the end of the systems design phase of the SDLC. Recall that back in the systems analysis phase, all functional primitives were identified and documented with process descriptions. The objective then was to identify the system's functions and determine *what* each logical module would do, without attempting to determine *how* that function would be carried out. Moving from analysis to design tasks, the development process continued with consideration of output and user interface design, data design, and system architecture issues. Now, based on a clear definition of system requirements and design, software applications can be developed, documented, and tested as part of the systems implementation phase of the SDLC, which is described in Chapter 11, Managing System Implementation.

Developers must also consider system management and support tools that can monitor system performance, deal with fault management, handle backup, and provide for disaster recovery. These topics are covered in detail in Chapter 12, Managing Systems Support and Security.

The final activities in the systems design phase are preparing a system design specification, obtaining user approval, and delivering a presentation to management.

System Design Specification

The **system design specification** is a document that presents the complete design for the new information system, along with detailed costs, staffing, and scheduling for completing the next SDLC phase — systems implementation.

The system design specification is the baseline against which the operational system will be measured. Unlike the system requirements document, which is written for users to understand, the system design specification is oriented toward the programmers who will use it to create the necessary programs. Some sections of the system requirements document are repeated in the system design specification, such as process descriptions, data dictionary entries, and data flow diagrams.

The system design specification varies in length, so you must organize it carefully and number all pages in sequence. You should include a cover page, a detailed table of contents, and an index. The contents of the system design specification depend on company standards and the complexity of the system. A typical system design specification typically includes the following sections.

1. *Management Summary.* This is a brief overview of the project for company managers and executives. It outlines the development efforts to date, provides a current status report, summarizes project costs, reviews the benefits of the new system, presents the systems implementation schedule, and highlights any issues that management will need to address.
2. *System Components*. This section contains the complete design for the new system, including the user interface, outputs, inputs, files, databases, and network specifications. You should include source documents, report and screen layouts, DFDs, and all other relevant documentation. You also should include the requirements for all support processing, such as backup and recovery, start-up processing, and file retention. If the purchase of a software package is part of the strategy, you must include any interface information required between the package and the system you are developing. If you use a CASE design tool, you can print design diagrams and most other documentation directly from the tool.
3. *System Environment*. This section describes the constraints, or conditions, affecting the system, including any requirements that involve operations, hardware, systems software, or security. Examples of operational constraints include transaction volumes that must be supported, data storage requirements, processing schedules, reporting deadlines, and online response times.
4. *Implementation Requirements*. In this section, you specify start-up processing, initial data entry or acquisition, user training requirements, and software test plans.
5. *Time and Cost Estimates*. This section provides detailed schedules, cost estimates, and staffing requirements for the systems development phase and revised projections for the remainder of the SDLC. You also present total costs-to-date for the project and compare those costs with your prior estimates.
6. *Additional Material.* Other material can be included at the end of the system design specification. In this section, you might insert documents from earlier phases if they would be helpful to readers.

User Approval

Users must review and approve the interface design, report and menu designs, data entry screens, source documents, and other areas of the system that affect them. The review and approval process continues throughout the systems design phase. When you complete the design for a report, you should meet with users to review the prototype, adjust the design if necessary, and obtain written approval. Chapter 8 contains guidelines and suggestions about report design.

Securing approvals from users throughout the design phase is very important. That approach ensures that you do not have a major task of obtaining approvals at the end, it keeps the users involved with the system's development, and it gives you feedback about whether or not you are on target. Some sections of the system design specification might not interest users, but anything that does affect them should be approved as early as possible.

Other IT department members also need to review the system design specification. IT management will be concerned with staffing, costs, hardware and systems software requirements, network impact, and the effect on the operating environment when the new system is added. The programming team will want to get ready for its role, and the operations group will be interested in processing support, report distribution, network loads, integration with other systems, and any hardware or software issues for which they need to prepare. You must be a good communicator to keep people up to date, obtain their input and suggestions, and obtain necessary approvals.

When the system design specification is complete, you distribute the document to a target group of users, IT department personnel, and company management. You should distribute the document at least one week before your presentation to allow the recipients enough time to review the material.

Presentations

Usually, you will give several presentations at the end of the systems design phase. The presentations give you an opportunity to explain the system, answer questions, consider comments, and secure final approval. Part A of the Systems Analyst's Toolkit can provide valuable guidelines and tips about oral presentations.

The first presentation is to the systems analysts, programmers, and technical support staff members who will be involved in future project phases or operational support for the system. Because of the audience, the presentation is technically oriented.

Your next presentation is to department managers and users from departments affected by the system. As in the first presentation, your primary objective is to obtain support and approval for the systems design. This is not a technical presentation; it is aimed at user interaction with the system and management's interest in budgets, schedules, staffing, and impact on the production environment.

The final presentation is delivered to management. By the time you give this presentation, you should have obtained all necessary approvals from prior presentations, and you should have the support of users and the IT department. Just like the management presentation at the end of the systems analysis phase, this presentation has a key objective: to obtain management's approval and support for the next development step — systems implementation — including a solid commitment for financial and other resources needed.

Based on the presentation and the data you submitted, management might reach one of three decisions: proceed with systems development, perform additional work on the systems design phase, or terminate the project.

TOOLKIT TIME

The Communication Tools in Part A of the Systems Analyst's Toolkit can help you develop better reports and presentations. To learn more about these tools, turn to Part A of the four-part Toolkit that follows Chapter 12.

A QUESTION OF ETHICS

The new accounting system is operational, but feedback from users has been negative. The most common complaint is that the system is not user-friendly. Some people in the IT department think that more user training would solve the problem. However, Sam, the IT manager, is opposed to a fresh round of training. "Let's just set up the network to monitor the users' keystrokes and mouse clicks, and see what the patterns are," he suggested. "We can analyze the data and come up with tips and suggestions that would make the system easier to use."

Your initial reaction is that Sam is wrong, for two reasons. First, you believe that monitoring would not be an effective method to learn what users really want. In your view, that should have been done in the system requirements phase. Second, you are bothered by an ethical question: Even though the proposed monitoring would involve company business, the company network, and company time, you feel that many users would resent the unannounced monitoring, and might feel that their performance or other computing activities were being appraised without their knowledge.

Sam has asked to you to write up a recommendation. What will you say about the ethical question that troubles you?

Chapter Summary

An information system combines hardware, software, data, procedures, and people into a system architecture. The architecture translates the system's logical design into a physical structure that includes hardware, software, and processing methods. The software consists of application programs, also called applications, that handle the input, manage the processing logic, and provide the required output.

Before selecting an architecture, the analyst must consider enterprise resource planning, initial cost and TCO, scalability, Web integration, legacy interface requirements, processing options, and security issues.

Enterprise resource planning (ERP) establishes an enterprise-wide strategy for IT resources and specific standards for data, processing, network, and user interface design. Companies can extend ERP systems to suppliers and customers in a process called supply chain management. A systems analyst must assess initial cost and TCO and ensure that the design is scalable. Scalability means that a system can be expanded, modified, or downsized easily to meet business needs. The analyst also must consider if the system will be Web-centric and follow Internet design protocols, and if it must interface with existing systems, called legacy systems. System security is an important concern throughout the design process, especially for e-commerce applications that involve credit card and personal data. Processing options affect system design and resources required.

An architecture requires servers and clients. Servers are computers that supply data, processing services, or other support to one or more computers called clients. In mainframe architecture, the server performs all processing, and terminals communicate with the centralized system. Clients can be connected in distributed systems to form local area networks (LANs) or wide area networks (WANs). A typical LAN design involves file server design, where the client requests a copy of a data file and the server responds by transmitting the entire file to the client.

Client/server architecture divides processing between one or more clients and a central server. In a typical client/server system, the client handles the entire user interface, including data entry, data query, and screen presentation logic. The server stores the data and provides data access and database management functions. Application logic is divided in some manner between the server and the clients. In a typical client/server interaction, the client submits a request for information from the server, which carries out the operation and responds to the client. Compared to file server designs, client/server systems are more scalable and flexible.

A fat, or thick, client design places all or most of the application processing logic at the client. A thin client design places all or most of the processing logic at the server. Thin client designs provide better performance, because program code resides on the server, near the data. In contrast, a fat client handles more of the processing, and must access and update the data more often. Compared with maintaining a central server, fat client TCO also is higher, because of initial hardware and software requirements and the ongoing expense of maintaining and updating remote client computers. The fat client design is simpler to develop, because the architecture resembles traditional file server designs where all processing is performed at the client.

Client/server designs can be two-tier or three-tier (also called n-tier). In a two-tier design, the user interface resides on the client, all data resides on the server, and the application logic can run either on the server or on the client, or be divided between the client and the server. In a three-tier design, the user interface runs on the client and the data is stored on the server, just as with a two-tier design. A three-tier design also has a middle layer between the client and server that processes the client requests and translates them into data access commands that can be understood and carried out by the server. The middle layer is called an application server, because it provides the application logic, or business logic. Middleware is software that connects dissimilar applications and enables them to communicate and pass data. In planning the system design, a systems analyst also must consider cost-benefit and performance issues.

The Internet has had an enormous impact on system architecture. In implementing a design, an analyst should consider e-commerce strategies, the availability of packaged solutions, and corporate portals, which are entrances to a multifunction Web site. The analyst also should understand the concepts of cloud computing and Web 2.0, which are shaping the future of Internet computing.

The primary processing methods are online and batch processing. Users interact directly with online systems that continuously process their transactions when and where they occur and continuously update files and databases. In contrast, batch systems process transactions in groups and execute them on a predetermined schedule. Many online systems also use batch processing to perform routine tasks, such as handling reports and accounting entries.

Networks allow the sharing of hardware, software, and data resources in order to reduce expenses and provide more capability to users. The network is represented by a seven-layer logical model called the OSI (Open Systems Interconnection) model. Various OSI layers handle specific functions as data flows down from the sending computer and up into the receiving computer.

The way a network is configured is called the network topology. Networks typically are arranged in four patterns: hierarchical, bus, ring, and star. A single mainframe computer usually controls a hierarchical network, a bus network connects workstations in a single-line communication path, a ring network connects workstations in a circular communication path, and a star network connects workstations to a central computer or networking device called a switch. Wireless networks, or WLANs, based on IEEE 802.11 standards, have seen explosive growth, especially in situations where the flexibility of

wireless is important. The IEEE 802.11n standard uses MIMO, or multipath technology, which has increased wireless network speed and range. WLANs have three major topologies: BSS, ESS, and ISS. Although wireless networks are very popular, they do have some limitations and disadvantages, including interference and security concerns.

The system design specification presents the complete systems design for an information system and is the basis for the presentations that complete the systems design phase. Following the presentations, the project either progresses to the systems development phase, requires additional systems design work, or is terminated.

Key Terms and Phrases

802.11 *482*
802.11g *483*
802.11n *483*
802.11y *483*
802.16 *485*
access point *483*
amendment *482*
application logic *464*
application server *464*
applications *454*
bandwidth *472*
Basic Service Set (BSS) *483*
batch processing *475*
Bluetooth *485*
bus network *479*
business logic *464*
client/server architecture *461*
clients *458*
cloud computing *470*
data frames *480*
data processing center *459*
distributed database management system (DDBMS) *466*
distributed systems *460*
enterprise resource planning (ERP) *454*
environment *454*
Extended Service Set (ESS) *483*
extensibility *454*
fat client *464*
file server *460*
file sharing architecture *460*
File Transfer Protocol (FTP) *477*
gateway *481*
hierarchical network *478*
hub *480*
Independent Service Set (ISS) *484*
infrastructure mode *483*
Institute of Electrical and Electronics Engineers (IEEE) *482*
Internet operating system *473*
legacy data *462*
legacy systems *457*
local area network (LAN) *460*
logical topology *477*
mainframe architecture *458*
MAN (metropolitan area network) *485*
Mbps (megabits per second) *482*
mesh network *480*
middleware *465*
multipath design *483*
multiple input/multiple output (MIMO) *483*
Multistation Access Unit (MAU) *480*
network topology *477*
n-tier *464*
node *456*
online system *474*
OSI (Open Systems Interconnection) model *476*
peer-to-peer mode *484*
physical topology *477*
platform *454*
point-of-sale (POS) *475*
portal *470*
protocol *477*
proxy server *481*
ring network *479*
roaming *483*
router *481*
scalability *454*
scaling on demand *472*
server *458*
stand-alone *460*
star network *480*
supply chain management (SCM) *454*
switch *480*
system architecture *452*
system design specification *486*
terminal *459*
thick client *464*
thin client *464*
three-tier *464*
Transmission Control Protocol/Internet Protocol (TCP/IP) *477*
transparent *460*
two-tier *464*
Web 2.0 *472*
Web-centric *457*
Wi-Fi Alliance *484*
Wi-Fi (wireless fidelity) *484*
wide area network (WAN) *460*
wiki *473*
wireless access point (WAP) *483*
wireless local area network (WLAN) *482*
WiMAX *485*

Learn It Online

Instructions: To complete the Learn It Online exercises, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the resources for this chapter, and click the link for the exercise you want to complete.

1 Chapter Reinforcement

TF, MC, and SA

Click one of the Chapter Reinforcement links for Multiple Choice, True/False, or Short Answer. Answer each question and submit to your instructor.

2 Flash Cards

Click the Flash Cards link and read the instructions. Type 20 (or a number specified by your instructor) in the Number of playing cards text box, type your name in the Enter your Name text box, and then click the Flip Card button. When the flash card is displayed, read the question and then click the ANSWER box arrow to select an answer. Flip through the Flash Cards. If your score is 15 (75%) correct or greater, click Print on the File menu to print your results. If your score is less than 15 (75%) correct, then redo this exercise by clicking the Replay button.

3 Practice Test

Click the Practice Test link. Answer each question, enter your first and last name at the bottom of the page, and then click the Grade Test button. When the graded practice test is displayed on your screen, click Print on the File menu to print a hard copy. Continue to take practice tests until you score 80% or better.

4 Who Wants To Be a Computer Genius?

Click the Computer Genius link. Read the instructions, enter your first and last name at the bottom of the page, and then click the Play button. When your score is displayed, click the PRINT RESULTS link to print a hard copy.

5 Wheel of Terms

Click the Wheel of Terms link. Read the instructions, and then enter your first and last name and your school name. Click the PLAY button. When your score is displayed on the screen, right-click the score and then click Print on the shortcut menu to print a hard copy.

6 Crossword Puzzle Challenge

Click the Crossword Puzzle Challenge link. Read the instructions, and then click the Continue button. Work the crossword puzzle. When you are finished, click the Submit button. When the crossword puzzle is redisplayed, submit it to your instructor.

SCR Associates Case Simulation Session 10: Systems Architecture

Overview

The SCR Associates case study is a Web-based simulation that allows you to practice your skills in a real-world environment. The case study transports you to SCR's intranet, where you complete 12 work sessions, each aligning with a chapter. As you work on the case, you will receive e-mail and voice mail messages, obtain information from SCR's online libraries, and perform various tasks.

How do I use the case?

- Review the SCR background material in Chapter 1.
- Read the Preview for this session and study the Task List.
- Visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the **SCR Case Simulation**, and locate the intranet link.
- Enter your name and the password **sad9e**. An opening screen will display the 12 sessions.
- Select this session. Check your e-mail and voice mail carefully, and then work on the tasks.

Preview: Session 10

Your supervisor, Jesse Baker, wants you to be familiar with the main issues that a systems analyst should consider when selecting an architecture, including enterprise resource planning, initial costs and TCO, scalability, Web integration, legacy interface requirements, processing options, and security issues.

Task List

1. *Jesse wants me to recommend a vendor who offers an ERP strategy. I need to review the SAP and Oracle Web sites, and at least two others that offer ERP solutions, and reply to her with the results and the reasons for my recommendations.*

2. *Visit SCR's data library to review SCR's network configuration and then send Jesse a recommendation for the TIMS system architecture. She wants me to suggest an overall client/server design, number of tiers, and network topology. She also asked me to comment on these issues: legacy data, Web-centricity, scalability, security, and batch processing that might be needed. Jesse said it was OK to make reasonable assumptions in my proposal to her.*

3. *Perform research on the Internet to learn more about TCO, and develop a TCO checklist that includes the five most important elements of TCO, because of their magnitude or potential impact on TIMS.*

4. *Prepare a system design specification as Jesse requested.*

FIGURE 10-45 Task list: Session 10.

Chapter Exercises

Review Questions

1. Define the term system architecture. Define the term scalability, and explain why it is important to consider scalability in system design.
2. When selecting an architecture, what items should a systems analyst consider as part of the overall design checklist?
3. What is enterprise resource planning (ERP)? What is supply chain management?
4. Explain the term server and provide an example of server-based processing; explain the term client and provide an example of client-based processing.
5. Describe client/server architecture, including fat and thin clients, client/server tiers, and middleware.
6. Describe the impact of the Internet on system architecture. Include examples.
7. Explain the difference between online processing and batch processing and provide an example of each type.
8. Explain the difference between a LAN and a WAN, define the term topology, and draw a sketch of each wired and wireless network model. Also describe the IEEE 802.11g and 802.11n amendments, and multiple input-multiple output (MIMO) technology.
9. Explain the differences between the BSS, ESS, and ISS wireless topologies. Which is rarely used in business? To what kind of network do the 802.16 standards apply?
10. List the sections of a system design specification, and describe the contents.

Discussion Topics

1. Information technology has advanced dramatically in recent years. At the same time, enormous changes in the business world have occurred as companies reflect global competition and more pressure for quality, speed, and customer service. Did the new technology inspire the business changes, or was it the other way around?
2. Internet-based sales have shown explosive growth in recent years. How does B2B interaction differ from consumer-based Internet marketing, and why is it growing so rapidly?
3. This chapter described seven guidelines that a systems analyst might use when considering an architecture. In your view, are all the items of equal weight and importance, or should some be ranked higher? Justify your position.
4. One manager states, "When a new system is proposed, I want a written report, not an oral presentation, which is like a sales pitch. I only want to see the facts about costs, benefits, and schedules." Do you agree with that point of view?

Projects

1. Visit the IT department at your school or a local company to determine what type of network it is using. Draw a sketch of the network configuration.
2. Prepare a 10-minute talk explaining Web 2.0 and cloud computing to a college class. Using the text and your own Internet research, briefly describe the five most important points you will include in your presentation.
3. Perform research on the Internet to identify an ASP that offers Web-based business solutions, and write a brief memo describing the firm and its services.
4. Perform research on the Internet to learn about trends in wireless networking, and typical costs involved in the installation of a wireless LAN.

Apply Your Knowledge

The Apply Your Knowledge section contains four mini-cases. Each case describes a situation, explains your role in the case, and asks you to respond to questions. You can answer the questions by applying knowledge you learned in the chapter.

1 Digital Dynamics

Situation:

After three years as a successful Web design firm in Southern California, Digital Dynamics has decided to add two new business ventures: a group that specializes in supplying qualified employees to high-tech firms and a training division that offers online courses in advanced Web design and e-commerce skills. As a senior systems analyst, you have been asked to study the situation and make recommendations.

1. Should Digital Dynamics adopt ERP? What specific advantages would ERP offer?
2. How could the concept of supply chain management apply to a company's service-based division? Provide some specific suggestions.
3. Should Digital Dynamics use separate portals for employees, customers, and suppliers?
4. Is the experience of other companies relevant? Use the Internet to locate examples of Web-based firms that offer personnel services and technical training. How would you evaluate the Web sites you visited? What specific features impressed you favorably or unfavorably?

2 R/Way Trucking

Situation:

As you learned earlier in this chapter, R/Way is a small but rapidly growing trucking company headquartered in Cleveland, Ohio. R/Way's information system currently consists of a file server and three workstations where freight clerks enter data, track shipments, and prepare freight bills. To perform their work, the clerks obtain data from a file server and use database and spreadsheet programs stored on stand-alone PCs to process the data. At your meeting yesterday, R/Way's president approved your recommendation to create a relational database to handle R/Way operations and provide links for R/Way shippers and customers.

1. Review the concept of supply chain management. Although R/Way offers services rather than products, could that concept apply to the design of R/Way's new system? If so, how?
2. What would be the advantages of selecting an Internet-based architecture for R/Way's system?
3. Should R/Way's new system be based on file-server or client/server architecture? Why?
4. What would be the pros and cons of selecting in-house development versus a packaged solution for the R/Way system?

3 Nothing But Net

Situation:

Nothing But Net is an IT consulting firm that specializes in e-commerce solutions. As a newly hired systems analyst, you have been asked to research some current topics.

1. Obtain at least two estimates of how much consumers are projected to spend on Internet purchases during the next three years. Are the estimates similar? If not, which forecast do you believe, and why?
2. Many of Nothing But Net's customers are start-up firms that must fight hard to attract investment capital, and many traditional lending institutions are skeptical of new Web-based firms. Perform research to determine the mortality rate of new e-commerce firms that use the Web as their primary marketing channel, and write a brief memo that describes the results of your research.
3. Some IT professionals predict that traditional companies will increase their Internet marketing efforts, making it even harder for new Web-based firms to compete. Perform research to find out more about the topic and share your results with the class.
4. Suppose you were asked to draft a sales brochure for Nothing But Net. List all the services in which potential customers might be interested.

4 Aunt Ann's Kitchen

Situation:

Aunt Ann's Kitchen offers a line of specialty food products to institutional customers and restaurant chains. The firm prides itself on using only the finest ingredients and preparation methods. The owner, Ann Rose, hired you as an IT consultant to help her plan the system architecture for a new WLAN that will connect employee computers to the main (wired) network. She asked you to start with the following questions:

1. What possible IEEE 802.11 amendments could be used for the new system? What are the pros and cons of each amendment?
2. Choose an amendment to implement and explain your choice.
3. Suppose that microwave ovens and cordless telephones are used extensively in some parts of the facility. Would that affect your IEEE 802.11 amendment choice? If so, why?
4. Suppose that the new WLAN will also provide roaming services for employees with portable notebook computers. Which wireless topology will be required?

Case Studies

Case studies allow you to practice specific skills learned in the chapter. Each chapter contains several case studies that continue throughout the textbook, and a chapter capstone case.

New Century Health Clinic

New Century Health Clinic offers preventive medicine and traditional medical care. In your role as an IT consultant, you will help New Century develop a new information system.

Background

The New Century clinic associates accepted your interface, output, input, and data designs and your recommendation to install a server and four personal computers as clients on a local area network. The network will include a tape backup unit and Internet access via a modem that can exchange data with insurance companies. A high-speed laser printer and an impact printer for multipart forms will be accessible by any of the four PCs. Now you will determine system architecture for the New Century system.

When you created ERDs and record designs for New Century during the data design process in Chapter 9, you considered whether to use a file-processing or database approach. As you know, each strategy has advantages and disadvantages, depending on the specific hardware and software environment and business requirements. At this point, you must decide which way to proceed, and Dr. Jones will accept your recommendation (with your instructor's approval). You should start by reviewing the DFDs and object-oriented diagrams that you prepared in the systems analysis phase, and the ERDs and table designs that you created in the systems design phase. Then, review the system architecture checklist at the beginning of this chapter.

Assignments

1. What would be the advantages of selecting an Internet-based architecture for the New Century system?
2. Should the New Century system be based on file-server or client/server architecture? Why?
3. Could the New Century system use both online and batch processing? How?
4. Prepare an outline for a system design specification and describe the contents of each section.

PERSONAL TRAINER, INC.

Personal Trainer, Inc., owns and operates fitness centers in a dozen Midwestern cities. The centers have done well, and the company is planning an international expansion by opening a new "supercenter" in the Toronto area. Personal Trainer's president, Cassia Umi, hired an IT consultant, Susan Park, to help develop an information system for the new facility. During the project, Susan will work closely with Gray Lewis, who will manage the new operation.

Background

Susan and Gray finished their work on the user interface, input, and output design. They developed a user-centered design that would be easy to learn and use. Now Susan turned her attention to the architecture for the new system.

Susan wanted to consider enterprise resource planning, total cost of ownership, scalability, Web integration, legacy systems, processing methods, and security issues. She also needed to select a network plan, or topology, that would dictate the physical cabling and network connections, or consider a wireless network. When all these tasks were completed, she would submit a system design specification for approval.

Assignments

1. Would an ERP strategy work well for Personal Trainer? Investigate ERP strategies and products available from Internet vendors and submit a recommendation based on your research.
2. If Susan chooses a client/server architecture, what issues must she consider? Prepare a checklist for her that includes the main topics and issues she should consider.
3. What would be the benefits of using a wireless network? What would be the drawbacks?
4. Prepare an outline for a system design specification and describe the contents of each section.

CHAPTER CAPSTONE CASE: SoftWear, Limited

SoftWear, Limited (SWL), is a continuing case study that illustrates the knowledge and skills described in each chapter. In this case study, the student acts as a member of the SWL systems development team and performs various tasks.

Background

Jane Rossman, manager of applications, and Rick Williams, systems analyst, had several meetings with True Blue Systems, the consulting firm hired to assist SWL in implementing the new ESIP system. Michael Jeremy, SWL's finance vice president, requested that True Blue also make recommendations about a possible SWL intranet to link all SWL locations and support client/server architecture.

The initial report from True Blue indicated that the new ESIP system should be designed as a DBMS so it could interface with the new mainframe payroll package. True Blue suggested that the ESIP system be implemented on a server in the payroll department and developed as a Microsoft Access application. They felt that this approach provided a relational database environment, client/server capability, and SQL command output to communicate with the mainframe. Figure 10-46 shows the proposed design of the system.

Jane Rossman met with Ann Hon to review True Blue's report and get her approval for training IT staff members. Jane had some prior experience in Access application development, but Rick had none, so Jane suggested that Rick and Becky Evans, another systems analyst, should attend a one-week workshop. Ann agreed.

Ann, Jane, and Rick met with Michael Jeremy to get his approval before proceeding further. He asked them to develop a specific budget and timetable including all necessary hardware, software, and training costs. They had most of the information, but they needed some help from True Blue to estimate the cost of network implementation, installation, and physical cabling.

The first phase of the project would use a local area network to link the various headquarter departments to the mainframe. A second phase, proposed by True Blue, would connect all SWL locations to a wide area network, with the possibility that employees could access their individual ESIP accounts over the internal network or from outside the company using the Internet.

A week later, Ann received a memo from Michael Jeremy that said he had approved the project and that she should start work immediately.

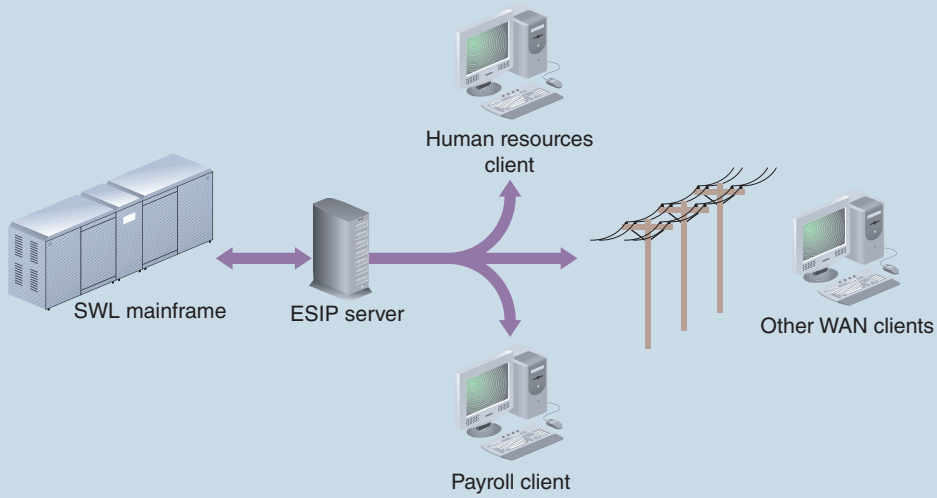


FIGURE 10-46 Diagram of the proposed ESIP system.

System Architecture

The ESIP development team included Jane, Rick, and Becky. The group discovered that the Microsoft Access application consists of interactive objects such as tables, queries, forms, reports, macros, and code modules.

They decided to begin by reviewing the entity-relationship diagrams they prepared previously to determine the overall structure of the DBMS design. Then they identified the tables, reviewed the relationships among them, and analyzed the record designs they had developed.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

They also reviewed output requirements, input screen designs, processing considerations, backup and recovery procedures, and controls that must be built into the new system.

As recommended by True Blue Systems, the new ESIP system would be implemented as a client/server design, with the data stored on the payroll department server, which would be linked to clients in the payroll and human resources department.

Planning the System

In their first meeting, Jane asked the team to define all the tasks that the new system would perform, including a list of all reports and other required output. Jane explained that ESIP data would be stored on the server, but the application logic and objects such as forms, queries, and reports would be located on client workstations. Separating the objects from the data would provide better security and reduce the network loads, she explained.

Security Issues

In their next planning session, Jane asked the group to consider all security issues affecting the new system design. Because the system contains payroll data, it is important to control user access and updates. The team decided to use the security features in Access for control, including passwords and user and workgroup accounts for employees authorized to use the ESIP application. Each user would be assigned a permission level that grants access only to certain objects.

Jane explained her plan for system security by saying, "We'll create a comprehensive security plan later that will cover all the operational security issues for the new system. Meanwhile, let's go back to the department heads, Michael Jeremy and Rob King, to get their input on what the security levels should be." She added that "Users will be allowed to create and modify certain forms and reports, but most other actions will be permitted only by authorized IT department members."

Creating the Database Objects

Before creating the database objects, the team reviewed the ERDs and verified that the records designs were in 3NF. In addition, they verified that the new payroll system permitted cross-platform access because the ESIP system required data from the payroll master file. They discovered that the payroll package used a standard data format called open database connectivity (ODBC) that supported links to the Access database. After planning the system, they started creating the objects.

Planning the User Interface

From earlier interviews, the IT team knew that users in the payroll and human resources departments wanted an interface that would be easy to learn and simple to operate. Jane asked Becky to start designing a main form, or switchboard, that would display automatically when the ESIP application started. All ESIP screen forms would use buttons, menus, and icons as shown in Figure 10-47.

Becky created a prototype of the input screens to show to users. After securing user approval, the screen designs were added to the system specification document for the ESIP system.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

Using Visual Basic and Macros

Because she was a programmer before her assignment as a systems analyst, Becky wanted to know if they would be using Visual Basic as a program development language. Jane told her that they would write many of the procedures in Visual Basic because it allows more powerful data manipulation than macros and makes it easier to customize error messages. They would use macros, however, to speed up the development process and handle simple tasks such as opening and closing forms and running reports.

FIGURE 10-47 Sample of a main switchboard that displays when the ESIP system starts.

Completing the Systems Design Phase

The IT team completed the systems design phase by writing the documentation and designing backup and recovery, file retention, and start-up procedures. The final step was to develop a system design specification for the ESIP system.

Rick and Becky showed a draft of the system design specification to Jane and Ann. After incorporating their changes, they e-mailed copies to a distribution list of users, IT staff members, and managers. Their presentations to users and IT staff members went well, and at the management presentation, they received final approval to implement the ESIP system.

SWL Team Tasks

1. Rick Williams asked you to suggest software products that can provide network management features suitable for a network with 25–50 users. You can use the Internet to research the topic.
2. In the Background section of the SWL case, you learned that True Blue Systems had recommended a local area network to link the various headquarters departments to the SWL mainframe. Should a wireless network have been considered? Your task is to research wireless network products and submit a recommendation for a WLAN installation, supported by an explanation of wireless advantages and possible limitations. Be sure to consider flexibility, scalability, and security issues.
3. Assume that the recommendation you made in the preceding task was accepted. Now you have been asked to develop a specific budget for a WLAN that would consist of 50 workstations. You need to research the TCO of this project, including the cost of hardware, software, installation, and maintenance. You can make reasonable assumptions where you might not have specific facts, but you should state those assumptions clearly.
4. Michael Jeremy, finance vice president, has been reading about cloud computing in his favorite IT magazine, and he is considering the possibility of using the concept at SWL. He asked you to answer the following questions: What would be the benefits of a cloud computing architecture? What would be the disadvantages? What are examples of cloud computing services currently available? Perform research on the Internet and prepare a brief report for him.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

Manage the SWL Project

You have been asked to manage SWL's new information system project. One of your most important activities will be to identify project tasks and determine when they will be performed. Before you begin, you should review the SWL case in this chapter. Then list and analyze the tasks, as follows:

LIST THE TASKS Start by listing and numbering at least 10 tasks that the SWL team needs to perform to fulfill the objectives of this chapter. Your list can include SWL Team Tasks and any other tasks that are described in this chapter. For example, Task 3 might be to Review the logical design, and Task 6 might be to Begin the physical design process.

ANALYZE THE TASKS Now study the tasks to determine the order in which they should be performed. First identify all concurrent tasks, which are not dependent on other tasks. In the example shown in Figure 10-48. Tasks 1, 2, 3, 4, and 5 are concurrent tasks, and could begin at the same time if resources were available.

Other tasks are called dependent tasks, because they cannot be performed until one or more earlier tasks have been completed. For each dependent task, you must identify specific tasks that need to be completed before this task can begin. For example, you would want to review the logical design before you could begin the physical design process, so Task 6 cannot begin until Task 3 is completed, as Figure 10-48 shows.

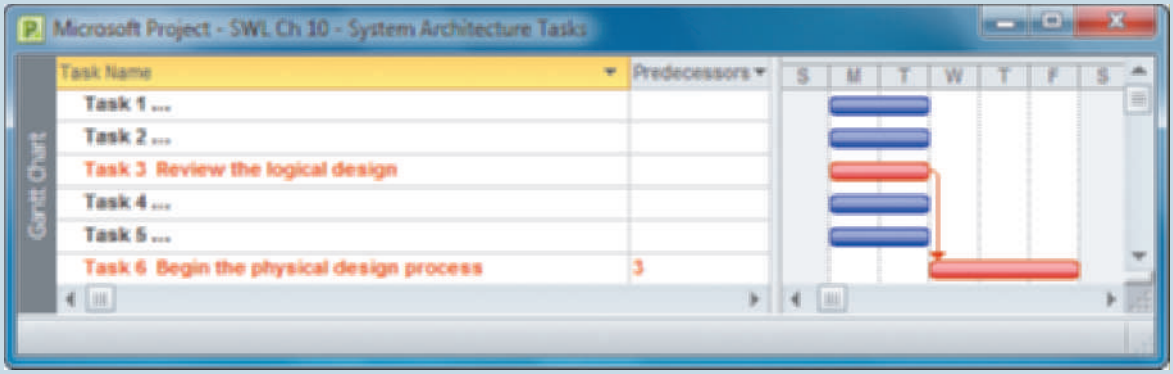


FIGURE 10-48 Tasks 1, 2, 3, 4, and 5 are concurrent tasks that could be performed at the same time. Task 6 is a dependent task that cannot be performed until Task 3 has been completed.

Chapter 3 describes project management tools, techniques, and software. To learn more, you can use the Features section on your Student Study Tool CD-ROM, or visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and locate the project management resources library for this book. On the Web, Microsoft offers demo versions, training, and tips for using Project 2010. You also can visit the OpenWorkbench.org site to learn more about this free, open-source software.

Ready for a Challenge?

In additional to technical skills, IT professionals need critical thinking skills such as perception, organization, analysis, problem-solving, and decision-making. The Ready for a Challenge feature can help you learn, practice, and apply critical thinking skills that you can take to the workplace.

The IT team at Game Technology is working on an overall architecture for the new C3 system. They solicited RFPs from several hardware vendors, and decided to work with Network Illusions, a well-known local firm. Your job is to help analyze the server test results, and to draw network diagrams when requested.

The Network Illusions sales rep recommended a Model DX server for the C3 network, and submitted the following data, which shows projected network response times for various numbers of Game Technology users:

Projected Data: Server Model DX	Number of Users				
	10	20	30	40	50
Response Time (sec)	.01	.015	.02	.025	.03

Practice Tasks

A. Using the data provided, create an XY chart that shows Response Time on the Y (vertical) axis and Number of Users on the X (horizontal) axis. You learned about XY charts in Chapter 2 of your textbook. Describe the results and your interpretation of the data.
B. Draw a bus network with a server, six workstations, a printer, a scanner, and a wireless access point (WAP). Also, determine how many separate data paths are needed in a mesh network with four nodes. Five nodes?

After you complete the Practice Tasks, to check your work and view sample answers, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the resources for this chapter, and locate Ready for a Challenge?.

The Challenge

Although the spec was for a Model XP server, Network Illusions delivered a newer model, called the DX+. The vendor sales rep said that performance would be the same or better, but the IT team decided to run a series of response time tests. The results are as follows:

Actual Data: Server Model DX+	Number of Users				
	10	20	30	40	50
Response Time (sec)	.01	.015	.02	.03	.05

Challenge Tasks

A. Use the actual test data to create another XY chart, similar to the first one. Describe the results and your interpretation of the data.
B. Draw the same network as a star topology with a central switch. Also determine how many separate data paths are needed in a mesh network that has six nodes.

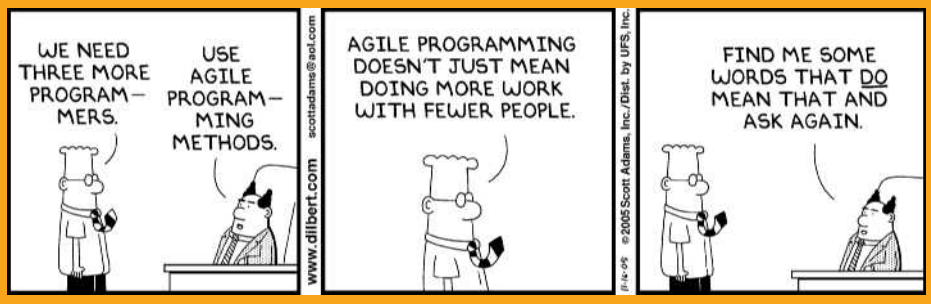
This page intentionally left blank

PHASE 4 SYSTEMS IMPLEMENTATION

DELIVERABLE
A functioning information system

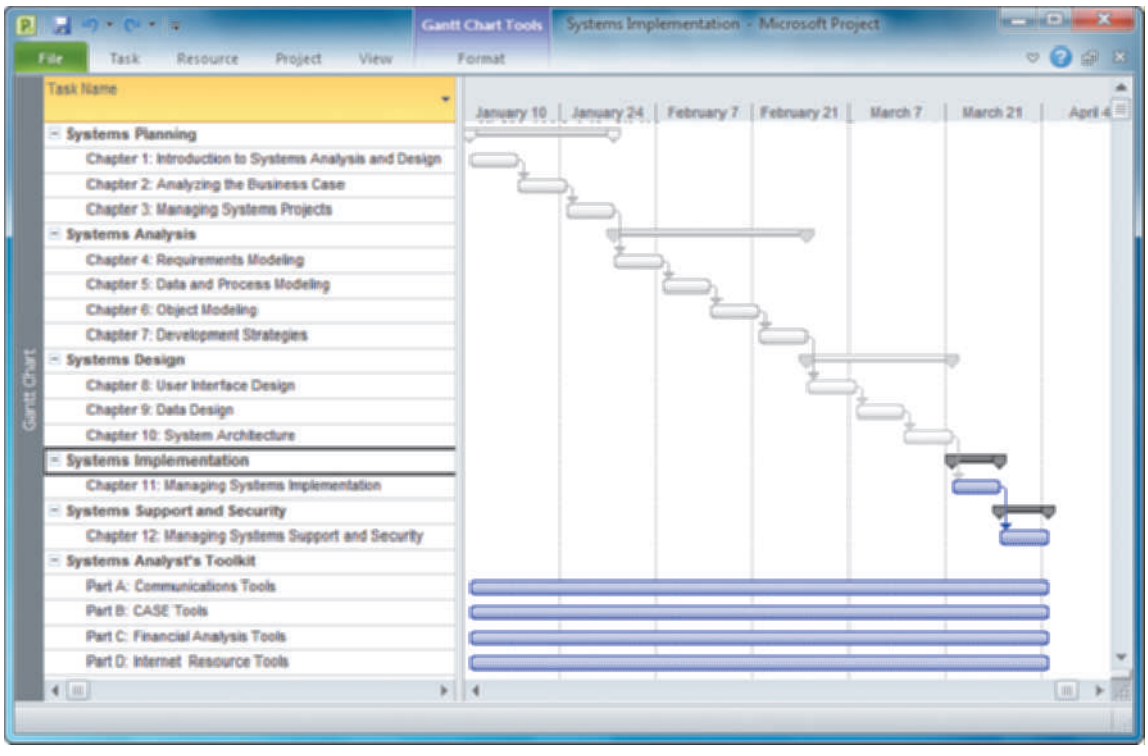
TOOLKIT SUPPORT
Primary tools: Communications and CASE tools

VIDEO LEARNING SESSION
Structure Charts



As the Dilbert cartoon suggests, successful systems implementation requires effective methods, a capable team, and management support. You will learn more about these topics in the systems implementation phase.

Systems implementation is the fourth of five phases in the systems development life cycle. In the previous phase, systems design, you created a physical model of the system. Now you will implement that design. Your tasks will include application development, documentation, testing, training, data conversion, and system changeover. The deliverable for this phase is a completely functioning information system.



CHAPTER

Managing Systems Implementation

Chapter 11 describes the systems implementation phase of the SDLC. This chapter describes application development, installation, and evaluation.

INTRODUCTION

OBJECTIVES

When you finish this chapter, you will be able to:

- Explain the importance of software quality assurance and software engineering
- Describe application development using structured, object-oriented, and agile methods
- Draw a structure chart showing top-down design, modular design, cohesion, and coupling
- Explain the coding process
- Explain unit, integration, and system testing
- Differentiate between program, system, operations, and user documentation
- List the main steps in system installation and evaluation
- Develop training plans for various user groups, compare in-house and outside training, and describe effective training techniques
- Describe data conversion and changeover methods
- Explain post-implementation evaluation and the final report to management

Managing systems implementation involves application development, testing, documentation, training, data conversion, system changeover, and post-implementation evaluation of the results.

During systems implementation, the system design specification serves as a blueprint for constructing the new system. The initial task is application development, which requires systems analysts and programmers to work together to construct the necessary programs and code modules. Before a changeover, the system must be tested and documented carefully, users must be trained, and existing data must be converted. After the new system is operational, a formal evaluation of the results takes place as part of a final report to management.

Chapter Introduction Case: Mountain View College Bookstore

Background: Wendy Lee, manager of college services at Mountain View College, wants a new information system that will improve efficiency and customer service at the three college bookstores.

In this part of the case, Tina Allen (systems analyst) and David Conroe (student intern) are talking about implementation tasks for the new system.

Participants: Wendy, Tina, and David
Location: Wendy Lee's office, Monday morning, February 6, 2012
Project status: The system design specification was approved, and Tina and David are ready to implement the new bookstore information system.
Discussion topics: Implementation tasks, including quality assurance, structure charts, testing, training, data conversion process, system changeover, and post-implementation evaluation

Tina: Good morning, Wendy. We're ready to start the implementation process, and I'd like to go over our plans. David will be assisting me, so I asked him to join us.

Wendy: *I'm glad you did. I met David during the interviews several months ago.*

David: Hi, Wendy, good to see you again. What's next?

Tina: Let's talk about quality assurance. We'll also discuss various implementation options, including agile methods, but we'll continue with a structured approach for now.

Wendy: *Sounds good. What are the major tasks on your list?*

Tina: Well, the biggest task is to translate the design into program code and produce a functioning system. We'll develop structure charts that the programmers can use as blueprints, and David will help me coordinate with the programmers.

David: It will be great to see all the design work finally turn into a functioning system.

Tina: It sure will. Anyway, we'll proceed to do several types of testing, and we'll document everything we do. When we're ready, we'll put the new system into what's called a test environment until we're ready to go online with the operational environment.

Wendy: *What about training?*

Tina: We'll consider several kinds of training — from vendors or we might do our own.

Wendy: *Then what?*

Tina: The final steps will be data conversion and system changeover. After the new system is up and running, we'll schedule a formal evaluation and submit a final report. Here's a task list to get us started:

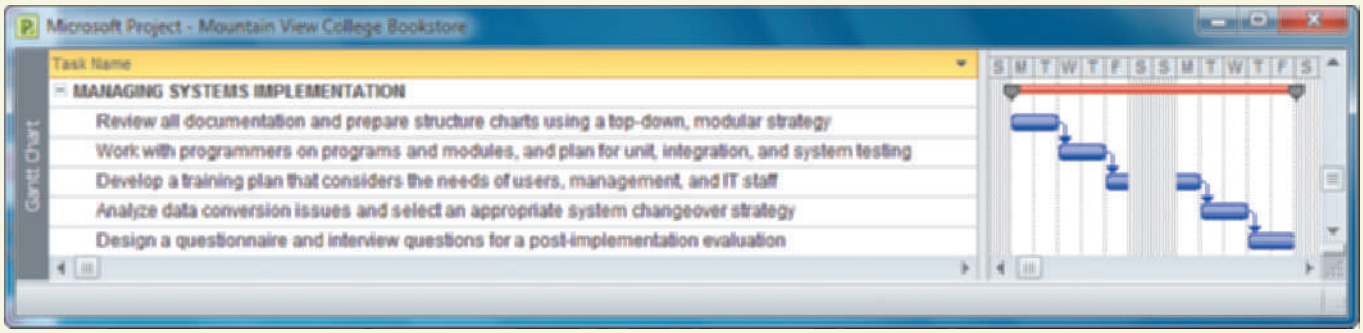


FIGURE 11-1 Typical systems implementation task list.

ON THE WEB

To learn more about software engineering, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Software Engineering link.

Software Quality Assurance

In today's competitive business environment, companies are intensely concerned with the quality of their products and services. A successful organization must improve quality in every area, including its information systems. Top management must provide the leadership, encouragement, and support needed for high-quality IT resources.

No matter how carefully a system is designed and implemented, problems can occur. Rigorous testing can detect errors during implementation, but it is much less expensive to correct mistakes earlier in the development process. The main objective of **quality assurance** is to avoid problems or to identify them as soon as possible. Poor quality can result from inaccurate requirements, design problems, coding errors, faulty documentation, and ineffective testing.

To improve the finished product, software systems developers should consider software engineering and internationally recognized quality standards.

Software Engineering

Because quality is so important, you can use an approach called software engineering to manage and improve the quality of the finished system. **Software engineering** is a software development process that stresses solid design, accurate documentation, and careful testing.

The Web site for the Software Engineering Institute (SEI) at Carnegie Mellon University is shown in Figure 11-2. SEI is a leader in software engineering and provides quality standards and suggested procedures for software developers and systems analysts. SEI's primary objective is to find better, faster, and less-expensive methods of software development. To achieve that goal, SEI designed a set of software development standards called the **Capability Maturity Model (CMM)®**, which has been used successfully by thousands of organizations around the globe. The purpose of the model, which was introduced in 1991, was to improve software quality, reduce development time, and cut costs. More recently, SEI established a new model, called **Capability Maturity Model Integration (CMMI)®**, that integrates software and systems development into a much

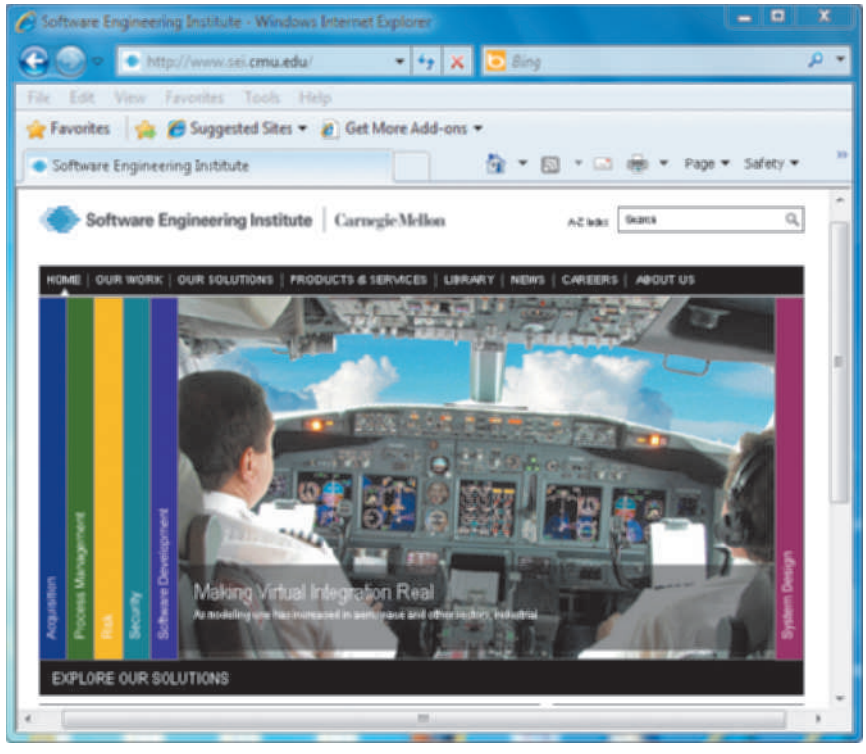


FIGURE 11-2 The Software Engineering Institute represents the cutting edge of software design and development technology.

larger framework called **process improvement**. The CMMI® regards software as part of a larger quality improvement process, rather than as an end in itself. The CMMI® tracks an organization's processes, using five maturity levels, from Level 1, which is referred to as unpredictable, poorly controlled, and reactive, to Level 5, in which the optimal result is process improvement. The five maturity levels are shown in Figure 11-3.

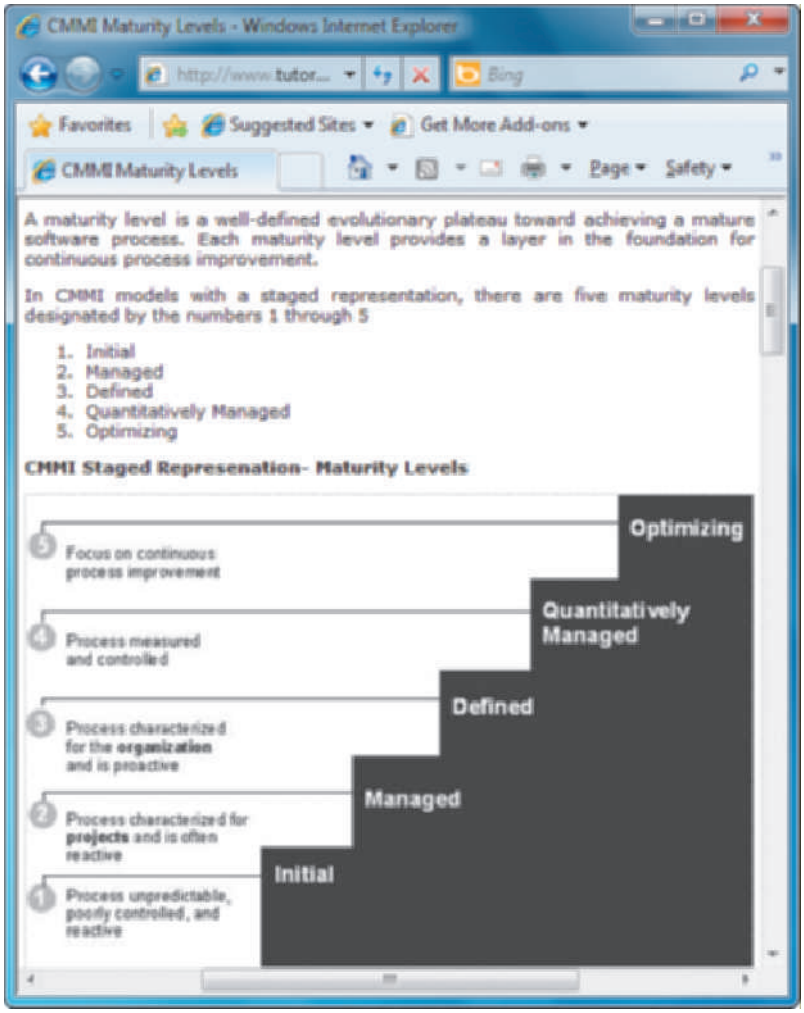


FIGURE 11-3 The CMMI® includes five maturity levels, from Level 1, which is referred to as unpredictable, poorly controlled, and reactive, to Level 5, in which the optimal result is process improvement.

International Organization for Standardization (ISO)

You learned in Chapter 9 that the International Organization for Standardization (ISO) is a worldwide body that establishes quality standards for products and services, as shown in Figure 11-4 on the next page. ISO standards include everything from internationally recognized symbols, such as those shown in Figure 11-5, to the ISBN numbering system that identifies this textbook. In addition, ISO seeks to offer a global consensus of what constitutes good management practices — practices that can help firms deliver consistently high-quality products and services.

Because software is so important to a company's success, many firms seek assurance that software systems, either purchased or developed in-house, will meet rigid quality standards. In 1991, ISO established a set of guidelines called ISO 9000-3, which provided a quality assurance framework for developing and maintaining software.

FIGURE 11-4 The International Organization for Standardization (ISO) is an international body that establishes standards for many products and services, including software development. ISO states that standards, which provide product quality, compatibility, and safety, often are taken for granted, and noticed only when they are absent.

ON THE WEB

To learn more about ISO, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the ISO link.

FIGURE 11-5 ISO standards include internationally recognized symbols.

The standard was updated in 2004 and is now referred to as **ISO 90003:2004**. A company can specify ISO standards when it purchases software from a supplier or use ISO guidelines for in-house software development to ensure that the final result measures up to ISO standards. ISO requires a specific development plan, which outlines a step-by-step process for transforming user requirements into a finished product. ISO standards can be quite detailed. For example, ISO requires that a software supplier document all testing and maintain records of test results. If problems are found, they must be resolved, and any modules affected must be retested. Additionally, software and hardware specifications of all test equipment must be documented and included in the test records.

Overview of Application Development

Application development is the process of constructing the programs and code modules that serve as the building blocks of the information system. In Chapter 1, you learned that structured analysis, object-oriented (O-O) analysis, and agile methods are three popular development options. Regardless of the method, the objective is to translate the design into program and code modules that will function properly. Because systems implementation usually is very labor-intensive, developers often use project management tools and techniques to control schedules and budgets.

ON THE WEB

To learn more about application development, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Application Development link.

Review the System Design

At this point, it might be helpful to review the tasks involved in the creation of the system design.

- In Chapter 4, you learned about requirements modeling and how to use functional decomposition diagrams (FDDs) to break complex business operations down into smaller units, or functions.
- In Chapter 5, you learned about structured data and process modeling, and you created data flow diagrams (DFDs). You also developed process descriptions for functional primitive processes that documented the business logic and processing requirements.
- In Chapter 6, you developed an object-oriented model of the new system that included use case diagrams, class diagrams, sequence diagrams, state transition diagrams, and activity diagrams.
- In Chapter 7, you selected a development strategy.
- In Chapter 8, you designed the user interface.
- In Chapter 9, you worked with data design issues, analyzed relationships between system entities, and constructed entity-relationship diagrams (ERDs).
- In Chapter 10, you considered an overall system architecture.

Taken together, this set of tasks produced an overall design and a plan for physical implementation.

Application Development Tasks

If you used traditional structured or object-oriented (O-O) methods, you now are ready to translate the design into a functioning application. If you selected an agile development method, you will plan the project, lay the groundwork, assemble the team, and prepare to interact with the customers.

TRADITIONAL METHODS Building a new system requires careful planning. After an overall strategy is established, individual modules must be designed, coded, tested, and documented. A **module** consists of related program code organized into small units that are easy to understand and maintain. After the modules are developed and tested individually, more testing takes place, along with thorough documentation of the entire system, as shown in Figure 11-6 on the next page.

When you create program modules using structured or object-oriented methods, you start by reviewing documentation from prior SDLC phases and creating a set of program designs. If you built a documentation file early in the development process and updated it regularly, you now have a valuable repository of information. The centerpiece of your documentation is the system design specification, accompanied by diagrams, source documents, screen layouts, report designs, data dictionary entries, and user comments. If you used a CASE tool during the systems analysis and design process, your job will be much

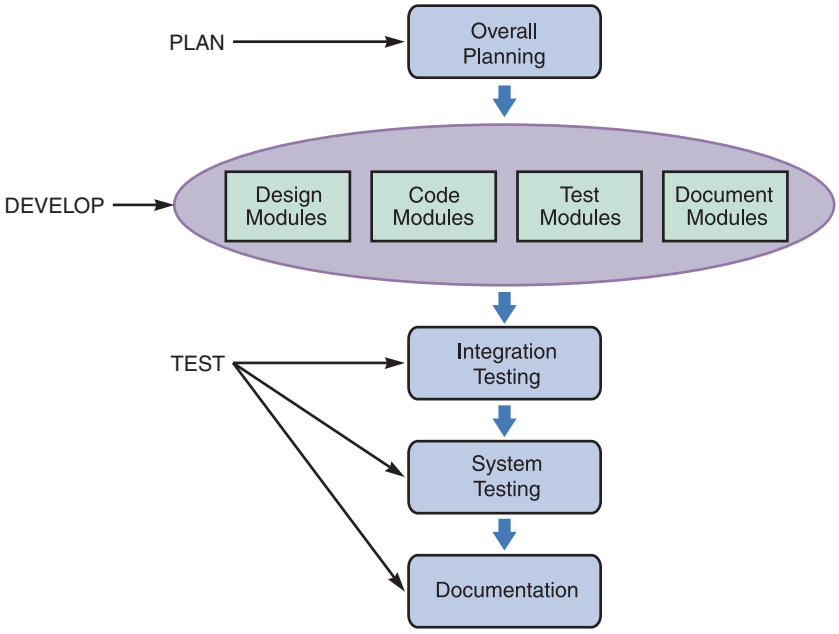


FIGURE 11-6 The main steps in application development.

easier. At this point, coding and testing tasks begin. Although programmers typically perform the actual coding, IT managers usually assign systems analysts to work with them as a team.

AGILE METHODS If you decided to use an agile approach, intense communication and collaboration will now begin between the IT team and the users or customers. The objective is to create the system through an iterative process of planning, designing, coding, and testing. Agile projects use various models, including the spiral model shown in Figure 1-30 on page 26, or the Extreme Programming (XP) example shown in Figure 11-7. Agile development and XP are discussed in detail later in this chapter.

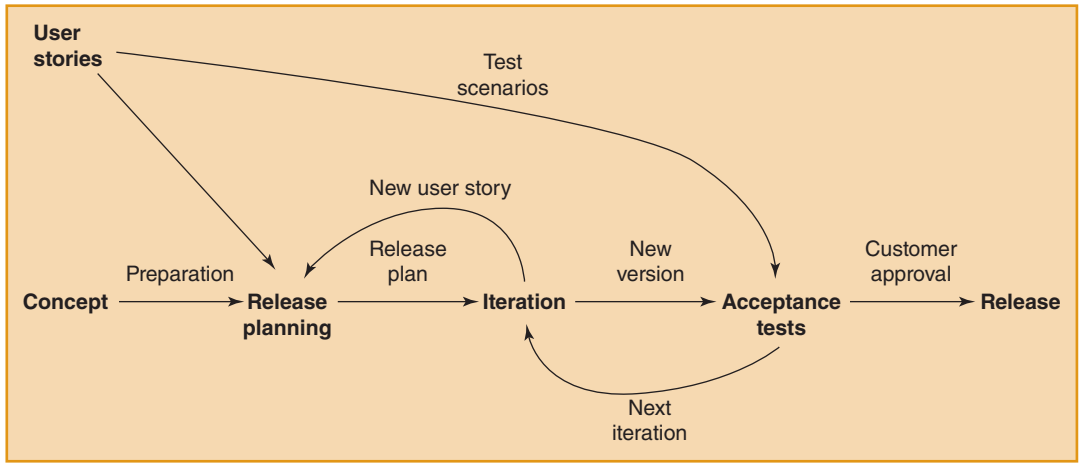


FIGURE 11-7 Simplified model of an Extreme Programming (XP) project. Note the emphasis on iteration and testing.

Systems Development Tools

Each systems development approach has its own set of tools that has worked well for that method. For example, structured development relies heavily on DFDs and structure charts; object-oriented methods use a variety of diagrams, including use case, class, sequence, and transition state diagrams; and agile methods tend to use spiral or other iterative models such as the example in Figure 11-7.

System developers also can use multipurpose tools to help them translate the system logic into properly functioning program modules. These generic tools include entity-relationship diagrams, flowcharts, pseudocode, decision tables, and decision trees.

ENTITY-RELATIONSHIP DIAGRAMS During data design, in Chapter 9, you learned how to use entity-relationship diagrams to show the interaction among system entities and objects. An ERD is a useful tool regardless of which methodology you are using, because the various relationships (one-to-one, one-to-many, and many-to-many) must be understood and implemented in the application development process.

FLOWCHARTS As you learned in Chapter 5, flowcharts can be used to describe program logic, and are very useful in visualizing a modular design. A **flowchart** represents logical rules and interaction graphically, using a series of symbols connected by arrows. Using flowcharts, programmers can break large systems into subsystems and modules that are easier to understand and code.

PSEUDOCODE **Pseudocode** is a technique for representing program logic. Pseudocode is similar to structured English, which was explained in Chapter 5. Pseudocode is not language-specific, so you can use it to describe a software module in plain English without requiring strict syntax rules. Using pseudocode, a systems analyst or a programmer can describe program actions that can be implemented in any programming language. Figure 11-8 illustrates an example of pseudocode that documents a sales promotion policy.

ON THE WEB

To learn more about pseudocode, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Pseudocode link.

SAMPLE OF A SALES PROMOTION POLICY

- Preferred customers who order more than $1,000 are entitled to a 5% discount, and an additional 5% discount if they used our charge card.
- Preferred customers who do not order more than $1,000 receive a $25 bonus coupon.
- All other customers receive a $5 bonus coupon.

PSEUDOCODE VERSION OF THE SALES PROMOTION POLICY

```
IF customer is a preferred customer, and
        IF customer orders more than $1,000 then
                Apply a 5% discount, and
                IF customer uses our charge card, then
                        Apply an additional 5% discount
        ELSE
                        Award a $25 bonus coupon
ELSE
        Award a $5 bonus coupon
```

FIGURE 11-8 Sample of a sales promotion policy with logical rules, and a pseudocode version of the policy. Notice the alignment and indentation of the logic statements.

DECISION TABLES AND DECISION TREES As you learned in Chapter 5, decision tables and decision trees can be used to model business logic for an information system. In addition to being used as modeling tools, analysts and programmers can use decision tables and decision trees during system development, as they develop code modules that implement the logical rules. Figure 11-9 shows an example of a decision tree that documents the sales promotion policy shown in Figure 11-8. Notice that the decision tree accurately reflects the sales promotion policy, which has three separate conditions, and four possible outcomes.

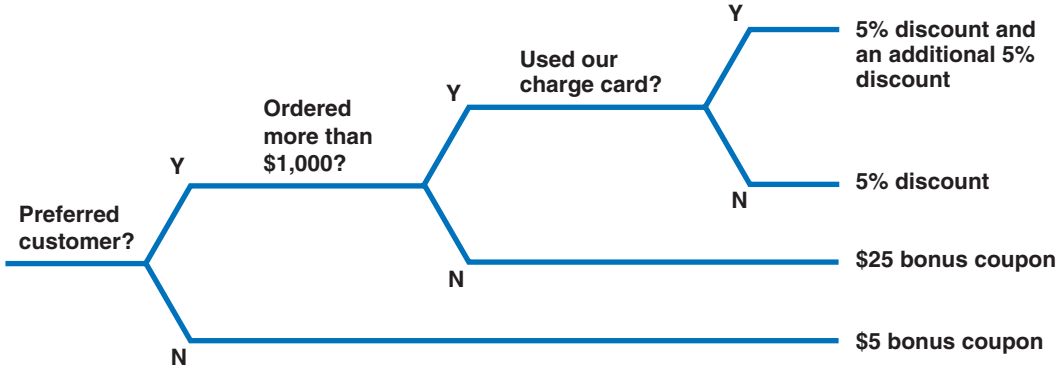


FIGURE 11-9 Sample decision tree that reflects the sales promotion policy in Figure 11-8. Like a decision table, a decision tree shows the action to be taken based on certain conditions.

Project Management

Regardless of whether structured analysis, object-oriented design, or agile methods are used, even a modest-sized project might have hundreds or even thousands of modules. For this reason, application development can become quite complex and difficult to manage. At this stage, project management is especially important. Users and managers are looking forward to the new system, and it is very important to set realistic schedules, meet project deadlines, control costs, and maintain quality. To achieve these goals, the systems analyst or project manager should use project management tools and techniques similar to those described in Chapter 3 to monitor and control the development effort.

The following sections describe the application development process. Structured development techniques and tools are discussed first, followed by object-oriented and agile development methods.

VIDEO LEARNING SESSION: STRUCTURE CHARTS

Video Learning Sessions can help you understand key concepts, practice your skills, and check your work. To access the sessions, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and navigate to the **Video Learning Sessions** for this book. This session is about structure charts. You'll learn about structure charts and why they are important modeling tools, how to draw structure charts, and how you can use a CASE tool to create structure charts.

STRUCTURED APPLICATION DEVELOPMENT

Structured application development usually involves a **top-down approach**, which proceeds from a general design to a detailed structure. After a systems analyst documents the system's requirements, he or she breaks the system down into subsystems and modules in a process called **partitioning**. This approach also is called **modular design** and is similar to constructing a leveled set of DFDs. By assigning modules to different programmers, several development areas can proceed at the same time. As explained in Chapter 3, you can use project management software to monitor work on each module, forecast overall development time, estimate required human and technical resources, and calculate a critical path for the project.

Because all the modules must work together properly, an analyst must proceed carefully, with constant input from programmers and IT management to achieve a sound, well-integrated structure. The analyst also must ensure that integration capability is built into each design and thoroughly tested.

Structure Charts

Structure charts show the program modules and the relationships among them. A **structure chart** consists of rectangles that represent the program modules, with arrows and other symbols that provide additional information. Typically, a higher-level module, called a **control module**, directs lower-level modules, called **subordinate modules**. In a structure chart, symbols represent various actions or conditions. Structure chart symbols represent modules, data couples, control couples, conditions, and loops.

MODULE A rectangle represents a module, as shown in Figure 11-10. Vertical lines at the edges of a rectangle indicate that module 1.3 is a library module. A **library module** is reusable code and can be invoked from more than one point in the chart.

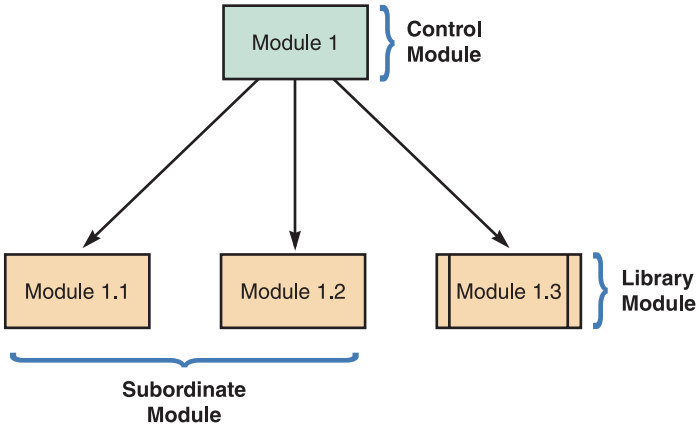


FIGURE 11-10 An example of structure chart modules.

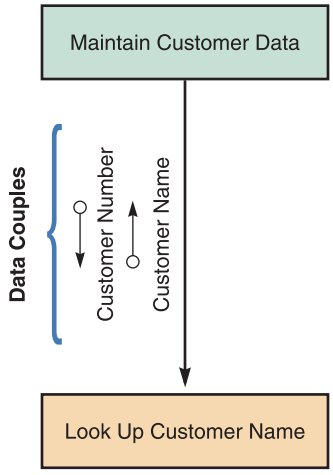


FIGURE 11-11 An example of a structure chart data.

DATA COUPLE An arrow with an empty circle represents a data couple. A **data couple** shows data that one module passes to another. In the data couple example shown in Figure 11-11, the *Look Up Customer Name* module exchanges data with the *Maintain Customer Data* module.

CONTROL COUPLE An arrow with a filled circle represents a control couple. A **control couple** shows a message, also called a **status flag**, which one module sends to another. In the example shown in Figure 11-12, the *Update Customer File* module sends an *Account Overdue* flag back to the *Maintain Customer Data* module. A module uses a flag to signal a specific condition or action to another module.

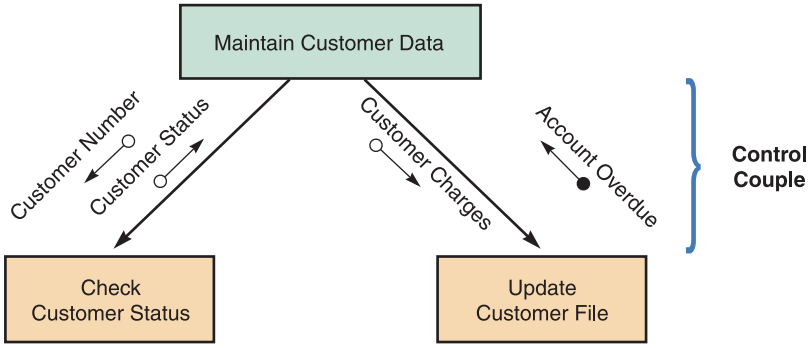


FIGURE 11-12 An example of a structure chart control couple.

CONDITION A line with a diamond on one end represents a condition. A **condition** line indicates that a control module determines which subordinate modules will be invoked, depending on a specific condition. In the example shown in Figure 11-13, *Sort Inventory Parts* is a control module with a condition line that triggers one of the three subordinate modules.

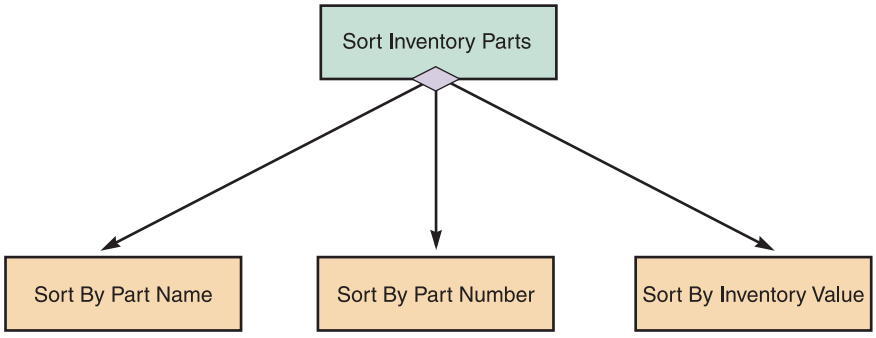


FIGURE 11-13 The diagram shows a control module that triggers three subordinate modules.

LOOP A curved arrow represents a loop. A **loop** indicates that one or more modules are repeated. In the example shown in Figure 11-14 on the next page, the *Get Student Grades* and *Calculate GPA* modules are repeated.

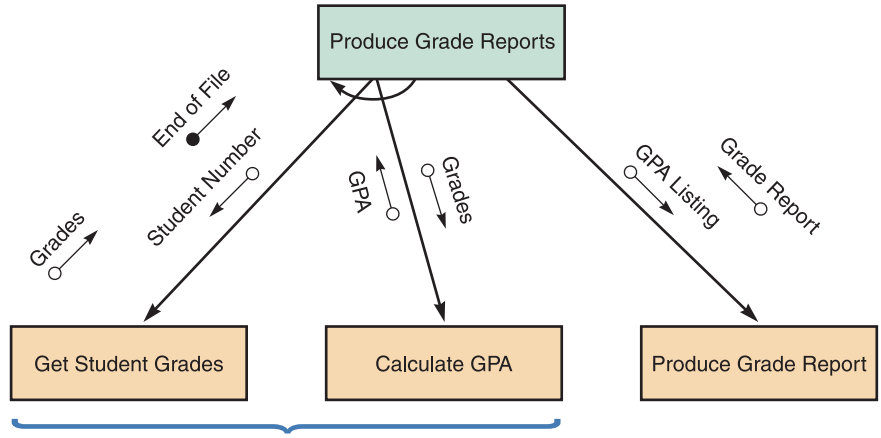


FIGURE 11-14 The diagram shows a structure chart loop with two repeating modules.

Cohesion and Coupling

Cohesion and coupling are important tools for evaluating the overall design. As explained in the following sections, it is desirable to have modules that are highly cohesive and loosely coupled.

Cohesion measures a module's scope and processing characteristics. A module that performs a single function or task has a high degree of cohesion, which is desirable. Because it focuses on a single task, a cohesive module is much easier to code and reuse. For example, a module named *Verify Customer Number* is more cohesive than a module named *Calculate and Print Statements*. If you notice the word *and* in a module name, you know that more than one task is involved.

If a module must perform multiple tasks, more complex coding is required and the module will be more difficult to create and maintain. If you need to make a module more cohesive, you can split it into separate units, each with a single function. For example, by splitting the module *Check Customer Number and Credit Limit* in Figure 11-15 into two separate modules, *Check Customer Number* and *Check Customer Credit Limit*, cohesion is greatly improved.

Coupling describes the degree of interdependence among modules. Modules that are independent are **loosely coupled**, which is desirable. Loosely coupled modules are easier to maintain and modify, because the logic in one module does not affect other modules. If a programmer needs to update a loosely coupled module, he or she can accomplish the task in a single location. If modules are **tightly coupled**, one module is linked to internal logic contained in another module. For example, Module A might refer to an internal variable contained in Module B. In that case, a logic error in the Module B will affect the processing in Module A. For that reason, passing a status flag down as a message from a control module is generally regarded as poor design. It is better to have subordinate modules handle processing tasks as independently as possible, to avoid a cascade effect of logic errors in the control module.

In Figure 11-16, the tightly coupled example on the left shows that the subordinate module *Calculate Current Charges* depends on a status flag sent down from the control module *Update Customer Balance*. It would be preferable to have the modules loosely coupled and logically independent. In the example on the right, a status flag is not needed because the subordinate module *Apply Discount* handles discount processing independently. Any logic errors are confined to a single location: the *Apply Discount* module.

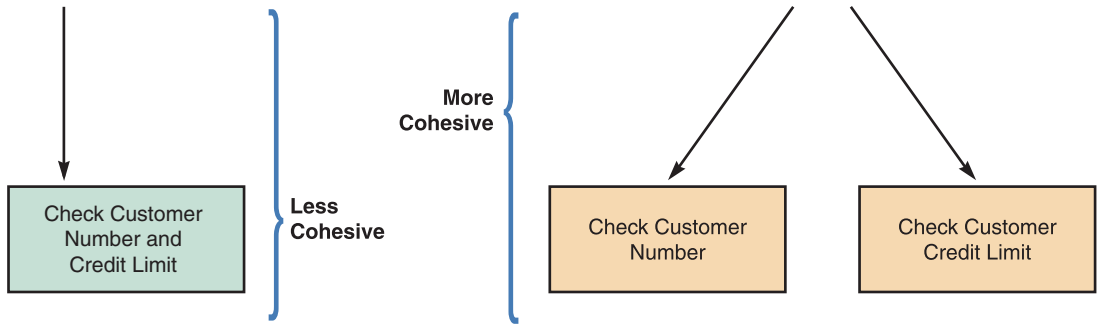


FIGURE 11-15 Two examples of cohesion. Notice that the single module on the left is less cohesive than the two modules on the right.

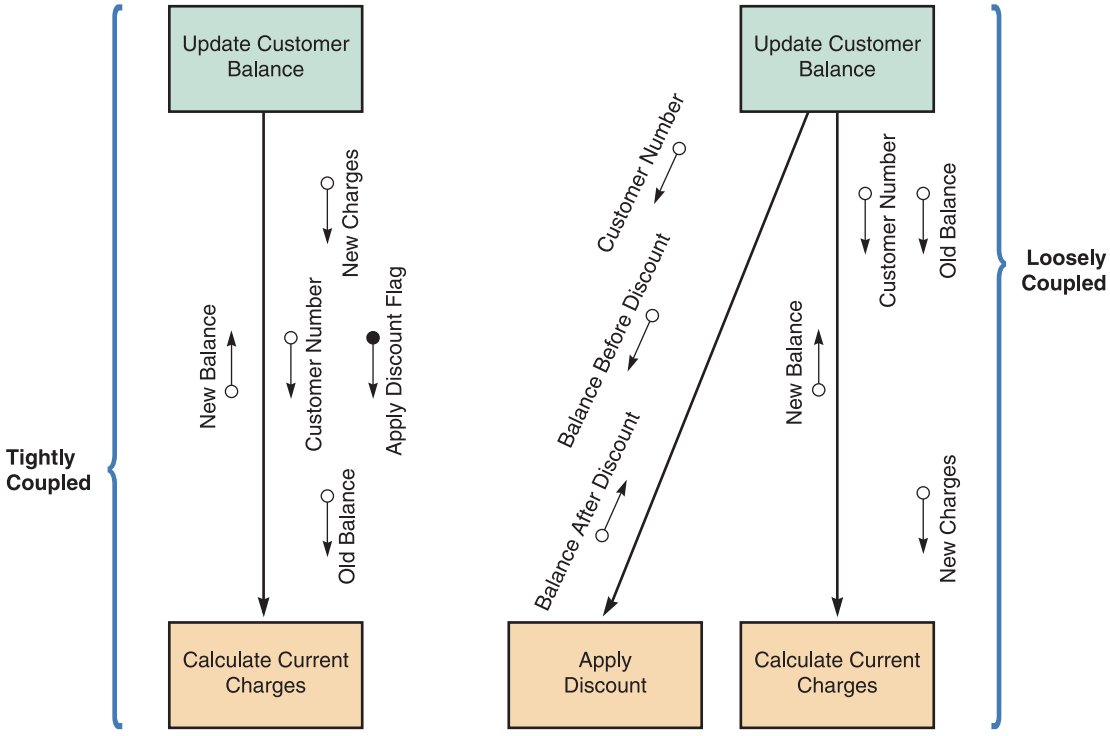


FIGURE 11-16 An example of tightly coupled and loosely coupled structure charts.

Drawing a Structure Chart

If you used a structured analysis method, your structure charts will be based on the DFDs you created during data and process modeling.

Typically, you follow four steps when you create a structure chart. You review DFDs to identify the processes and methods, identify the program modules and determine control-subordinate relationships, add symbols for couples and loops, and analyze the structure chart to ensure that it is consistent with your system documentation.

STEP 1: REVIEW THE DFDS Your first step is to review all DFDs for accuracy and completeness, especially if changes have occurred since the systems analysis phase. If object models also were developed, you should analyze them to identify the objects, the methods that each object must perform, and the relationships among the objects. A method is similar to a functional primitive, and requires code to implement the necessary actions.

STEP 2: IDENTIFY MODULES AND RELATIONSHIPS Working from the logical model, you transform functional primitives or object methods into program modules. When analyzing a set of DFDs, remember that each DFD level represents a processing level. If you are using DFDs, you would work your way down from the context diagram to the lower-level diagrams, identifying control modules and subordinate modules, until you reach functional primitives. If more cohesion is desired, you can divide processes into smaller modules that handle a single task. Figure 11-17 on the next page shows a structure chart based on the order system shown in Figures 5-16, 5-17, and 5-18 on pages 212–214. Notice how the three-level structure chart relates to the three DFD levels.

STEP 3: ADD COUPLES, LOOPS, AND CONDITIONS Next, you add couples, loops, and conditions to the structure chart. If you are working with DFDs, you can review the data flows and the data dictionary to identify the data elements that pass from one module to another. In addition to adding the data couples, you add control couples where a module is sending a control parameter, or flag, to another module. You also add loops and condition

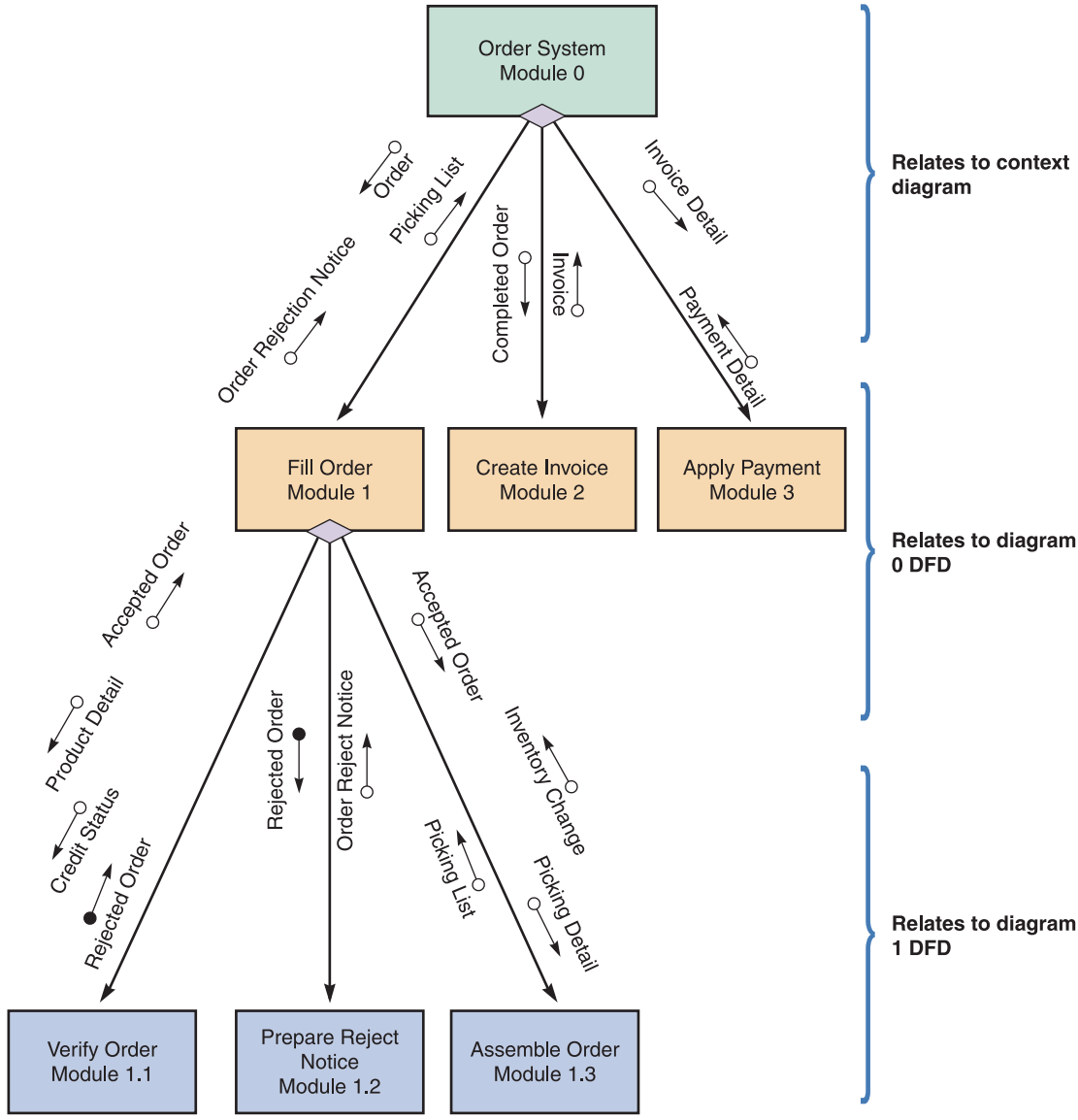


FIGURE 11-17 A structure chart based on the order system DFDs on pages 212–214. The three-level structure chart relates to the three DFD levels.

lines that indicate repetitive or alternative processing steps, as shown in Figure 11-17. If you also developed an object model, you can review the class diagrams and object relationship diagrams to be sure that you understand the interaction among the objects.

STEP 4: ANALYZE THE STRUCTURE CHART AND THE DATA DICTIONARY At this point, the structure chart is ready for careful analysis. You should check each process, data element, or object method to ensure that the chart reflects all previous documentation and that the logic is correct. You also should determine that modules are strongly cohesive and loosely coupled. Often, you must draw several versions of the chart. Some CASE tools can help you analyze the chart and identify problem areas.

Object-Oriented Application Development

When you studied the object-oriented methods described in Chapter 6, you learned that O-O analysis makes it easier to translate an object model directly into an object-oriented programming language. This process is called **object-oriented development**, or **OOD**. Although

many structured design concepts also apply to object-oriented methodology, there are some differences.

Characteristics of Object-Oriented Application Development

When implementing a structured design, a structure chart is used to describe the interaction between program modules, as explained earlier. In contrast, when implementing an object-oriented design, relationships between objects already exist. Because object interaction is defined during the O-O analysis process, the application's structure is represented by the object model itself.

As Chapter 6 explains, objects contain both data and program logic, called methods. Individual object instances belong to classes of objects with similar characteristics. The relationship and interaction among classes are described using a class diagram, such as the one shown in Figure 11-18. A class diagram includes the class **attributes**, which describe the characteristics of objects in the class, and **methods**, which represent program logic. For example, the Customer class describes customer objects. Customer attributes include Number, Name, Address, and so on. Methods for the Customer class include Place order, Modify order, and Pay invoice, among others. The Customer class can exchange messages with the Order class.

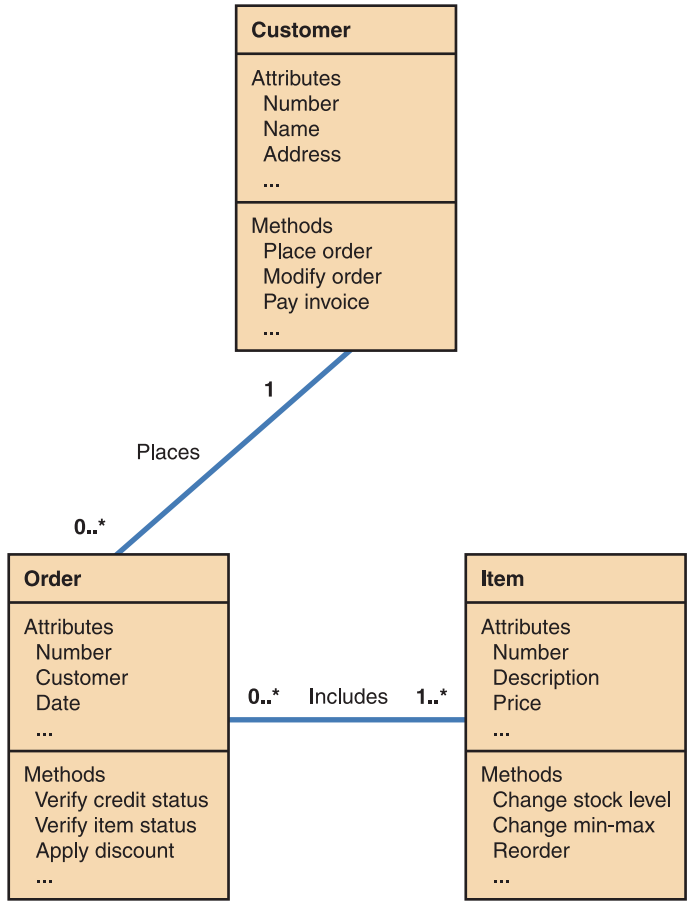


FIGURE 11-18 A simplified class diagram for a customer order processing system.

In addition to class diagrams, programmers get an overview of object interaction by using object relationship diagrams that were developed during the O-O analysis process. For example, Figure 11-19 shows an object relationship diagram for a fitness center. Notice that the model shows the objects and how they interact to perform business functions and transactions.

Properly implemented, object-oriented development can speed up projects, reduce costs, and improve overall quality. However, these results are not always achieved. Organizations sometimes have unrealistic expectations, and do not spend enough time learning about, preparing for, and implementing the OOD process. For example, no one would build a bridge without a analysis of needs, supporting data, and a detailed blueprint — and the bridge would not be opened for traffic until it had been carefully inspected and checked to ensure that all specifications were met. O-O software developers sometimes forget that the basic rules of architecture also apply to their projects.

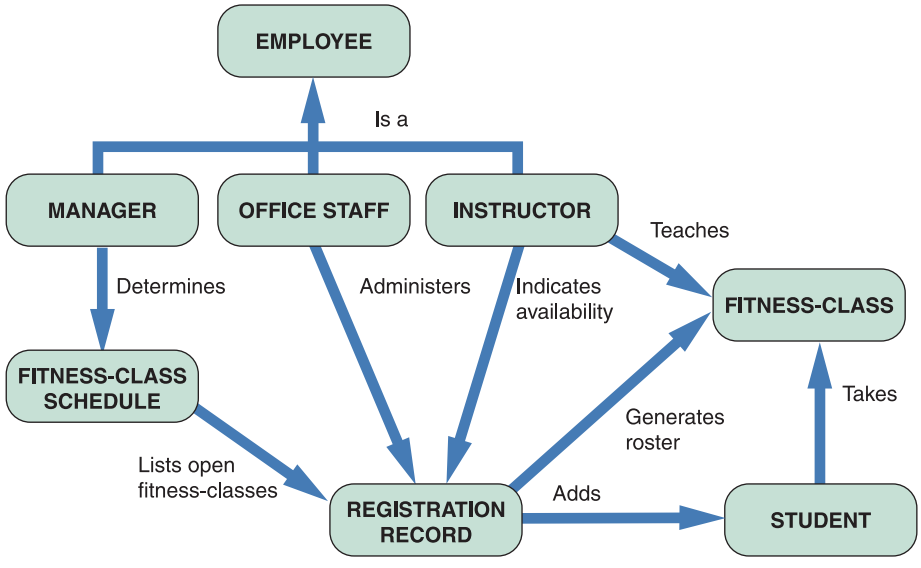


FIGURE 11-19 An object-relationship diagram for a fitness center.

In summary, to secure the potential benefits of object-oriented development, systems analysts must carefully analyze, design, implement, test, and document their O-O projects.

Implementation of Object-Oriented Designs

When a programmer translates an object-oriented design into an application, he or she analyzes the classes, attributes, methods, and messages that are documented in the object model. During this process, the programmer makes necessary revisions and updates to class diagrams, sequence diagrams, state transition diagrams, and activity diagrams.

The programmer's main objective is to translate object methods into program code modules and determine what event or message will trigger the execution of each module. To accomplish the task, the programmer analyzes sequence diagrams and state transition diagrams that show the events and messages that trigger changes to an object. O-O applications are called event-driven, because each event, transaction, or message triggers a corresponding action. The programmer can represent the program steps in pseudocode initially, or use CASE tools and code generators to create object-oriented code directly from the object model.

Object-Oriented Cohesion and Coupling

The principles of cohesion and coupling also apply to object-oriented application development. Classes should be as loosely coupled (independent of other classes) as possible. In addition, an object's methods also should be loosely coupled (independent of other methods) and highly cohesive (perform closely related actions). By following these principles, classes and objects are easier to understand and edit. O-O programmers who ignore cohesion and coupling concepts may end up creating a web of code that is difficult to maintain. When code is scattered in various places, editing becomes complicated and expensive.

Agile Application Development

As you learned in Chapter 1, agile development is a distinctly different systems development method. It shares many of the steps found in traditional development, but uses a highly iterative process. The development team is in constant communication with the primary user, who is called the **customer**, shaping and forming the system to match the customer's specifications. Agile development is aptly named because it is based on a quick and nimble development process that easily adapts to change. Agile development focuses on small teams, intense communication, and rapid development iterations.

As agile methods become more popular, many software firms are offering packages that teams can use to manage and document the agile process. For example, as shown in Figure 11-20, Serena claims that its Agile on Demand software can help teams be more effective in dealing with multiple stakeholders. The figure also notes the well-known agile development parable about pigs, who, like software developers, are totally committed to writing and delivering software, and chickens, who, like some stakeholders, are involved in the process but not totally committed to it.

You also learned in Chapter 1 about Extreme Programming (XP), which is one of the newest agile methods. XP is an iterative approach, as shown in Figure 11-21 on page 522, where a team of users and developers immerse themselves in systems development. XP supporters emphasize values such as simplicity, communication, feedback, respect, and courage. Success requires strong commitment to the process, corporate support, and dedicated team members. The following section describes a typical XP project.

ON THE WEB

To learn more about extreme programming (XP), visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Extreme Programming (XP) link.

An Extreme Programming (XP) Example

Suppose that a customer has requested a sales tracking system. The first step in the XP process, like any other development method, would be to define the system requirements.

FIGURE 11-20 Serena offers agile development software, and a retelling of the well-known pigs and chickens story.

The customer begins by meeting with programmers and providing user stories. A **user story** is a short, simple requirements definition. Programmers use user stories to determine the project's requirements, priorities, and scope.

In our example, suppose we have the following user stories:

- *As the sales manager, I want to identify fast or slow moving items so I can manage our inventory more effectively.*
- *As a store manager, I need enough lead time to replenish my stock so I don't run out of hot items.*
- *As a sales representative, I want to offer the best selection of fast selling items and clear out the old stock that is not moving.*

User stories do not deal with technical details and are so short that they are often written on index cards. Each user story is given a priority by the customer, so the requirements can be ranked. In addition, programmers assign a score to each user story that indicates the estimated difficulty of implementation. This information helps the team form a plan and assign its resources. Projects are often composed of many user stories, from which programmers can estimate the scope, time requirements, and difficulty of the project. In addition to the user stories, frequent face-to-face meetings with customers provide a higher level of detail as the project progresses.

The team must also develop a **release plan**, which specifies when user stories will be implemented and the timing of the releases. Releases are relatively frequent, and each system release is like a prototype that can be tested and modified as needed.

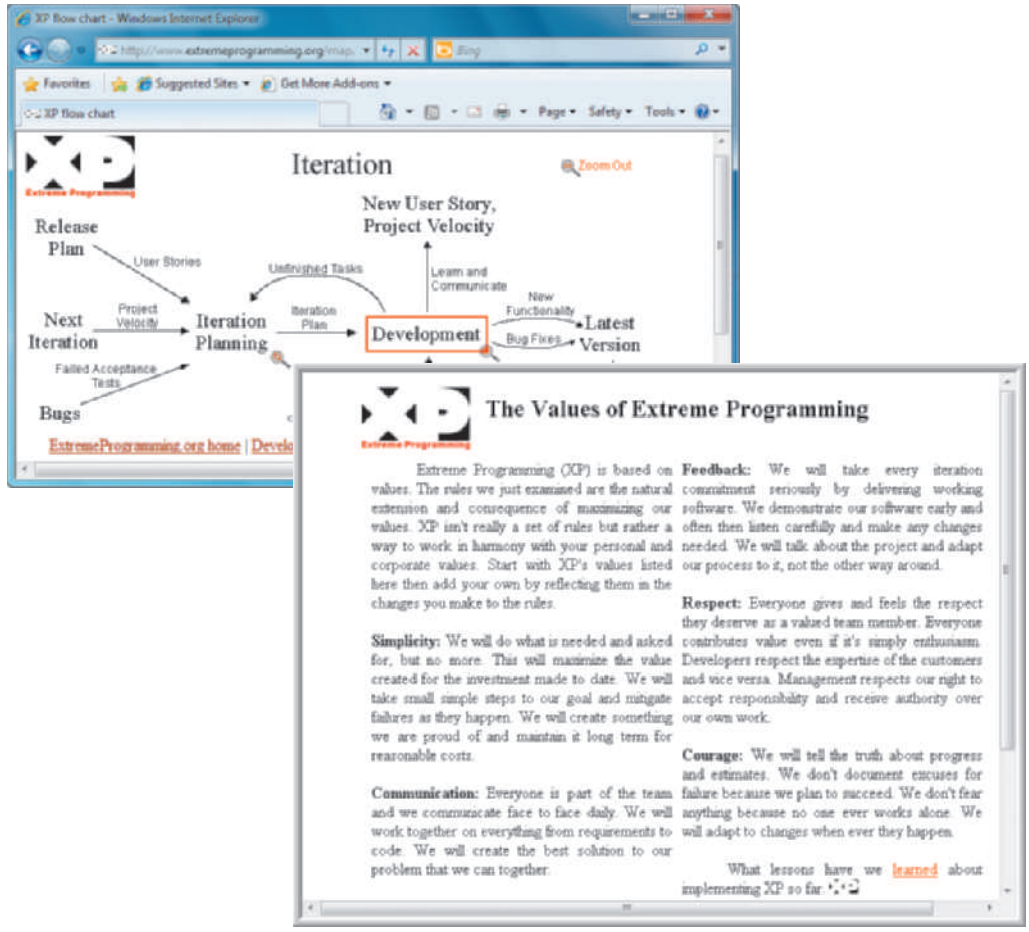


FIGURE 11-21 Extreme Programming is based on iteration, values, and a different approach to systems development.

User stories are implemented in a series of iteration cycles. An **iteration cycle** includes planning, designing, coding, and testing of one or more features based on user stories. At the beginning of each iteration cycle, which is often two weeks long, the team holds an **iteration planning meeting** to break down the user stories into specific tasks that are assigned to team members. As new user stories or features are added, the team reviews and modifies the release plan.

As with any development process, success is determined by the customer's approval. The programming team regularly meets with the customer, who tests prototype releases as they become available. This process usually results in additional user stories, and changes are implemented in the next iteration cycle. As the project's code changes during each iteration, obsolete code is removed and remaining code is restructured to keep the system up to date. The iteration cycles continue until all user stories have been implemented, tested, and accepted.

Extreme Programming uses an interesting concept called parallel programming. In **parallel programming**, two programmers work on the same task on the same computer; one drives (programs) while the other navigates (watches). The onlooker examines the code strategically to see the *forest* while the driver is concerned with the individual *trees* immediately in front of him or her. The two discuss their ideas continuously throughout the process.

Another important concept in XP is that unit tests are designed *before* code is written. This **test-driven design** focuses on end results from the beginning and prevents programmers from straying from their goals. Because of the magnitude and intensity of

the multicycle process, agile testing relies heavily on automated testing methods and software.

Programmers can use popular agile-friendly languages such as Python, Ruby, and Perl. However, agile methods do not require a specific programming language, and programmers also use various object-oriented languages such as Java, C++, and C#.

The Future of Agile Development

Agile methodology is becoming very popular for software projects. Its supporters boast that it speeds up software development and delivers precisely what the customer wants, when the customer wants it, while fostering teamwork and empowering employees. However, there are drawbacks to this adaptive rather than predictive method. Critics of agile development often claim that because it focuses on quick iterations and fast releases, it lacks discipline and produces systems of questionable quality. In addition, agile methodology may not work as well for larger projects because of their complexity and the lack of focus on a well-defined end product.

Before implementing agile development, the proposed system and development methods should be examined carefully. As experienced IT professionals know, a one-size-fits-all solution does not exist. For more information on agile methods, refer to the discussion of systems development methods that begins on page 21 in Chapter 1.

ON THE WEB

To learn more about the future of agile development, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Future of Agile Development link.

CODING

Coding is the process of turning program logic into specific instructions that the computer system can execute. Working from a specific design, a programmer uses a programming language to transform program logic into code statements. An individual programmer might create a small program, while larger programs typically are divided into modules that several individuals or groups can work on simultaneously.

Programming Environments

Each IT department has its own programming environment and standards. Visual Basic, Python, Ruby, and SQL are examples of commonly used programming languages, and many commercial packages use a proprietary set of commands. As the trend toward Internet-based applications continues, HTML/XML, Java, and other Web-centric languages will be used extensively.

To simplify the integration of system components and reduce code development time, many programmers use an **integrated development environment** (**IDE**). IDEs can make it easier to program interactive software products by providing built-in tools and advanced features, such as real-time error detection, syntax hints, highlighted code, class browsers, and version control. As you learned in Chapter 7, IBM WebSphere and Microsoft .NET are popular IDEs. In addition to these commercial packages, programmers can use open-source IDEs such as Java-based NetBeans IDE and Eclipse. You can learn more about IDEs in Part B of the Systems Analyst's Toolkit.

TOOLKIT TIME

The CASE tools in Part B of the Systems Analyst's Toolkit can help you understand IDEs. To learn more about these tools, turn to Part B of the four-part Toolkit that follows Chapter 12.

Generating Code

You learned in earlier chapters that systems analysts use application generators, report writers, screen generators, fourth-generation languages, and other CASE tools that produce code directly from program design specifications. Some commercial applications can generate editable program code directly from macros, keystrokes, or mouse actions. Figure 11-22 on the next page shows a very simple example of a Visual Basic code module in Microsoft Access that opens a customer order form and produces a beep sound.

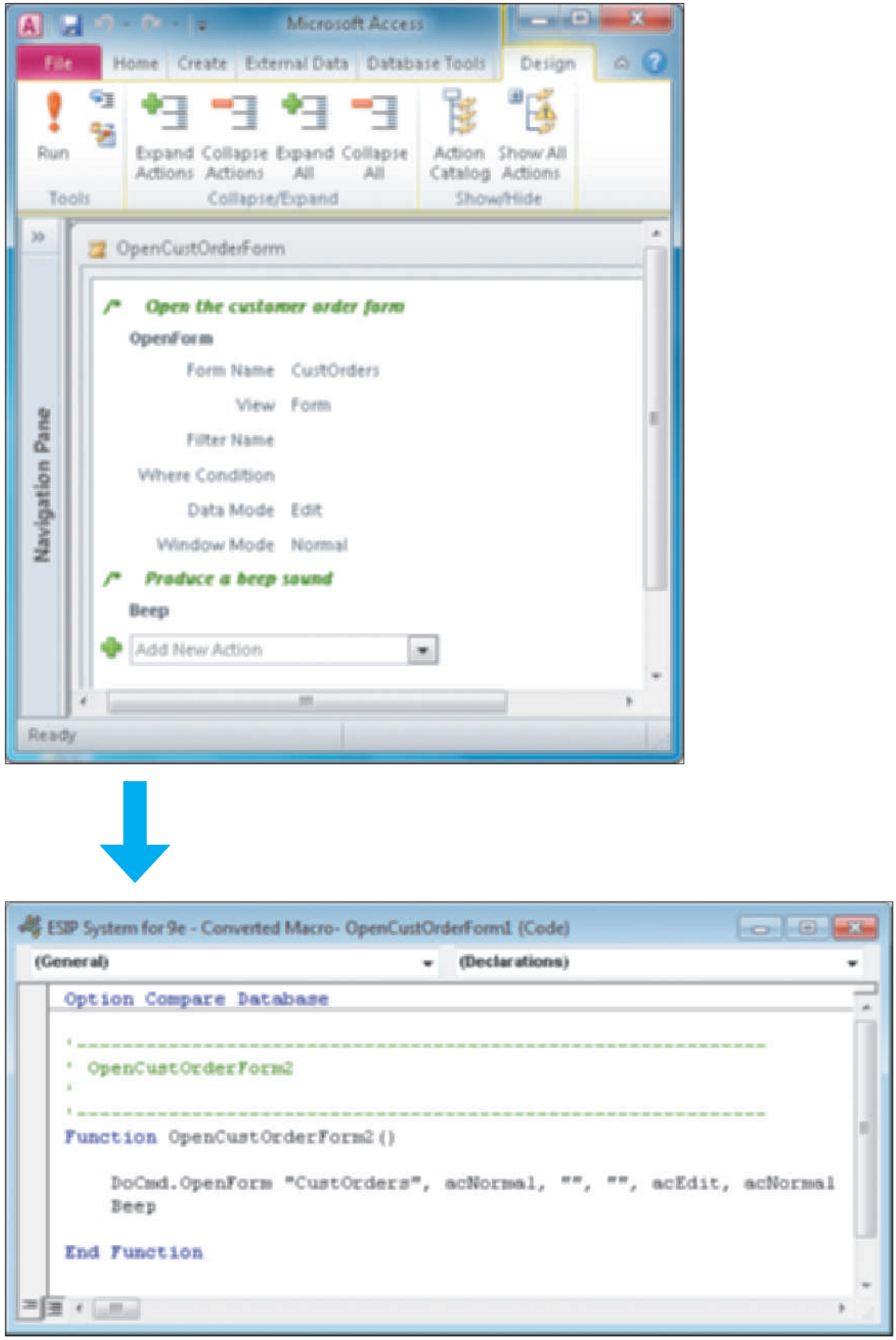


FIGURE 11-22 The simple Microsoft Access macro in the upper screen was created using keystrokes and mouse clicks. The macro's editable code is shown in the lower screen.

Note that a macro automatically generated the code, and the macro itself was created by a series of keystrokes and mouse actions. The code module shown in Figure 11-22 includes program commands, comments, and error-handling procedures.

Testing the System

After coding, a programmer must test each program to make sure it functions correctly. Later, programs are tested in groups, and finally the development team must test the entire system. The first step is to compile the program using a CASE tool or a language compiler. This process detects **syntax errors**, which are language grammar errors. The programmer corrects the errors until the program executes properly.

Next, the programmer desk checks the program. **Desk checking** is the process of reviewing the program code to spot **logic errors**, which produce incorrect results. This process can be performed by the person who wrote the program or by other programmers. Many organizations require a more formal type of desk checking called a **structured walkthrough**, or **code review**.

Typically, a group of three to five IT staff members participate in code review. The group usually consists of project team members and might include other programmers and analysts who did not work on the project. The objective is to have a peer group identify errors, apply quality standards, and verify that the program meets the requirements of the system design specification. Errors found during a structured walkthrough are easier to fix while coding is still in the developmental stages.

In addition to analyzing logic and program code, the project team usually holds a session with users called a **design walkthrough**, to review the interface with a cross-section of people who will work with the new system and ensure that all necessary features have been included. This is a continuation of the modeling and prototyping effort that began early in the systems development process.

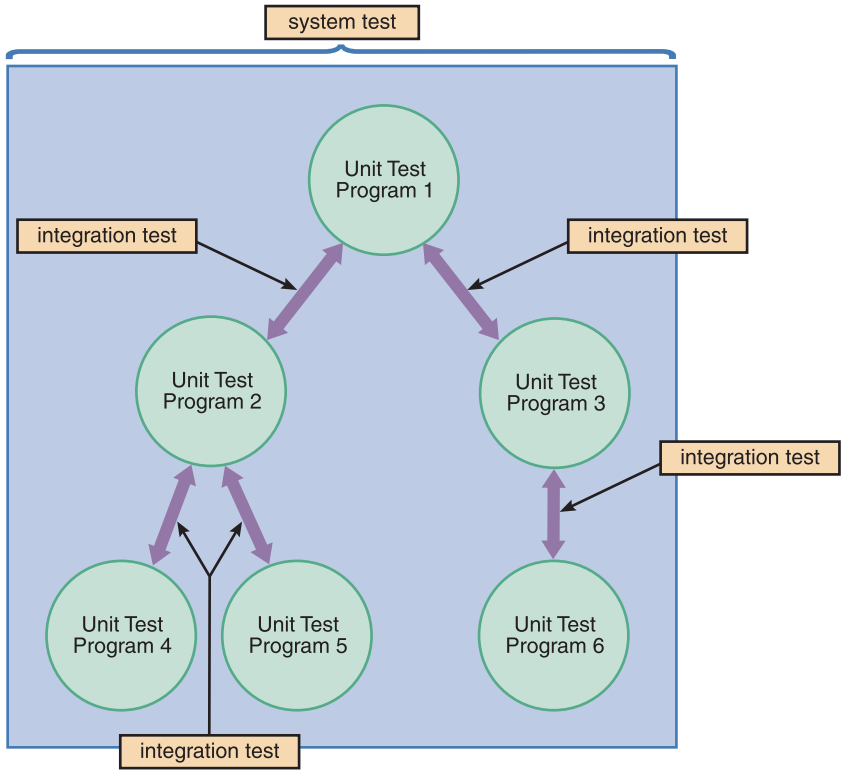


FIGURE 11-23 The first step in testing is unit testing, followed by integration testing, and then system testing.

The next step in application development is to initiate a sequence of unit testing, integration testing, and system testing, as shown in Figure 11-23.

Unit Testing

The testing of an individual program or module is called **unit testing**. The objective is to identify and eliminate execution errors that could cause the program to terminate abnormally, and logic errors that could have been missed during desk checking.

CASE IN POINT 11.1: YOUR MOVE, INC.

You work for April Olivia, the IT manager at Your Move, Inc., a large retailer specializing in games of all kinds. The company is in the final stages of developing a new inventory management system, and April wants you to handle the testing.

"Be sure you put lots of errors into the test data," she said. "Users are bound to make mistakes, and we need to design built-in safeguards that will catch the mistakes, and either fix them automatically, or alert the user to the problem."

Of course, April's comment makes a lot of sense, but you've never done this before and you wonder how to proceed. Should you try to invent every possible data error? How will you know that you've thought of every situation that could occur? Consider the problem, develop an approach, and write up your plan in a brief memo.

Test data should contain both correct data and erroneous data and should test all possible situations that could occur. For example, for a field that allows a range of numeric values, the test data should contain minimum values, maximum values, values outside the acceptable range, and alphanumeric characters. During testing, programmers can use software tools to determine the location and potential causes of program errors.

During unit testing, programmers must test programs that interact with other programs and files individually, before they are integrated into the system. This requires a technique called stub testing. In **stub testing**, the programmer simulates each program outcome or result and displays a message to indicate whether or not the program executed successfully. Each stub represents an entry or exit point that will be linked later to another program or data file.

To obtain an independent analysis, someone other than the programmer who wrote the program usually creates the test data and reviews the results. Systems analysts frequently create test data during the systems design phase as part of an overall test plan. A **test plan** consists of detailed procedures that specify how and when the testing will be performed, who will participate, and what test data will be used. A comprehensive test plan should include scenarios for every possible situation the program could encounter.

Regardless of who creates the test plan, the project manager or a designated analyst also reviews the final test results. Some organizations also require users to approve final unit test results.

Integration Testing

ON THE WEB

To learn more about system testing, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the System Testing link.

Testing two or more programs that depend on each other is called **integration testing**, or **link testing**. For example, consider an information system with a program that checks and validates customer credit status, and a separate program that updates data in the customer master file. The output from the validation program becomes input to the master file update program. Testing the programs independently does not guarantee that the data passed between them is correct. Only by performing integration testing for this pair of programs can you make sure that the programs work together properly. Figure 11-23 on the previous page shows integration testing for several groups of programs. Notice that a program can have membership in two or more groups.

Systems analysts usually develop the data they use in integration testing. As is the case with all forms of testing, integration test data must consider both normal and unusual situations. For example, integration testing might include passing typical records between two programs, followed by blank records, to simulate an unusual event or an operational problem. You should use test data that simulates actual conditions because you are testing the interface that links the programs. A testing sequence should not move to the integration test stage unless it has performed properly in all unit tests.

System Testing

After completing integration testing, you must perform **system testing**, which involves the entire information system, as shown in Figure 11-23. A system test includes all typical processing situations and is intended to assure users, developers, and managers that the program meets all specifications and that all necessary features have been included.

During a system test, users enter data, including samples of actual, or live, data, perform queries, and produce reports to simulate actual operating conditions. All processing options and outputs are verified by users and the IT project development team to ensure that the system functions correctly. Commercial software packages must undergo system testing similar to that of in-house developed systems, although unit and integration testing usually are not performed. Regardless of how the system was developed, system testing has the following major objectives:

- Perform a final test of all programs
- Verify that the system will handle all input data properly, both valid and invalid
- Ensure that the IT staff has the documentation and instructions needed to operate the system properly and that backup and restart capabilities of the system are adequate (the details of creating this sort of documentation are discussed later in this chapter)
- Demonstrate that users can interact with the system successfully
- Verify that all system components are integrated properly and that actual processing situations will be handled correctly
- Confirm that the information system can handle predicted volumes of data in a timely and efficient manner

Successful completion of system testing is the key to user and management approval, which is why system tests sometimes are called **acceptance tests**. Final acceptance tests, however, are performed during systems installation and evaluation, which is described later in this chapter.

How much testing is necessary? The answer depends on the situation and requires good judgment and input from other IT staff members, users, and management, as shown in Figure 11-24. Unfortunately, IT project managers often are pressured to finish testing quickly and hand the system over to users. Common reasons for premature or rushed testing are demands from users, tight systems development budgets, and demands from top management to finish projects early. Those pressures hinder the testing process and often have detrimental effects on the final product.

You should regard thorough testing as a cost-effective means of providing a quality product. Every error caught during testing eliminates potential expenses and operational problems. No system, however, is 100% error-free. Often, errors go undetected until the system becomes operational. Errors that affect the integrity or accuracy of data must be corrected immediately. Minor errors, such as typographical errors in screen titles, can be corrected later.

Some users want a system that is a completely finished product, while others realize that minor changes can be treated as maintenance items after the system is operational. In the final analysis, you must decide whether or not to postpone system installation if problems are discovered. If conflicting views exist, management will decide whether or not to install the system after a full discussion of the options.

FIGURE 11-24 System testing requires good judgment and input from other IT staff members, users, and IT management.

CASE IN POINT 11.2: WEBTEST, INC.

As a new systems analyst, you suspect that testing Web-based systems probably involves a different set of tools and techniques, compared to testing traditional LAN-based systems. Because you've always wanted to run your own IT company, you have decided to launch a start-up firm called WebTest, Inc., that would offer consulting services specifically aimed at testing the performance, integrity, efficiency, and security of Internet-based systems.

Your idea is to identify and purchase various Web site testing tools that currently are available, then use these tools as a Web site testing consultant. No one in your area offers this type of consulting service, so you have high hopes.

Now, you need to perform Internet research to learn more about Web testing software that is available. Review the Internet Resources Tools section, which is Part D of the Systems Analyst's Toolkit that follows Chapter 12, and use a search engine to develop a list of at least four products that you might want to use. For each product, write up a brief description of its overall purpose, its features and benefits, its cost, and how it would fit into your business game plan.

DOCUMENTATION

Documentation describes an information system and helps the users, managers, and IT staff who must interact with it. Accurate documentation can reduce system downtime, cut costs, and speed up maintenance tasks. Figure 11-25 shows an example of software that can automate the documentation process and help software developers generate accurate, comprehensive reference material.

Documentation is essential for successful system operation and maintenance. In addition to supporting a system's users, accurate documentation is essential for IT staff members who must modify the system, add a new feature, or perform maintenance. Documentation includes program documentation, system documentation, operations documentation, and user documentation.

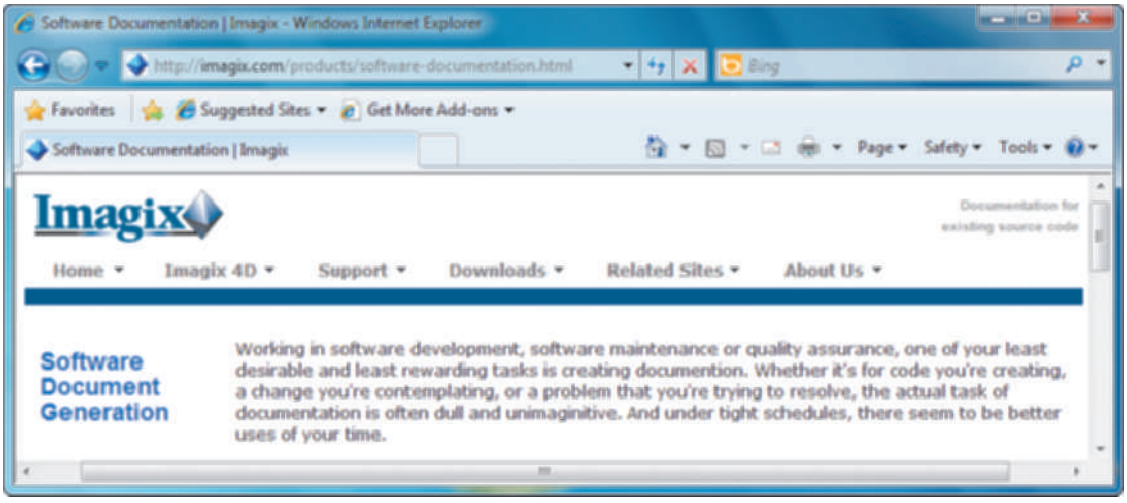


FIGURE 11-25 In addition to CASE tools, software such as Imagix can automate the task of software documentation.

Program Documentation

Program documentation describes the inputs, outputs, and processing logic for all program modules. The program documentation process starts in the systems analysis phase and continues during systems implementation. Systems analysts prepare overall documentation, such as process descriptions and report layouts, early in the SDLC. This documentation guides programmers, who construct modules that are well supported by internal and external comments and descriptions that can be understood and maintained easily. A systems analyst usually verifies that program documentation is complete and accurate.

FIGURE 11-26 Bugzilla is an example of a defect tracking program that can track bugs and manage software quality assurance.

System developers also use **defect tracking software**, sometimes called **bug tracking software**, to document and track program defects, code changes, and replacement code, called **patches**. One popular example is Bugzilla, shown in Figure 11-26. According to its Web site, Bugzilla is a free, open-source program that can track bugs and manage software quality assurance.

System Documentation

System documentation describes the system's functions and how they are implemented. System documentation includes data dictionary entries, data flow diagrams, object models, screen layouts, source documents, and the systems request that initiated the project. System documentation is necessary reference material for the programmers and analysts who must support and maintain the system.

Most of the system documentation is prepared during the systems analysis and systems design phases. During the systems implementation phase, an analyst must review prior documentation to verify that it is complete, accurate, and up to date, including any changes made during the implementation process. For example, if a screen or report has been modified, the analyst must update the documentation. Updates to the system documentation should be made in a timely manner to prevent oversights.

ON THE WEB

To learn more about documentation, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Documentation link.

Operations Documentation

If the information system environment involves a minicomputer, a mainframe, or centralized servers, the analyst must prepare documentation for the IT group that supports centralized operations. A mainframe installation might require the scheduling of batch jobs and the distribution of printed reports. In this type of environment, the IT operations staff serves as the first point of contact when users experience problems with the system.

Operations documentation contains all the information needed for processing and distributing online and printed output. Typical operations documentation includes the following information:

- Program, systems analyst, programmer, and system identification
- Scheduling information for printed output, such as report run frequency and deadlines

- Input files and where they originate; and output files and destinations
- E-mail and report distribution lists
- Special forms required, including online forms
- Error and informational messages to operators and restart procedures
- Special instructions, such as security requirements

Operations documentation should be clear, concise, and available online if possible. If the IT department has an operations group, you should review the documentation with them, early and often, to identify any problems. If you keep the operations group informed at every phase of the SDLC, you can develop operations documentation as you go along.

User Documentation

User documentation consists of instructions and information to users who will interact with the system and includes user manuals, Help screens, and tutorials. Programmers or systems analysts usually create program documentation and system documentation. To produce effective and clear user documentation — and hence have a successful project — you need someone with expert skills in this area doing the development, just as you need someone with expert skills developing the software. The skill set required to develop documentation usually is not the same as that to develop a system. This is particularly true as you move into the world of online documentation, which needs to coordinate with print documentation and intranet and Internet information. Technical writing requires specialized skills, and competent technical writers are valuable members of the IT team.

Just as you cannot throw a system together in several days, you cannot add documentation at the end. That is a common misconception and often proves fatal to a project. While that has always been true of software user documentation, this is an even more critical issue now that online Help and context-sensitive Help so often are needed. Context-sensitive Help is part of the program. You must put coded callouts in the text that link to the correct page of information in the documentation. To try to go back and add this after the fact would take a great deal of time; depending on the project size, it could take months! Additionally, it could introduce other coding errors — and it all has to be tested as well.

Systems analysts usually are responsible for preparing documentation to help users learn the system. In larger companies, a technical support team that includes technical writers might assist in the preparation of user documentation and training materials. Regardless of the delivery method, user documentation must be clear, understandable, and readily accessible to users at all levels.

User documentation includes the following:

- A system overview that clearly describes all major system features, capabilities, and limitations
- Description of source document content, preparation, processing, and samples
- Overview of menu and data entry screen options, contents, and processing instructions
- Examples of reports that are produced regularly or available at the user's request, including samples
- Security and audit trail information
- Explanation of responsibility for specific input, output, or processing requirements
- Procedures for requesting changes and reporting problems

- Examples of exceptions and error situations
- Frequently asked questions (FAQs)
- Explanation of how to get help and procedures for updating the user manual

Most users prefer **online documentation**, which provides immediate Help when they have questions or encounter problems. Many users are accustomed to context-sensitive help screens, hints and tips, hypertext, on-screen demos, and other user-friendly features commonly found in popular software packages; they expect the same kind of support for in-house developed software.

If the system will include online documentation, that fact needs to be identified as one of the system requirements. If the documentation will be created by someone other than the analysts who are developing the system, that person or group needs to be involved as early as possible to become familiar with the software and begin developing the required documentation and support material. In addition, system developers must determine whether the documentation will be available from within the program, or as a separate entity in the form of a tutorial, slide presentation, reference manual, or Web site. If necessary, links should be created within the program that will take the user to the appropriate documentation.

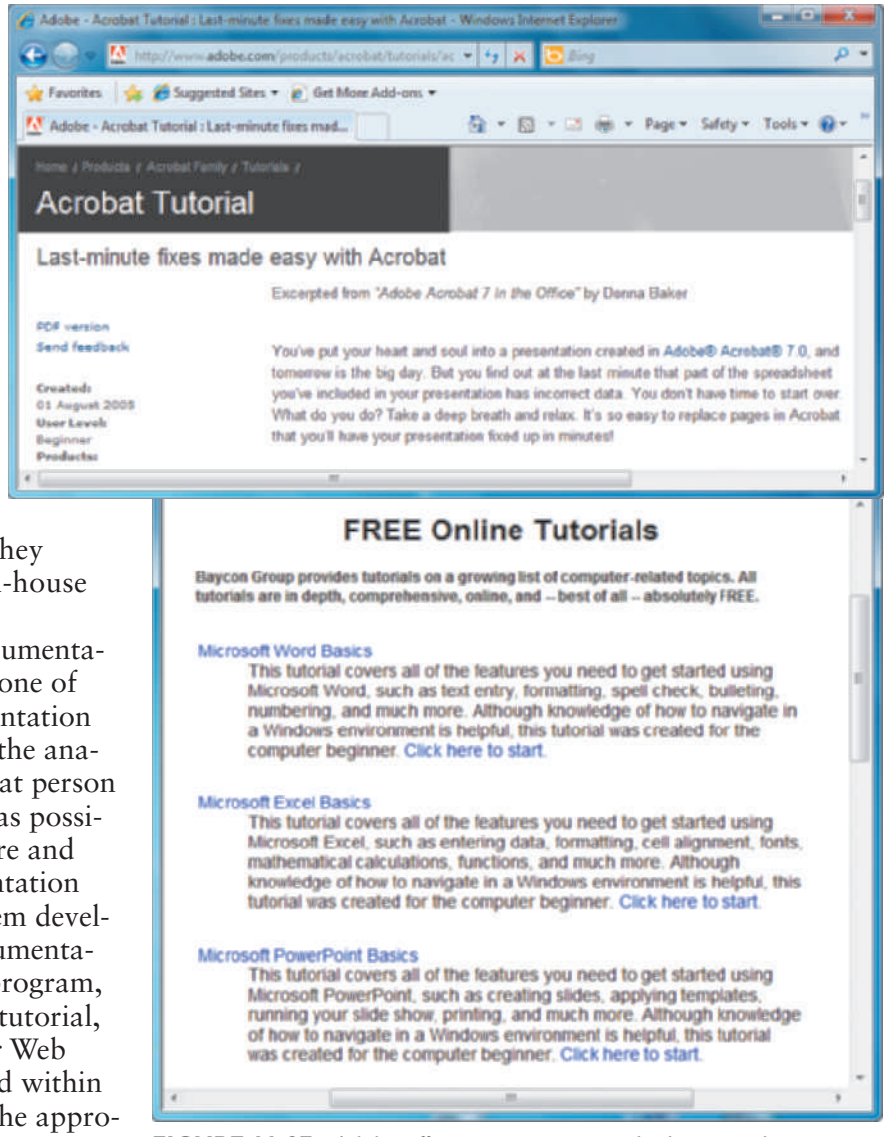


FIGURE 11-27 Adobe offers interactive tutorials, demos, and seminars for its products, while Baycon Group provides a variety of free tutorials.

Effective online documentation is an important productivity tool because it empowers users and reduces the time that IT staff members must spend in providing telephone, e-mail, or face-to-face assistance. Interactive tutorials are especially popular with users who like to learn by doing. Many software packages include tutorials, and additional tutorials are available online, as shown in Figure 11-27.

In addition to program-based assistance, the Internet offers an entirely new level of comprehensive, immediate support. Many programs include links to Web sites, intranet sites, and Internet-based technical support. For example, the Cisco Systems site shown in Figure 11-28 on the next page offers a wide range of support services, including a wiki that allows Cisco users to collaborate and share their knowledge.

Although online documentation is essential, written documentation material also is valuable, especially in training users and for reference purposes. A sample page from a user manual is shown in Figure 11-29 on page 533. Systems analysts or technical writers usually prepare the manual, but many companies invite users to review the material and participate in the development process.

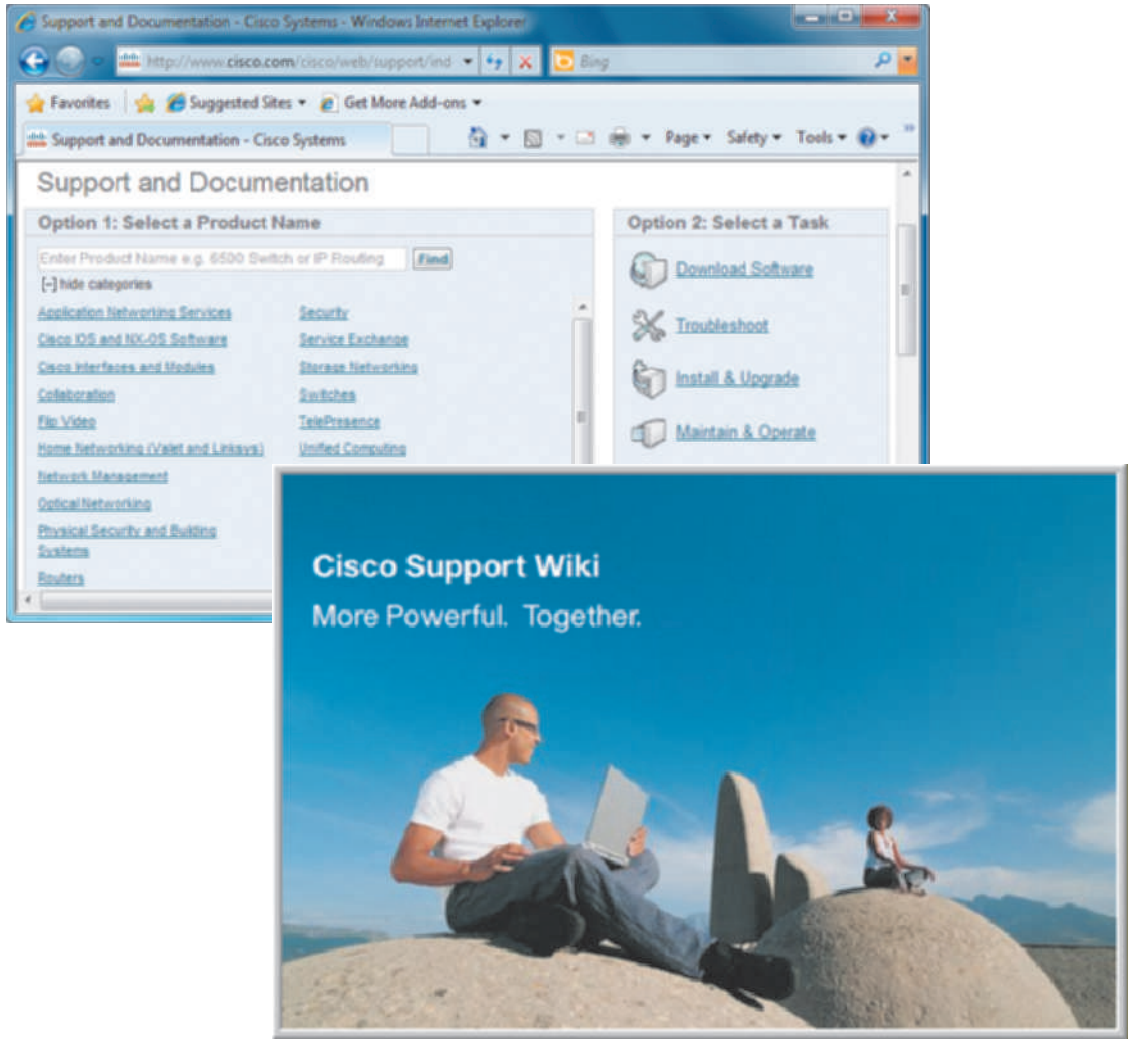


FIGURE 11-28 In addition to traditional types of technical support, the Cisco Systems Web site features a support wiki.

No matter what form of user documentation your system will require, you must keep in mind that it can take a good deal of time to develop. The time between finishing software coding and the time when a complete package — including documentation — can be released to users is entirely dependent on how well the documentation is thought out in advance. If the completion of your project includes providing user documentation, this issue needs to be addressed from the very beginning of the project. Determining what the user documentation requirements are and ascertaining who will complete the documents is critical to a timely release of the project.

Neglecting user documentation issues until after all the program is complete often leads to one of two things: (1) The documentation will be thrown together quickly just to get it out the door on time, and it more than likely will be inadequate; or (2) it will be done correctly, and the product release will be delayed considerably.

User training typically is scheduled when the system is installed; the training sessions offer an ideal opportunity to distribute the user manual and explain the procedures for updating it in the future. Training for users, managers, and IT staff is described later in this chapter.

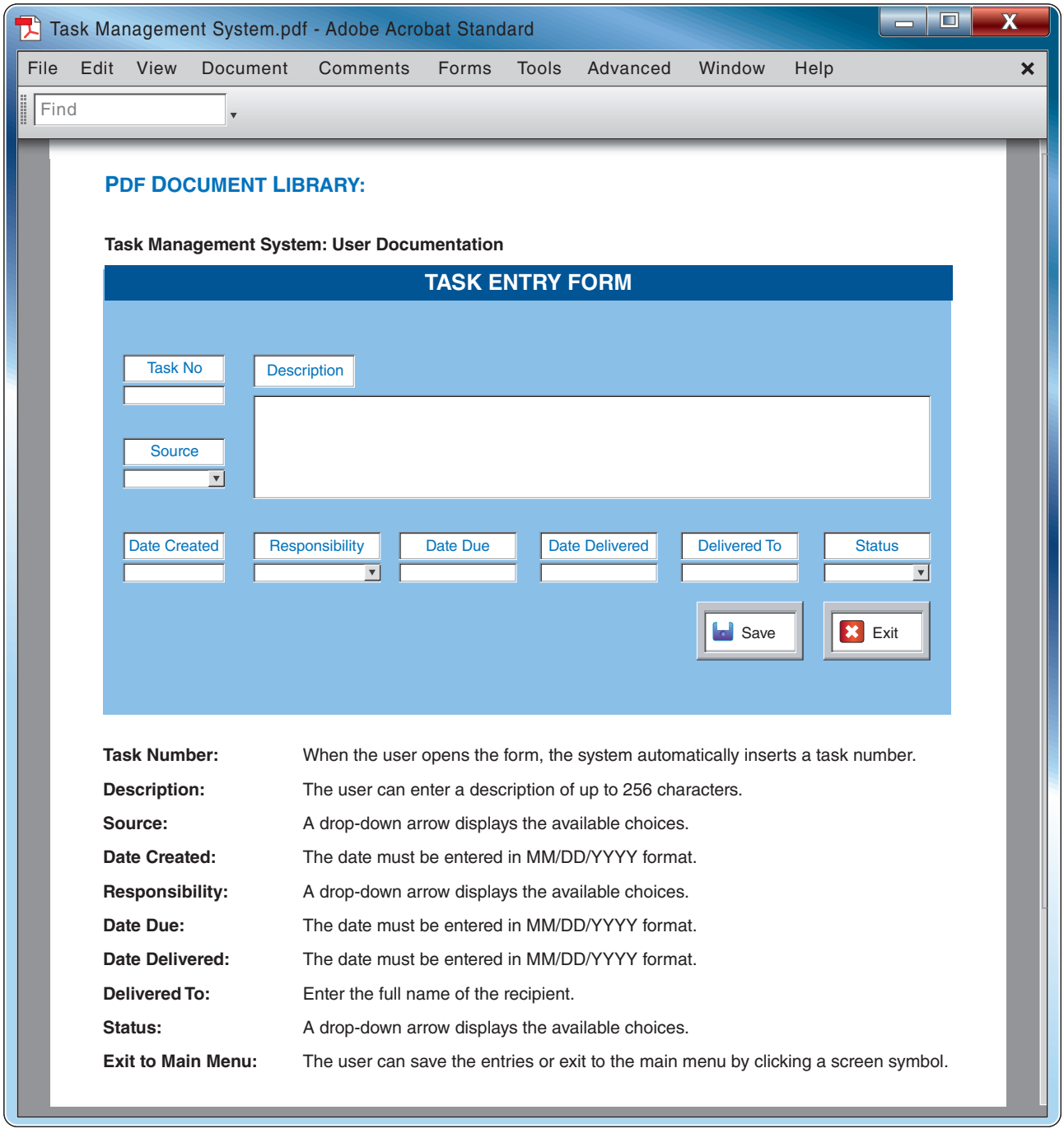


FIGURE 11-29 A sample page from a user manual. The instructions explain how to add a new task to the system.

Management Approval

After system testing is complete, you present the results to management. You should describe the test results, update the status of all required documentation, and summarize input from users who participated in system testing. You also must provide detailed time schedules, cost estimates, and staffing requirements for making the system fully operational. If system testing produced no technical, economical, or operational problems, management determines a schedule for system installation and evaluation.

System Installation and Evaluation

The following sections describe system installation and evaluation tasks that are performed for every information systems project, whether you develop the application in-house or purchase it as a commercial package.

The new system now is ready to go to work. Your earlier design activities produced the overall architecture and processing strategy, and you consulted users at every stage of development. You developed and tested programs individually, in groups, and as a complete system. You prepared the necessary documentation and checked it for accuracy, including support material for IT staff and users. Now, you will carry out the remaining steps in systems implementation:

- Prepare a separate operational and test environment
- Provide training for users, managers, and IT staff
- Perform data conversion and system changeover
- Carry out a post-implementation evaluation of the system
- Present a final report to management

Operational and Test Environments

You learned earlier that an environment, or platform, is a specific combination of hardware and software. The environment for the actual system operation is called the **operational environment** or **production environment**. The environment that analysts and programmers use to develop and maintain programs is called the **test environment**. A separate test environment is necessary to maintain system security and integrity and protect the operational environment. Typically, the test environment resides on a limited-access workstation or server located in the IT department.

Access to the operational environment is limited to users and must strictly be controlled. Systems analysts and programmers should not have access to the operational environment except to correct a system problem or to make authorized modifications or enhancements. Otherwise, IT department members have no reason to access the day-to-day operational system.

The test environment for an information system contains copies of all programs, procedures, and test data files. Before making any changes to an operational system, you must verify them in the test environment and obtain user approval. Figure 11-30 shows the differences between test environments and operational environments.

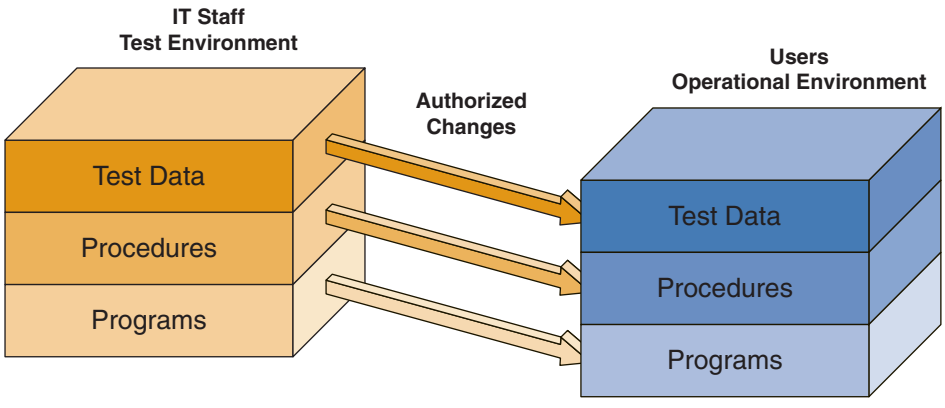


FIGURE 11-30 The test environment versus the operational environment. Notice that access to the test environment is limited to IT staff, while the operational environment is restricted to users.

An effective testing process is essential, whether you are examining an information system or a batch of computer chips. Every experienced systems analyst can tell you a story about an apparently innocent program change that was introduced without being tested properly. In those stories, the innocent change invariably ends up causing some unexpected and unwanted changes to the program. After any modification, you should repeat the same acceptance tests you ran

when the system was developed. By restricting access to the operational area and performing all tests in a separate environment, you can protect the system and avoid problems that could damage data or interrupt operations.

The operational environment includes hardware and software configurations and settings, system utilities, telecommunications resources, and any other components that might affect system performance. Because network capability is critically important in a client/server environment, you must verify connectivity, specifications, and performance before installing any applications. You should check all communications features in the test environment carefully, and then check them again after loading the applications into the operational environment. Your documentation should identify all network specifications and settings, including technical and operational requirements for communications hardware and software. If you have to build or upgrade network resources to support the new system, you must test the platform rigorously before system installation begins.

FIGURE 11-31 In any situation, training must fit the needs of users and help them carry out their job functions.

Training

No system can be successful without proper training, whether it involves software, hardware, or manufacturing, as shown in Figure 11-31. A successful information system requires training for users, managers, and IT staff members. The entire systems development effort can depend on whether or not people understand the system and know how to use it effectively.

Training Plan

You should start to consider a **training plan** early in the systems development process. As you create documentation, you should think about how to use the material in future training sessions. When you implement the system, it is essential to provide the right training for the right people at the right time. The first step is to identify who should receive training and what training is needed. You must look carefully at the organization, how the system will support business operations, and who will be involved or affected. Figure 11-32 on the next page shows specific training topics for users, managers, and IT staff. Notice that each group needs a mix of general background and detailed information to understand and use the system.

As shown in Figure 11-32, the three main groups for training are users, managers, and IT staff. A manager does not need to understand every submenu or feature, but he or she does need a system overview to ensure that users are being trained properly and are using the system correctly. Similarly, users need to know how to perform their day-to-day job functions, but do not need to know how the company allocates system operational charges among user departments. IT staff people probably need the most information. To support the new system, they must have a clear understanding of how the system functions, how it supports business requirements, and the skills that users need to operate the system and perform their tasks.

After you identify the objectives, you must determine how the company will provide training. The main choices are to obtain training from vendors, outside training firms, or use IT staff and other in-house resources.

ON THE WEB

To learn more about training, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Training link.

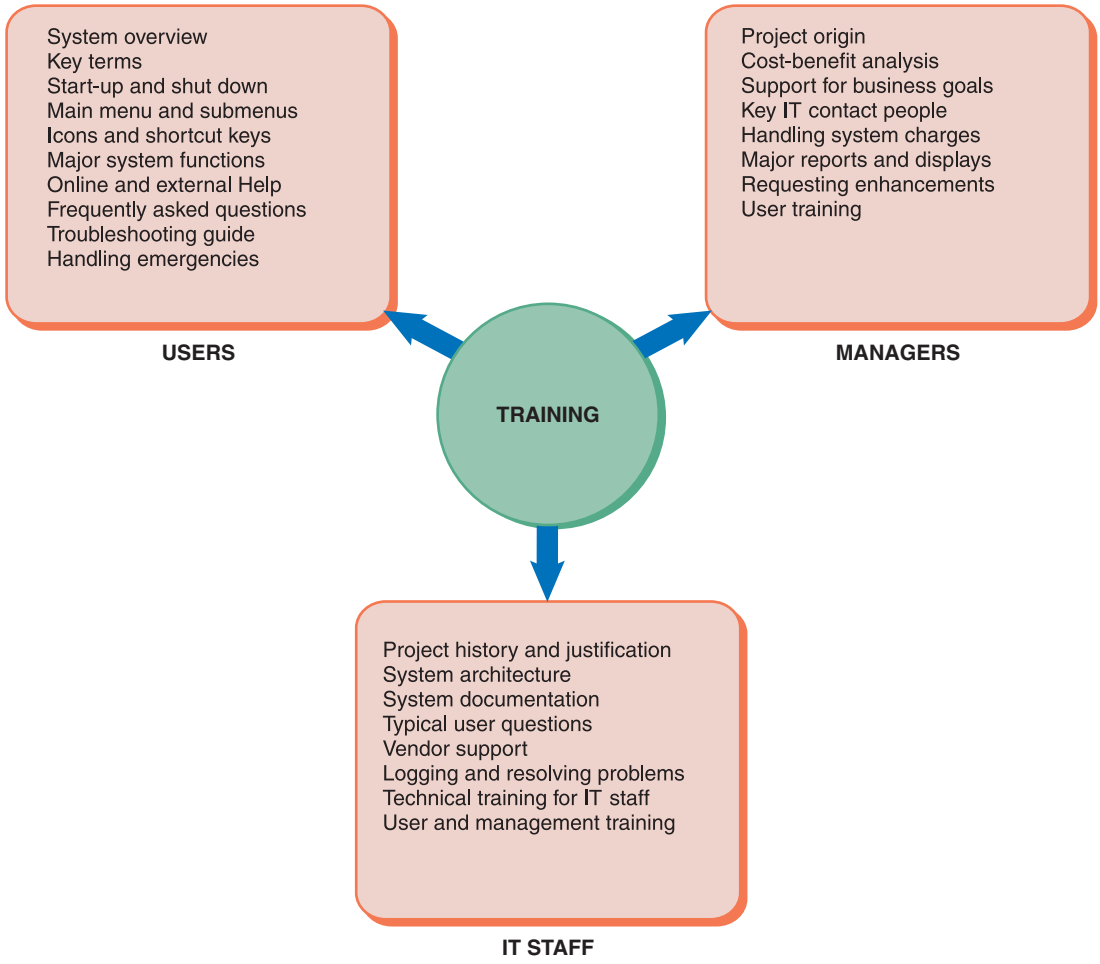


FIGURE 11-32 Examples of training topics for three different groups. Users, managers, and IT staff members have different training needs.

Vendor Training

If the system includes the purchase of software or hardware, then vendor-supplied training is one of the features you should include in the RFPs (requests for proposal) and RFQs (requests for quotation) that you send to potential vendors.

Many hardware and software vendors offer training programs free or at a nominal cost for the products they sell. In other cases, the company might negotiate the price for training, depending on their relationship with the vendor and the prospect of future purchases. The training usually is conducted at the vendor's site by experienced trainers who provide valuable hands-on experience. If a large number of people need training, you might be able to arrange classes at your location.

Vendor training often gives the best return on your training dollars because it is focused on products that the vendor developed. The scope of vendor training, however, usually is limited to a standard version of the vendor's software or hardware. You might have to supplement the training in-house, especially if your IT staff customized the package.

Webinars, Podcasts, and Tutorials

Many vendors offer Web-based training options, including Webinars, podcasts, and tutorials. Figure 11-33 shows a Webinar and a podcast. A **Webinar**, which combines the words Web and seminar, is an Internet-based training session that provides an interactive experience. Most Webinars are scheduled events with a group of preregistered users and an online presenter or instructor. A prerecorded Webinar session also can be delivered as a **Webcast**, which is a one-way transmission, whenever a user wants or needs training support.

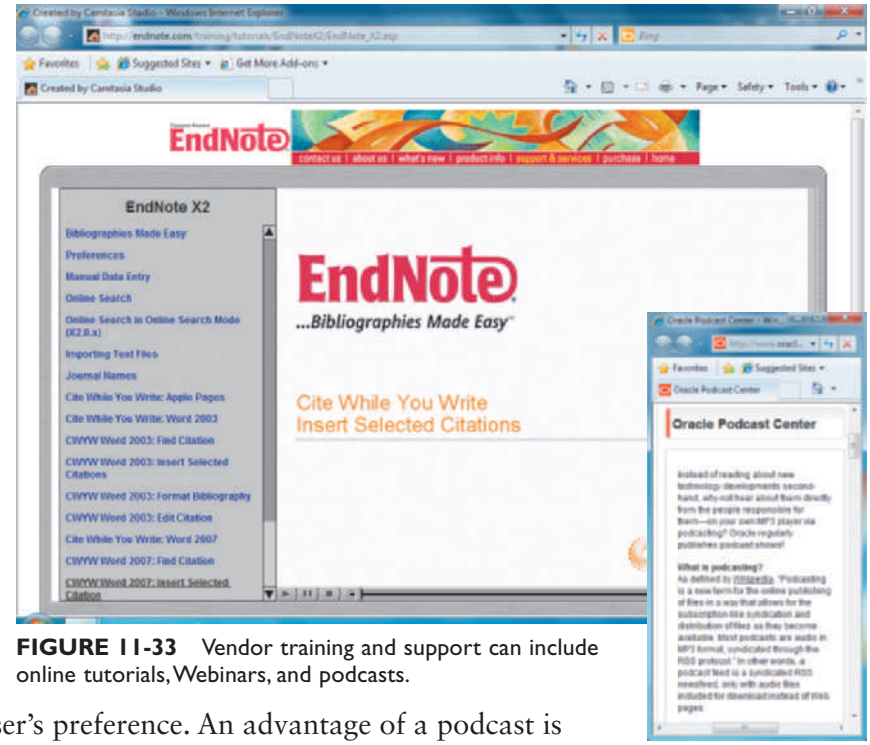


FIGURE 11-33 Vendor training and support can include online tutorials, Webinars, and podcasts.

A **podcast** refers to a Web-based broadcast that allows a user to download multimedia files to a PC or portable device. Podcasts can be prescheduled, made available on demand, or delivered as automatic updates, depending on a user's preference. An advantage of a podcast is that **subscribers** can access the recorded material anywhere, anytime.

A **tutorial** is a series of online interactive lessons that present material and provide a dialog with users. Tutorials can be developed by software vendors, or by a company's IT team. A tutorial example is included in the in-house training section.

Outside Training Resources

You also can look into an independent training firm to provide in-house hardware or software training. If vendor training is not practical and your organization does not have the internal resources to perform the training, you might find that outside training consultants are a desirable alternative.

The rapid expansion of information technology has produced tremendous growth in the computer-training field. Many training consultants, institutes, and firms are available that provide either standardized or customized training packages. IT industry leaders, such as Hewlett-Packard and IBM, offer a wide variety of training solutions, as shown in Figure 11-34 on the next page.

If you decide to investigate outside training resources, you can contact a training provider and obtain references from clients. You also can seek assistance from nonprofit sources with an interest in training, including universities, industry associations, and information management organizations. For example, Figure 11-35 on the next page shows the Web site for Western Illinois University's Center for the Application of Information Technologies (CAIT), which describes a variety of IT education and training resources.

Training Tips

The IT staff and user departments often share responsibility for developing and conducting training programs for internally developed software. If your organization has a help desk, the staff might be able to handle user training.

Multimedia is an effective training method. Presentation software, such as Microsoft PowerPoint, OpenOffice Impress, or Corel Presentations, allows you to design training sessions that combine slides, animation, and sound. You also can use programs that capture actual keystrokes and mouse actions, and then replay the screens as a demonstration for

FIGURE 11-34 IT industry leaders, such as Hewlett-Packard and IBM, offer a wide variety of training solutions.

users. If your firm has a media or graphic arts group, they can help you prepare training aids such as videotapes, charts, and other instructional materials. When developing a training program, you should keep the following guidelines in mind.

- *Train people in groups, with separate training programs for distinct groups.* Group training, as shown in Figure 11-36, makes the most efficient use of time and training facilities. In addition, if the group is small, trainees can learn from the questions and problems of others. A training program must address the job interests and skills of a wide range of participants. For example, IT staff personnel and users require very different information. Problems often arise when some participants have technical backgrounds and others do not. A single program will not meet everyone's needs.

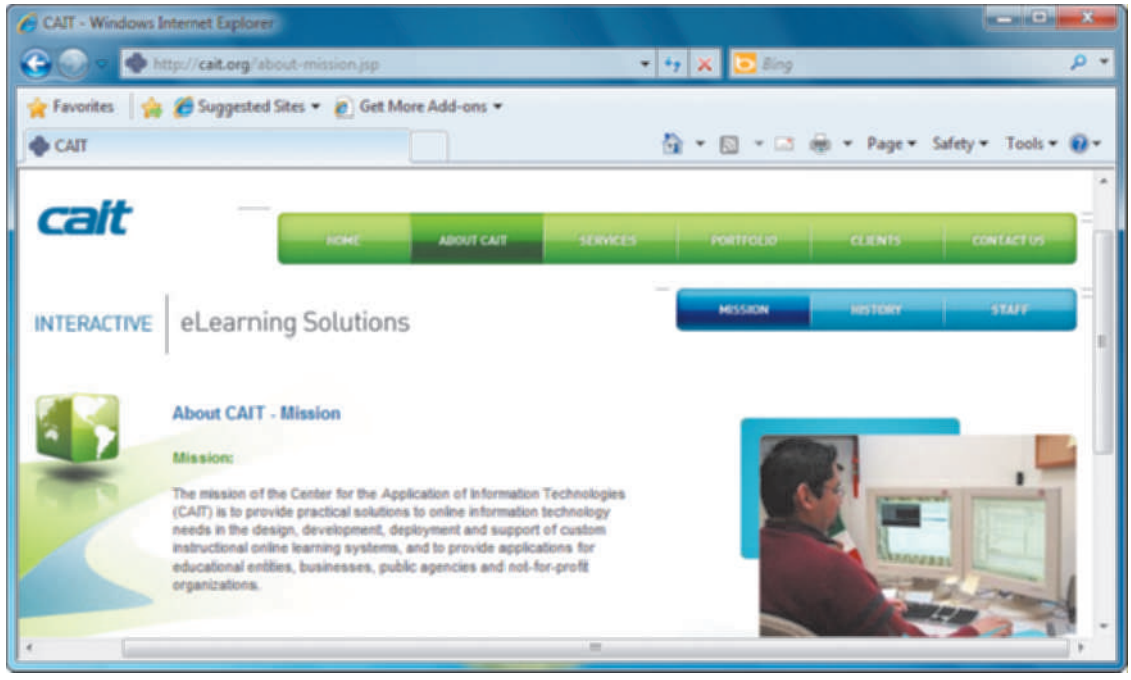


FIGURE 11-35 Western Illinois University's Center for the Application of Information Technologies (CAIT) helps organizations design, develop, and deploy information technology education and training.

- *Select the most effective place to conduct the training.* Training employees at your company's location offers several advantages. Employees incur no travel expense, they can respond to local emergencies that require immediate attention, and training can take place in the actual environment where the system will operate. You can encounter some disadvantages, however. Employees who are distracted by telephone calls and other duties will not get the full benefit of the training. In addition, using the organization's computer facilities for training can disrupt normal operations and limit the amount of actual hands-on training.
- *Provide for learning by hearing, seeing, and doing.* Some people learn best from lectures, discussions, and question-and-answer sessions. Others learn best from viewing demonstrations or from reading documentation and other material. Most people learn best from hands-on experience. You should provide training that supports each type of learning.
- *Rely on previous trainees.* After one group of users has been trained, they can assist others. Users often learn more quickly from coworkers who share common experience and job responsibilities. Using a **train-the-trainer** strategy, you can select knowledgeable users who then conduct sessions for others. When utilizing train-the-trainer techniques, the initial training must include not only the use of the application or system, but some instruction on how to present the materials effectively.

FIGURE 11-36 Users must be trained on the new system. Training sessions might be one-on-one or group situations such as the one shown here. Many vendors provide product training as part of an overall service to customers.

Interactive Training

Usually, a relationship exists between training methods and costs. Training an airline pilot in a state-of-the-art simulator is quite different from helping corporate users learn a new inventory system. Obviously, training budgets are business decisions, and IT staff sometimes has to work with the resources that are available, rather than the resources they wish they had. Most people prefer hands-on training. However, other less-expensive methods can be used, including training manuals, printed handouts, and online materials. Even the user documentation like that shown in Figure 11-29 on page 533 can be valuable, if users know how to find and use it.

If you are launching a new system and you lack the resources to develop formal training materials, you can design a series of dialog boxes that respond with Help information and suggestions whenever users select various menu topics. A good user interface also includes helpful error messages and hints, as discussed in Chapter 8. However, the most effective training is interactive, self-paced, and multimedia-based. Online training and video tutorials are discussed in the following sections.

ONLINE TRAINING Regardless of the instructional method, training lessons should include step-by-step instructions for using the features of the information system. Training materials should resemble actual screens, and tasks should be typical of a user's daily work — the more realistic, the better. For example, Figure 11-37 on the next page shows a sample tutorial lesson for a sales prospect management system. In Lesson 1, the user learned how to enter and exit the system. In Lesson 2, the user learns how to add a sales prospect and return to the main menu.

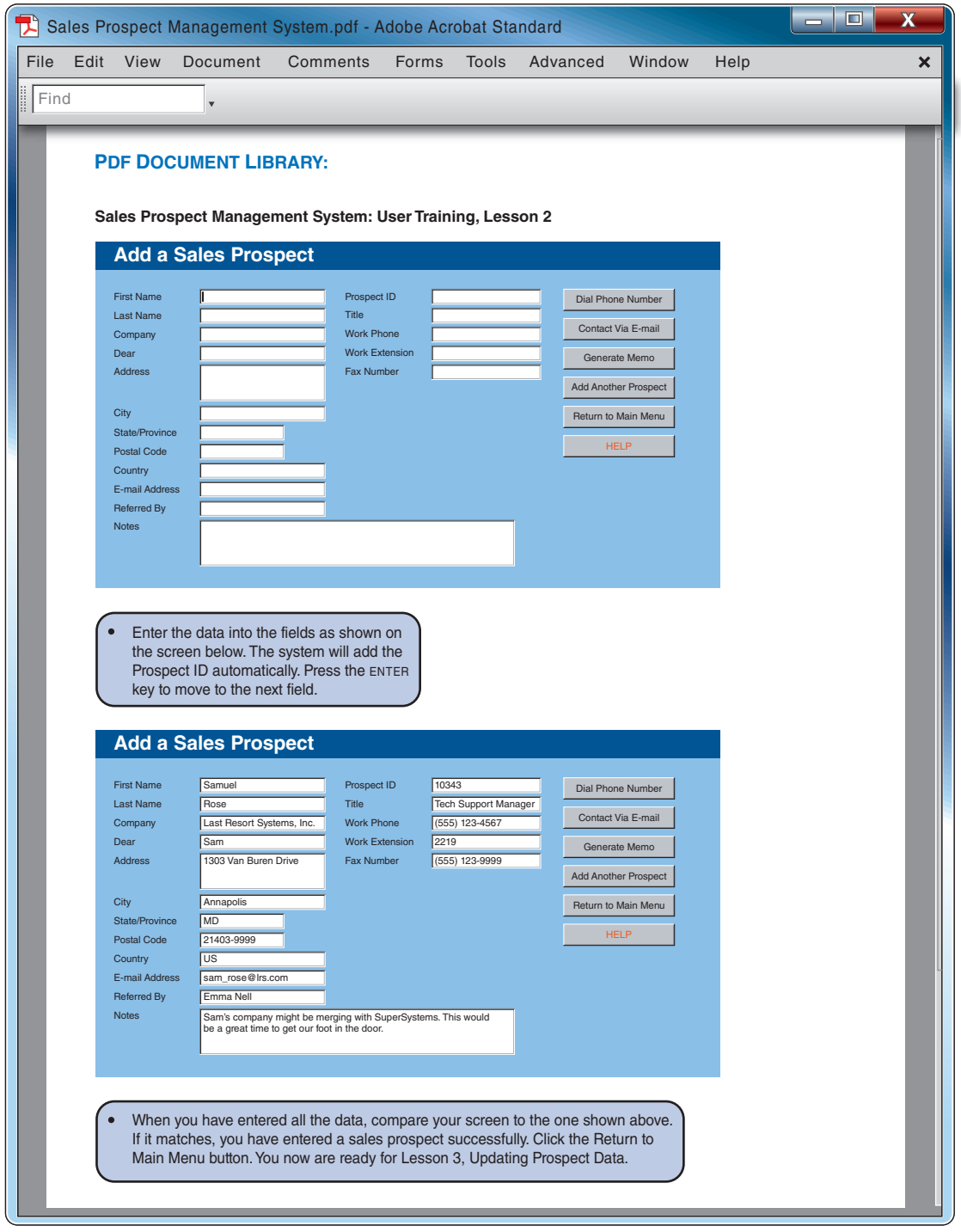


FIGURE 11-37 A sample lesson in an online tutorial.

More sophisticated tutorials might offer interactive sessions where users can perform practice tasks and view feedback. Online training materials also should include a reference section that summarizes all options and commands, lists all possible error messages, and what actions the user should take when a problem arises.

VIDEO TUTORIALS You don't have to be a professional video developer to create effective training tutorials. For example, the Video Learning Sessions for this textbook were initially created as classroom teaching tools. Later, they were polished, edited, and transformed into streaming videos.

Suppose you want to develop a tutorial that shows how to create a structure chart, but you have no budget for special multimedia software. You do have Windows 7 and Office 2010, so you could start by creating a set of slides with screen text and graphic images. Then, you could add a live-motion screen capture and an audio narration. Finally, you could import the media into Windows Live Moviemaker where you could edit, save, and publish your training movie. Figure 11-38 shows a step-by-step plan for creating a video tutorial.

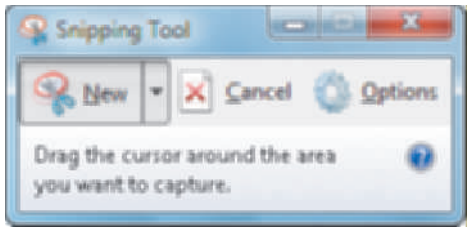
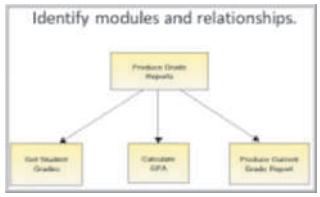
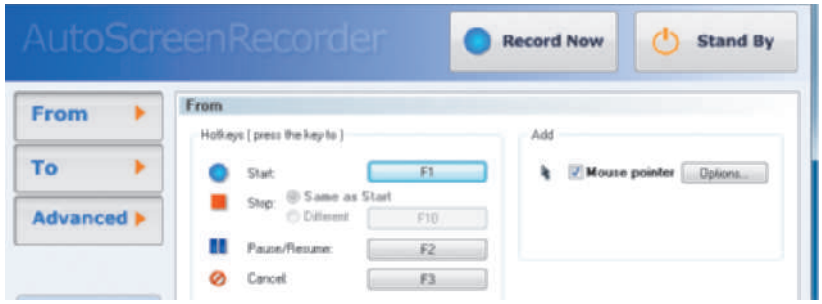
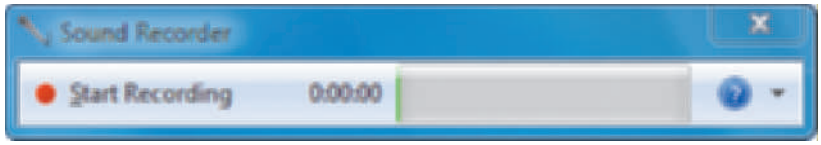
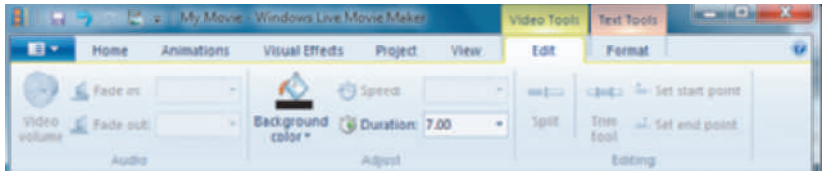
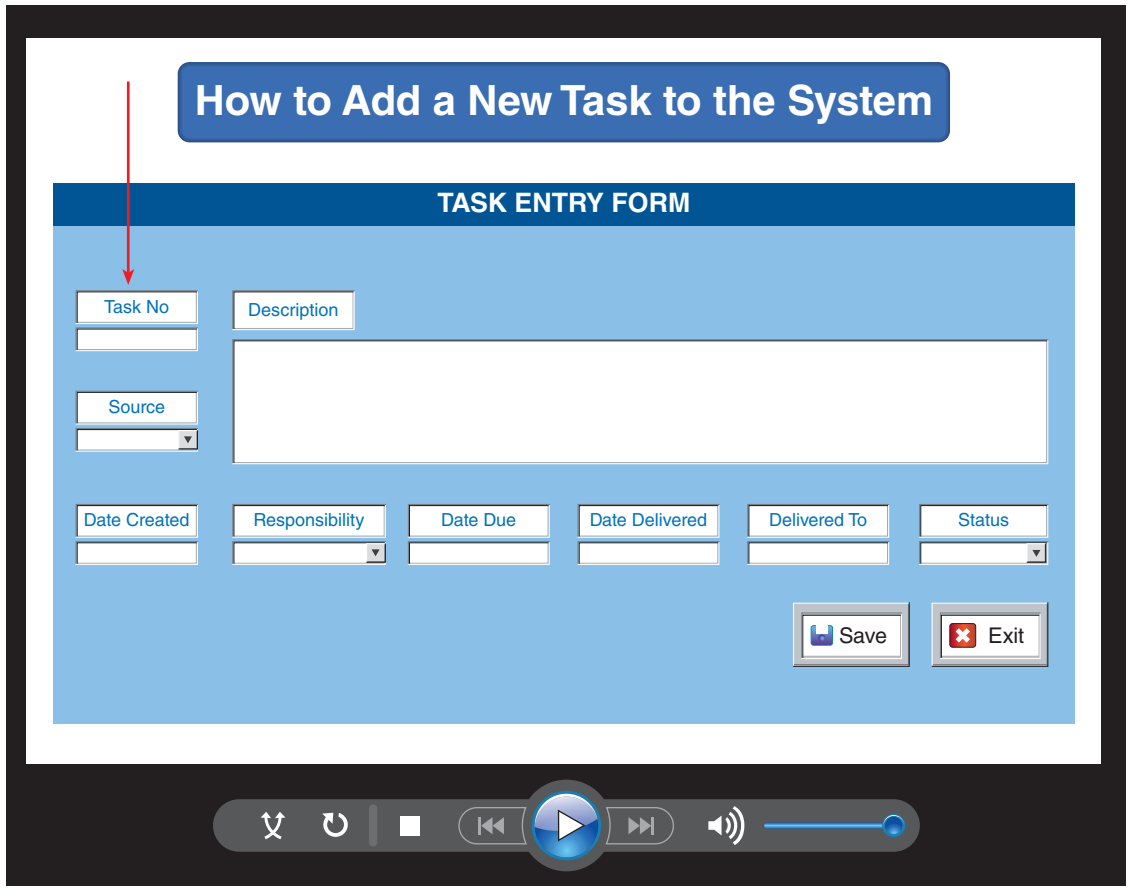
1. Capture individual screens	Use Windows Snipping Tool to capture full or partial screen images	
2. Integrate screen images and text	Paste the images into Microsoft PowerPoint slides, then add suitable narration text in the Notes section	
3. Capture live-motion video	Use freeware such as Wisdom-Soft Auto Screen Recorder	
4. Record audio narration	Use Windows Sound Recorder to read the narration text	
5. Produce a training video	Use Windows Moviemaker to import media, edit, and produce your training video	

FIGURE 11-38 You can use free software such as Windows Snipping Tool for image capture, Wisdom-Soft Auto Screen Recorder for live-motion video, and Windows Sound Recorder for audio narration. After you create the media, you can import the material into Windows Live Movie Maker.

Figure 11-39 shows a video training session design. The topic is the same Task Management System shown in Figure 11-29 on page 533. There, the focus was on effective documentation. Here, the goal is self-paced, interactive training for users. Figure 11-39 shows a screen design with instructions to the video developer, and a script with instructions to the narrator. At the end of the production process, these materials will be transformed into an integrated multimedia presentation.



Note to video developer:	*Add a red arrow to highlight each bullet of the narration. The first arrow is shown as an example.*
Note to narrator:	*Pause briefly after each bulleted section and speak slowly enough for viewers to follow the steps.*
Narration text:	*Welcome to this video training session. In the session, you will learn how to add a new task to the system. As a reminder, you can pause the session at any time, or go back to review an earlier section. Now let's get started.* • *When you open the Task Entry Form, the system will insert a task number for you.* • *Next, click the Description section and enter a description of up to 256 characters.* • *Now click the drop-down arrow in the Source section and select one of the choices. Do the same in the Responsibility section.* • *In the Date Created section, enter the date in em-em slash dee-dee slash why-why-why-why format. Do the same in the other three date fields.* • *Finally, click the drop-down arrow in the Status section and select—one of the choices.* • *You can save your entries, or exit to the main menu at anytime. Just click the SAVE or EXIT symbols.*

FIGURE 11-39 A sample video tutorial might include images, narration text, and notes to the video developer and narrator.

In addition to shareware and built-in software, you can use a video editing application such as Camtasia, which offers a powerful, user-friendly interface. Figure 11-40 shows a Camtasia screen, where the user has imported streaming video clip, several images, and a recorded narration. Now the user can crop, split, and extend the video and audio tracks, and can add various special effects.

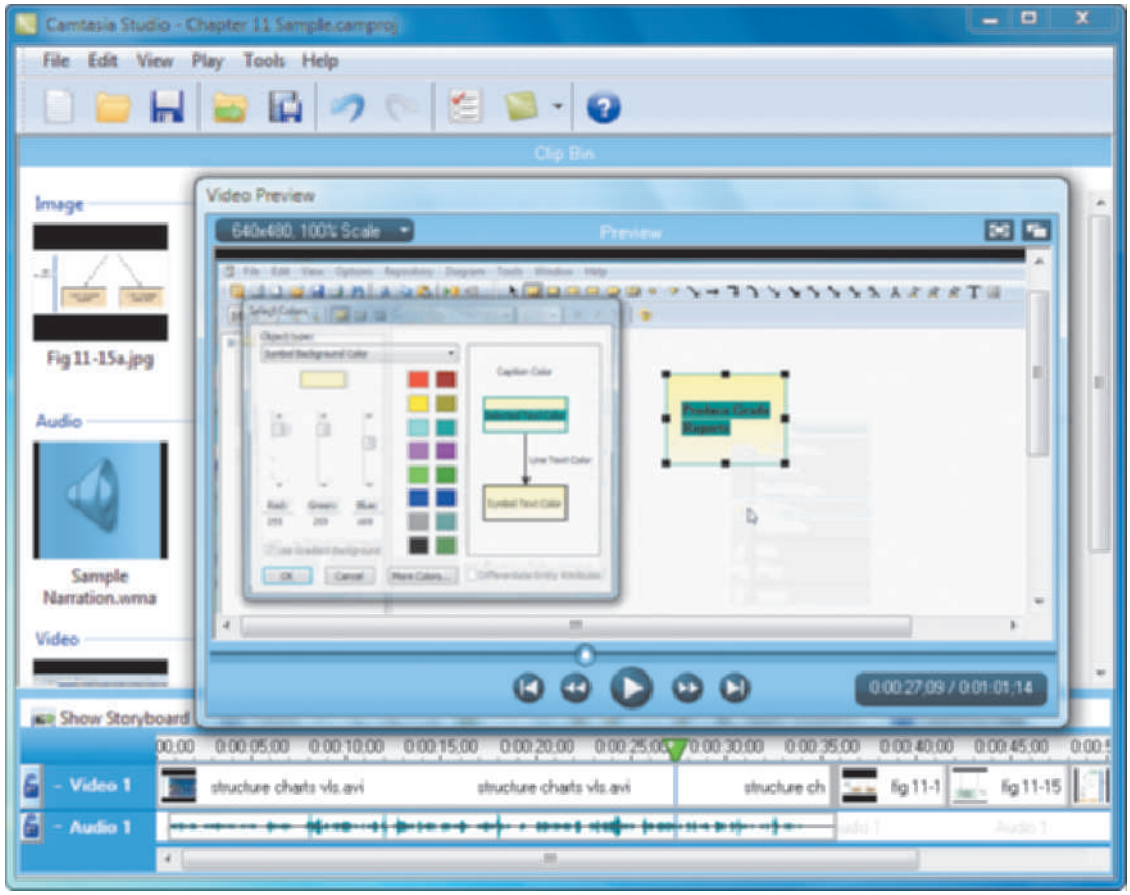


FIGURE 11-40 Camtasia is a moderately-priced video editing tool that can produce professional-quality training videos.

When training is complete, many organizations conduct a full-scale test, or **simulation**, which is a dress rehearsal for users and IT support staff. Organizations include all procedures, such as those that they execute only at the end of a month, quarter, or year, in the simulation. As questions or problems arise, the participants consult the system documentation, help screens, or each other to determine appropriate answers or actions. This full-scale test provides valuable experience and builds confidence for everyone involved with the new system.

Data Conversion

Data conversion is an important part of the system installation process. During **data conversion**, existing data is loaded into the new system. Depending on the system, data conversion can be done before, during, or after the operational environment is complete. You should develop a data conversion plan as early as possible, and the conversion process should be tested when the test environment is developed.

ON THE WEB

To learn more about data conversion, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to **On the Web Links** for this chapter, and locate the Data Conversion link.

Data Conversion Strategies

When a new system replaces an existing system, you should automate the data conversion process, if possible. The old system might be capable of **exporting** data in an acceptable format for the new system or in a standard format, such as ASCII or ODBC. **ODBC** (**Open Database Connectivity**) is an industry-standard protocol that allows DBMSs from various vendors to interact and exchange data. Most database vendors provide ODBC drivers, which are a form of middleware. As you learned in Chapter 10, middleware connects dissimilar applications and enables them to communicate.

If a standard format is not available, you must develop a program to extract the data and convert it to an acceptable format. Data conversion is more difficult when the new system replaces a manual system, because all data must be entered manually unless it can be scanned. Even when you can automate data conversion, a new system often requires additional data items, which might require manual entry.

Data Conversion Security and Controls

You should maintain strict input controls during the conversion process, when data is extremely vulnerable. You must ensure that all system control measures are in place and operational to protect data from unauthorized access and to help prevent erroneous input.

Even with careful data conversion and input controls, some errors will occur. For example, duplicate customer records or inconsistent part numbers might have been tolerated by the old system, but will cause the new system to crash. Most organizations require that users verify all data, correct all errors, and supply every missing data item during conversion. Although the process can be time-consuming and expensive, it is essential that the new system be loaded with accurate, error-free data.

System Changeover

System changeover is the process of putting the new information system online and retiring the old system. Changeover can be rapid or slow, depending on the method. The four changeover methods are direct cutover, parallel operation, pilot operation, and phased operation. Direct cutover is similar to throwing a switch that instantly changes over from the old system to the new. Parallel operation requires that both systems run simultaneously for a specified period, which is the slowest method. The other methods, pilot and phased operation, fall somewhere between direct cutover and parallel operation. Figure 11-41 illustrates the four system changeover methods.

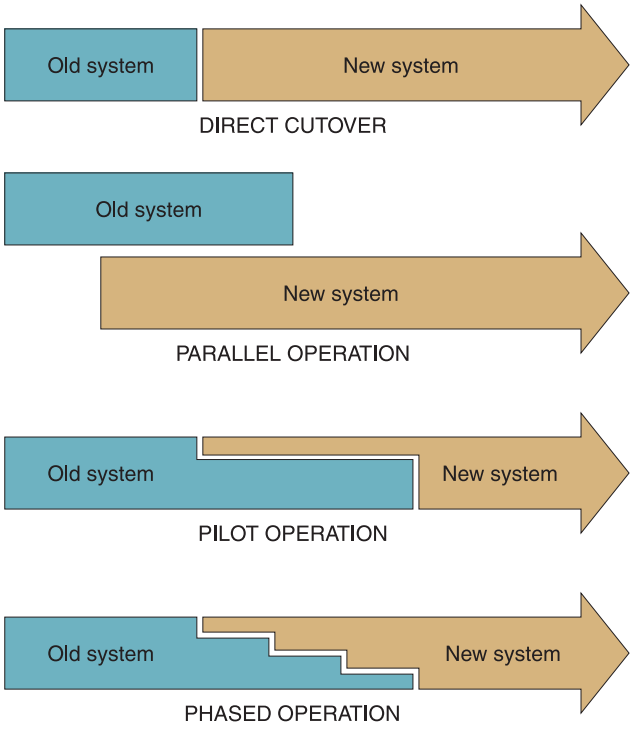


FIGURE 11-41 The four system changeover methods.

Direct Cutover

The **direct cutover** approach causes the changeover from the old system to the new system to occur immediately when the new system becomes operational. Direct cutover usually is the least expensive changeover method because the IT group has to operate and maintain only one system at a time.

Direct cutover, however, involves more risk than other changeover methods. Regardless of how thoroughly and carefully you conduct testing and training, some difficulties can arise when the system goes into operation. Problems can result from data situations that

were not tested or anticipated or from errors caused by users or operators. A system also can encounter difficulties because live data typically occurs in much larger volumes than test data.

Although initial implementation problems are a concern with all four changeover methods, they are most significant when the direct cutover approach is used. Detecting minor errors also is more difficult with direct cutover because users cannot verify current output by comparing it to output from the old system. Major errors can cause a system process to terminate abnormally, and with the direct cutover method, you cannot revert to the old system as a backup option.

Companies often choose the direct cutover method for implementing commercial software packages because they feel that commercial packages involve less risk of total system failure. Commercial software is certainly not risk-free, but the software vendor usually maintains an extensive knowledge base and can supply reliable, prompt fixes for most problems.

For systems developed in-house, most organizations use direct cutover only for noncritical situations. Direct cutover might be the only choice, however, if the operating environment cannot support both the old and new systems or if the old and new systems are incompatible.

Timing is very important when using a direct cutover strategy. Most systems operate on weekly, monthly, quarterly, and yearly cycles. For example, consider a payroll system that produces output on a weekly basis. Some employees are paid twice a month, however, so the system also operates semimonthly. Monthly, quarterly, and annual reports also require the system to produce output at the end of every month, quarter, and year. When a cyclical information system is implemented in the middle of any cycle, complete processing for the full cycle requires information from both the old and the new systems. To minimize the need to require information from two different systems, cyclical information systems usually are converted using the direct cutover method at the beginning of a quarter, calendar year, or fiscal year.

Parallel Operation

The **parallel operation** changeover method requires that both the old and the new information systems operate fully for a specified period. Data is input into both systems, and output generated by the new system is compared with the equivalent output from the old system. When users, management, and the IT group are satisfied that the new system operates correctly, the old system is terminated.

The most obvious advantage of parallel operation is lower risk. If the new system does not work correctly, the company can use the old system as a backup until appropriate changes are made. It is much easier to verify that the new system is working properly under parallel operation than under direct cutover, because the output from both systems is compared and verified during parallel operation.

Parallel operation, however, does have some disadvantages. First, it is the most costly changeover method. Because both the old and the new systems are in full operation, the company pays for both systems during the parallel period. Users must work in both systems and the company might need temporary employees to handle the extra workload. In addition, running both systems might place a burden on the operating environment and cause processing delays.

Parallel operation is not practical if the old and new systems are incompatible technically, or if the operating environment cannot support both systems. Parallel operation also is inappropriate when the two systems perform different functions or if the new system involves a new method of business operations. For example, until a company installs data scanners in a factory, it is impractical to launch a new production tracking system that requires such technology.

Pilot Operation

The **pilot operation** changeover method involves implementing the complete new system at a selected location of the company. A new sales reporting system, for instance, might be implemented in only one branch office, or a new payroll system might be installed in only one department. In these examples, the group that uses the new system first is called the **pilot site**. During pilot operation, the old system continues to operate for the entire organization, including the pilot site. After the system proves successful at the pilot site, it is implemented in the rest of the organization, usually using the direct cutover method. Therefore, pilot operation is a combination of parallel operation and direct cutover methods.

Restricting the implementation to a pilot site reduces the risk of system failure, compared with a direct cutover method. Operating both systems for only the pilot site is less expensive than a parallel operation for the entire company. In addition, if you later use a parallel approach to complete the implementation, the changeover period can be much shorter if the system proves successful at the pilot site.

Phased Operation

The **phased operation** changeover method allows you to implement the new system in stages, or modules. For example, instead of implementing a new manufacturing system all at once, you first might install the materials management subsystem, then the production control subsystem, then the job cost subsystem, and so on. You can implement each subsystem by using any of the other three changeover methods.

Analysts sometimes confuse phased and pilot operation methods. Both methods combine direct cutover and parallel operation to reduce risks and costs. With phased operation, however, you give a part of the system to all users, while pilot operation provides the entire system, but to only some users.

One advantage of a phased approach is that the risk of errors or failures is limited to the implemented module only. For instance, if a new production control subsystem fails to operate properly, that failure might not affect the new purchasing subsystem or the existing shop floor control subsystem.

Phased operation is less expensive than full parallel operation because you have to work with only one part of the system at a time. A phased approach is not possible, however, if the system cannot be separated easily into logical modules or segments. In addition, if the system involves a large number of separate phases, phased operation can cost more than a pilot approach.

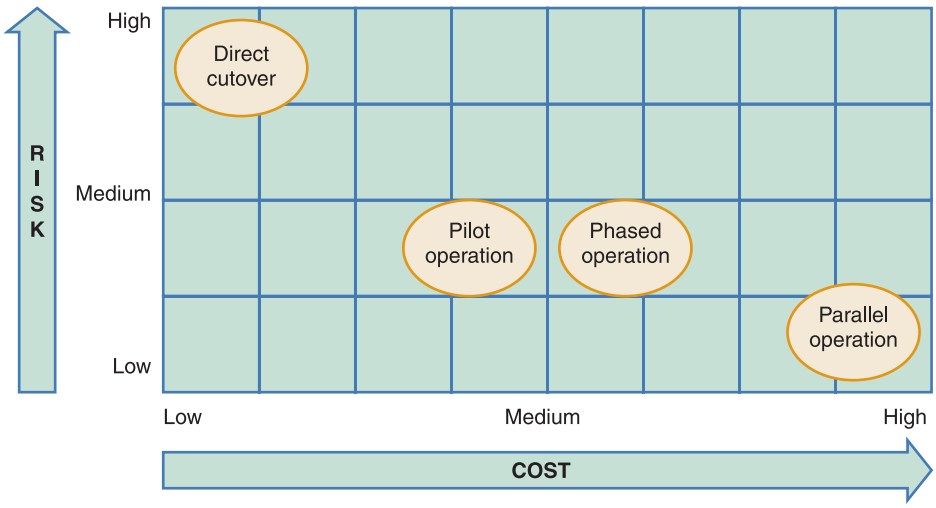


FIGURE 11-42 Relative risk and cost characteristics of the four changeover methods.

Figure 11-42 shows that each changeover method has risk and cost factors. As a systems analyst, you must weigh the advantages and disadvantages of each method and recommend the best choice in a given situation. The final changeover decision will be based on input from the IT staff, users, and management — and the choice must reflect the nature of the business and the degree of acceptable risk.

CASE IN POINT 11.3: GLOBAL COOLING

You are a systems analyst at Global Cooling, a leading manufacturer of air conditioning units. You are leading a team that is developing a new production scheduling system. The project is now in the application development stage. Unit testing has been completed, and you are in the final stages of integration testing. Your supervisor, Ella Pham, is eager to implement the new application ahead of schedule and asked if you could trim system testing from two weeks to three days, and use a direct cutover method instead of the parallel changeover method that originally was planned. Write a brief memo expressing your views.

POST-IMPLEMENTATION TASKS

Once the new system is operational, you must perform two additional tasks: Prepare a post-implementation evaluation and deliver a final report to management.

Post-Implementation Evaluation

A **post-implementation evaluation** assesses the overall quality of the information system. The evaluation verifies that the new system meets specified requirements, complies with user objectives, and produces the anticipated benefits. In addition, by providing feedback to the development team, the evaluation also helps improve IT development practices for future projects.

A post-implementation evaluation should examine all aspects of the development effort and the end product — the developed information system. A typical evaluation includes feedback for the following areas:

- Accuracy, completeness, and timeliness of information system output
- User satisfaction
- System reliability and maintainability
- Adequacy of system controls and security measures
- Hardware efficiency and platform performance
- Effectiveness of database implementation
- Performance of the IT team
- Completeness and quality of documentation
- Quality and effectiveness of training
- Accuracy of cost-benefit estimates and development schedules

You can apply the same fact-finding techniques in a post-implementation evaluation that you used to determine the system requirements during the systems analysis phase. When evaluating a system, you should:

- Interview members of management and key users
- Observe users and computer operations personnel actually working with the new information system
- Read all documentation and training materials
- Examine all source documents, output reports, and screen displays
- Use questionnaires to gather information and opinions from a large number of users
- Analyze maintenance and help desk logs

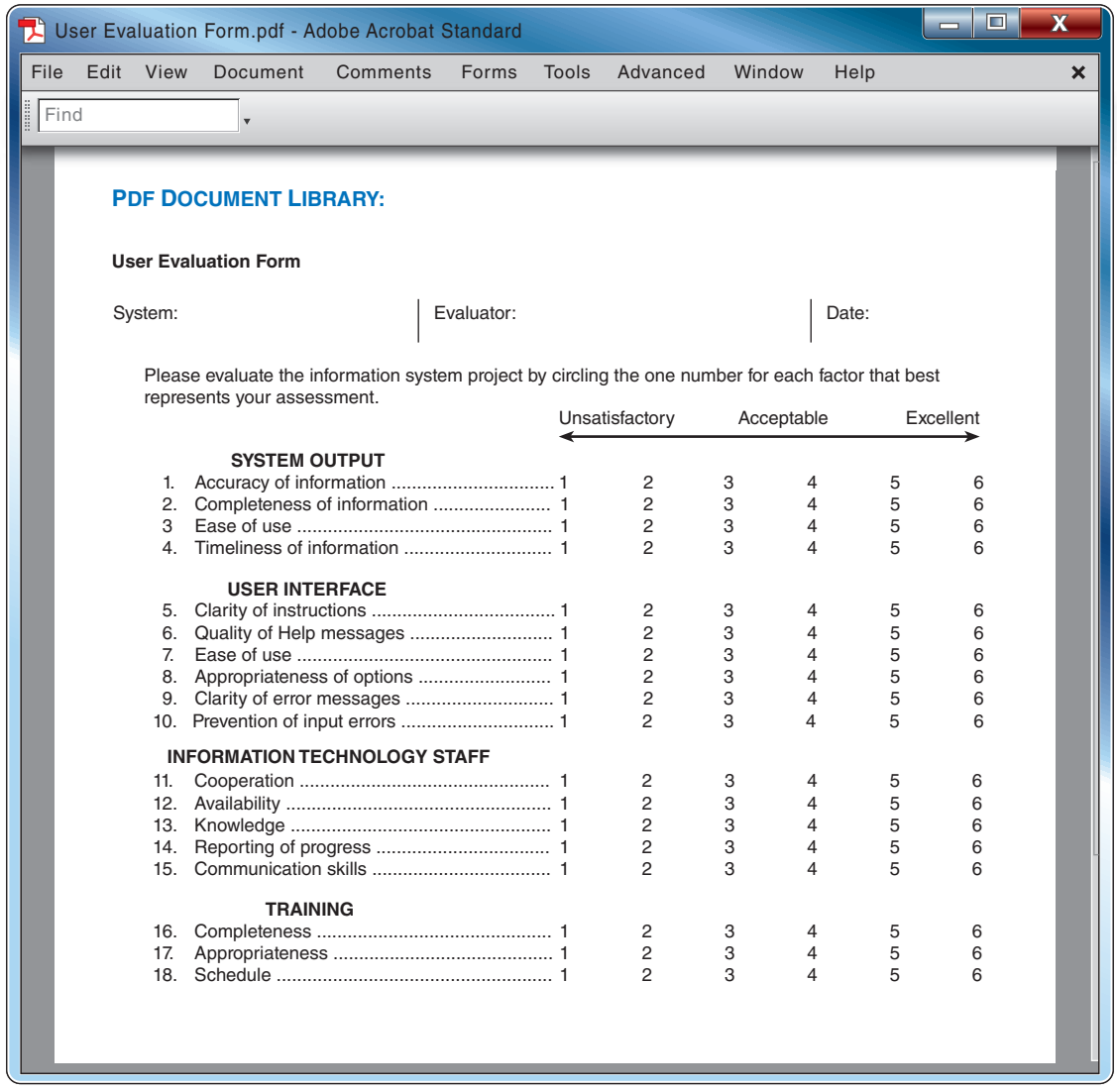


FIGURE 11-43 Sample user evaluation form. The numerical scale allows easy tabulation of results. Following this section, the form provides space for open-ended comments and suggestions.

Figure 11-43 shows the first page of a sample user evaluation form for the new information system where users evaluate 18 separate elements on a numerical scale, so the results can be tabulated easily. Following that section, the form provides space for open-ended comments and suggestions.

Whenever possible, people who were not directly involved in developing the system should conduct the post-implementation evaluation. IT staff and users usually perform the evaluation, although some firms use an internal audit group or independent auditors to ensure the accuracy and completeness of the evaluation.

When should post-implementation evaluation occur? Is it better to wait until the new system has been in operation for one month, six months, one year, or longer? Users can forget details of the developmental effort if too much time elapses before the evaluation. After several months or a year, for instance, users might not remember whether they learned a procedure through training, from user documentation, or by experimenting with the system on their own.

Users also might forget their impressions of IT team members over time. An important purpose of the post-implementation evaluation is to improve the quality of IT department functions, including interaction with users, training, and documentation. Consequently,

the evaluation team should perform the assessment while users are able to recall specific incidents, successes, and problems so they can offer suggestions for improvement. Post-implementation evaluation primarily is concerned with assessing the quality of the new system. If the team performs the evaluation too soon after implementation, users will not have enough time to learn the new system and appreciate its strengths and weaknesses. Although many IT professionals recommend conducting the evaluation after at least six months of system operation, pressure to finish the project sooner usually results in an earlier evaluation in order to allow the IT department to move on to other tasks.

Ideally, conducting a post-implementation evaluation should be standard practice for all information systems projects. Sometimes, evaluations are skipped because users are eager to work with the new system, or because IT staff members have more pressing priorities. In some organizations, management might not recognize the importance and benefits of a post-implementation evaluation. The evaluations are extremely important, however, because they enable the development team and the IT department to learn what worked and what did not work. Otherwise, developers might commit the same errors in another system.

CASE IN POINT 11.4: YORKTOWN INDUSTRIES

Cindy Winslow liked her new job as lead systems analyst at Yorktown Industries. She was pleased that her development team completed the new human resources system ahead of schedule and under budget. Cindy looked forward to receiving the post-implementation evaluation because she was confident that both the system and the development team would receive high marks from users and managers.

After the system operated for one month, Cindy received a call from her supervisor, Ted Haines. Ted told her that she would have to handle the evaluation, even though she headed the development effort. Cindy told Ted that she did not feel comfortable evaluating her own team's work. She explained that someone who was not involved in its development should do an independent evaluation. Ted responded that he had full confidence in Cindy's ability to be objective. He explained that no one else was available and he needed the evaluation quickly so he could move forward with the next stage in the corporate development plan.

Cindy was troubled about the situation and she called you, a professional acquaintance, for your advice. What would you tell her and why?

Final Report to Management

At the end of each SDLC phase, you submit a report to management, and the systems implementation phase is no exception. Your report should include the following:

- Final versions of all system documentation
- Planned modifications and enhancements to the system that have been identified
- Recap of all systems development costs and schedules
- Comparison of actual costs and schedules to the original estimates
- Post-implementation evaluation, if it has been performed

The final report to management marks the end of systems development work. In the next chapter, you will study the role of a systems analyst during systems operation, security, and support, which is the final phase of the SDLC.

TOOLKIT TIME

The Communication tools in Part A of the Systems Analyst's Toolkit can help you develop better reports and presentations. To learn more about these tools, turn to Part A of the four-part Toolkit that follows Chapter 12.

A QUESTION OF ETHICS

Your friend Jill is handling the testing for the new accounting system, and right now she is very upset about the most recent results. "It seems like every time we fix one thing, another issue pops up! After ten days of testing and adjusting, we are meeting over 90% of the goals and benchmarks. If we're looking for perfection, we'll never make the implementation deadline for the new system, and the users will be all over us. Not to mention top management's reaction to a delay. I'm sure we can resolve some of these issues after the system becomes operational."

How would you respond to Jill? Are ethical issues involved? What are your responsibilities, as an employee, as an IT professional, and as a friend?

Chapter Summary

The systems implementation phase consists of application development, testing, installation, and evaluation of the new system. During application development, analysts determine the overall design strategy and work with programmers to complete design, coding, testing, and documentation. Quality assurance is essential during the implementation phase. Many companies utilize software engineering concepts and quality standards established by the International Organization for Standardization (ISO).

Each systems development approach has its own set of tools. For example, structured development relies heavily on DFDs and structure charts. A structure chart consists of symbols that represent program modules, data couples, control couples, conditions, and loops. Object-oriented methods use a variety of diagrams, including use case, class, sequence, and transition state diagrams. Agile methods tend to use a spiral or other iterative model.

System developers also can use more generic tools to help them translate the system logic into properly functioning program modules. These tools include entity-relationship diagrams, flowcharts, pseudocode, decision tables, and decision trees.

If an agile development approach is used, then the customer creates user stories that describe required features and priority levels. In agile methodology, new system releases are made after many iterations and each is test-driven carefully by the customer.

Cohesion measures a module's scope and processing characteristics. A module that performs a single function or task has a high degree of cohesion, which is desirable. Coupling measures relationships and interdependence among modules. Modules that are relatively independent are loosely coupled, which is desirable. Cohesion and coupling concepts are used in structured development, but also are applicable to object-oriented development.

Typically, you follow four steps when you create a structure chart. You review DFDs and object models to identify the processes and methods, identify the program modules and determine control-subordinate relationships, add symbols for couples and loops, and analyze the structure chart to ensure that it is consistent with your system documentation.

Programmers perform desk checking, code review, and unit testing tasks during application development. Systems analysts design the initial test plans, which include test steps and test data for integration testing and system testing. Integration testing is necessary for programs that interact. The final step is system testing for the completed system. System testing includes users in the testing process.

In addition to system documentation, analysts and technical writers also prepare operations documentation and user documentation. Operations documentation provides instructions and information to the IT operations group. User documentation consists of instructions and information for users who interact with the system and includes user manuals, help screens, and tutorials.

During the installation process, you establish an operational, or production, environment for the new information system that is completely separate from the test environment. The operational environment contains live data and is accessible only by authorized users. All future changes to the system must be verified in the test environment before they are applied to the operational environment.

Everyone who interacts with the new information system should receive training appropriate to his or her role and skills. The IT department usually is responsible for training. Software or hardware vendors or professional training organizations also can provide training. When you develop a training program, remember the following guidelines: Train people in groups; utilize people already trained to help train others; develop separate programs for distinct employee groups; and provide for learning by using discussions, demonstrations, documentation, training manuals, tutorials, Webinars, and podcasts. Users learn better with interactive, self-paced training methods.

Data conversion often is necessary when installing a new information system. When a new system replaces a computerized system, you should automate the data conversion process if possible. The old system might be capable of exporting data in a format that the new system can use, or you might have to extract the data and convert it to an acceptable format. Data conversion from a manual system often requires labor-intensive data entry or scanning. Even when data conversion can be automated, a new system often requires additional data items, which might require manual entry. Strict input controls are important during the conversion process to protect data integrity and quality. Typically, data is verified, corrected, and updated during the conversion process.

System changeover is the process of putting the new system into operation. Four changeover methods exist: direct cutover, parallel operation, pilot operation, and phased operation. With direct cutover, the old system stops and the new system starts simultaneously; direct cutover is the least expensive, but the riskiest changeover method. With parallel operation, users operate both the old and new information systems for some period of time; parallel operation is the most expensive and least risky of the changeover methods. Pilot operation and phased operation represent compromises between direct cutover and parallel operation; both methods are less risky than direct cutover and less costly than parallel operation. With pilot operation, a specified group within the organization uses the new system for a period of time, while the old system continues to operate for the rest of the users. After the system proves successful at the pilot site, it is implemented throughout the organization. With phased operation, you implement the system in the entire organization, but only one module at a time, until the entire system is operational.

A post-implementation evaluation assesses and reports on the quality of the new system and the work done by the project team. Although it is best if people who were not involved in the systems development effort perform the evaluation, that is not always possible. The evaluation should be conducted early so users have a fresh recollection of the development effort, but not before users have experience using the new system.

The final report to management includes the final system documentation, describes any future system enhancements that already have been identified, and details the project costs. The report represents the end of the development effort and the beginning of the new system's operational life.

Key Terms and Phrases

acceptance tests *527*
application development *511*
attributes *519*
bug tracking software *529*
Capability Maturity Model (CMM)® *508*
Capability Maturity Model Integration (CMMI)® *508*
code review *525*
coding *523*
cohesion *516*
condition *515*
control couple *515*
control module *514*
coupling *516*
customer *520*
data conversion *543*
data couple *515*
defect tracking software *529*
design walkthrough *525*
desk checking *525*
direct cutover *544*
documentation *528*
exporting *544*
flowchart *513*
integrated development environment (IDE) *523*
integration testing *526*
ISO 90003:2004 *510*
iteration cycle *522*
iteration planning meeting *522*
library module *514*
link testing *526*
logic errors *525*
loop *515*
loosely coupled *516*
methods *519*
modular design *514*
module *511*
object-oriented development (OOD) *518*
ODBC (Open Database Connectivity) *544*
online documentation *531*
operational environment *534*
operations documentation *529*
parallel operation *545*
parallel programming *522*
partitioning *514*
patches *529*
phased operation *546*
pilot operation *546*
pilot site *546*
podcast *537*
post-implementation evaluation *547*
process improvement *509*
production environment *534*
program documentation *529*
pseudocode *513*
quality assurance *508*
release plan *521*
simulation *543*
software engineering *508*
status flag *515*
structure chart *514*
structured walkthrough *525*
stub testing *526*
subordinate modules *514*
subscribers *537*
syntax errors *525*
system changeover *544*
system documentation *529*
system testing *527*
test data *526*
test-driven design *522*
test environment *534*
test plan *526*
tightly coupled *516*
top-down approach *514*
training plan *535*
train-the-trainer *539*
tutorial *537*
unit testing *525*
user documentation *530*
user story *521*
Webcast *537*
Webinar *537*

Learn It Online

Instructions: To complete the Learn It Online exercises, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the resources for this chapter, and click the link for the exercise you want to complete.

1 Chapter Reinforcement

TF, MC, and SA

Click one of the Chapter Reinforcement links for Multiple Choice, True/False, or Short Answer. Answer each question and submit to your instructor.

2 Flash Cards

Click the Flash Cards link and read the instructions. Type 20 (or a number specified by your instructor) in the Number of playing cards text box, type your name in the Enter your Name text box, and then click the Flip Card button. When the flash card is displayed, read the question and then click the ANSWER box arrow to select an answer. Flip through the Flash Cards. If your score is 15 (75%) correct or greater, click Print on the File menu to print your results. If your score is less than 15 (75%) correct, then redo this exercise by clicking the Replay button.

3 Practice Test

Click the Practice Test link. Answer each question, enter your first and last name at the bottom of the page, and then click the Grade Test button. When the graded practice test is displayed on your screen, click Print on the File menu to print a hard copy. Continue to take practice tests until you score 80% or better.

4 Who Wants To Be a Computer Genius?

Click the Computer Genius link. Read the instructions, enter your first and last name at the bottom of the page, and then click the Play button. When your score is displayed, click the PRINT RESULTS link to print a hard copy.

5 Wheel of Terms

Click the Wheel of Terms link. Read the instructions, and then enter your first and last name and your school name. Click the PLAY button. When your score is displayed on the screen, right-click the score and then click Print on the shortcut menu to print a hard copy.

6 Crossword Puzzle Challenge

Click the Crossword Puzzle Challenge link. Read the instructions, and then click the Continue button. Work the crossword puzzle. When you are finished, click the Submit button. When the crossword puzzle is redisplayed, submit it to your instructor.

SCR Associates Case Simulation Session 11: Systems Implementation

Overview

The SCR Associates case study is a Web-based simulation that allows you to practice your skills in a real-world environment. The case study transports you to SCR's intranet, where you complete 12 work sessions, each aligning with a chapter. As you work on the case, you will receive e-mail and voice mail messages, obtain information from SCR's online libraries, and perform various tasks.

How do I use the case?

- Review the SCR background material in Chapter 1.
- Read the Preview for this session and study the Task List.
- Visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the **SCR Case Simulation**, and locate the intranet link.
- Enter your name and the password **sad9e**. An opening screen will display the 12 sessions.
- Select this session. Check your e-mail and voice mail carefully, and then work on the tasks.

Preview: Session 11

You assisted your supervisor, Jesse Baker, in planning an architecture for the new TIMS system. Now she wants you to work on system implementation tasks and issues. Specifically, she wants you to develop a structure chart, testing and training plans, implementation guidelines, and a post-implementation review.

Task List

1. *Jesse wants to see a structure chart. She said to use program modules based on the processes we identified earlier. She wants the modules to be cohesive and loosely coupled.*

2. *Need a testing plan that includes unit integration and system testing as Jesse requested. Also, need a reminder about careful documentation.*

3. *Jesse says it's important to understand the difference between the test and operational environments, and she wants me to draft a message that explains the installation process and includes guidelines for all concerned. Also need to develop a training plan for TIMS, with groups that should receive training, the topics that should be covered, and training methods we might use.*

4. *Jesse wants me to recommend a data conversion plan and a changeover method for TIMS. She also wants a plan for post-implementation review, including fact-gathering methods, topics to cover, the timing of the review, and who should perform it.*

FIGURE 11-44 Task list: Session 11.

Chapter Exercises

Review Questions

1. Where does systems implementation fit in the SDLC, what tasks are performed during this phase, and why is quality assurance so important?
2. How are structured, object-oriented, and agile methods similar? How are they different?
3. Describe structure charts and symbols, and define cohesion and coupling.
4. Define unit testing, integration testing, and system testing.
5. What types of documentation does a systems analyst prepare, and what would be included in each type?
6. What is the purpose of an operational environment and a test environment?
7. Who must receive training before a new information system is implemented?
8. List and describe the four system changeover methods. Which one generally is the most expensive? Which is the riskiest? Explain your answers.
9. Who should be responsible for performing a post-implementation evaluation?
10. List the information usually included in the final report to management.

Discussion Topics

1. A supervisor states, "Integration testing is a waste of time. If each program is tested adequately, integration testing is not needed. Instead, we should move on to system testing as soon as possible. If modules don't interact properly, we'll handle it then." Do you agree or disagree with this comment? Justify your position.
2. Suppose you are a systems analyst developing a detailed test plan. Explain the testing strategies you will use in your plan. Will you use live or simulated data?
3. Using the Internet, locate an example of training for a software or hardware product. Write a brief summary of the training, the product, the type of training offered, and the cost of training (if available). Discuss your findings with the class.
4. Suppose that you designed a tutorial to train a person in the use of specific software or hardware, such as a Web browser. What specific information would you want to know about the recipient of the training? How would that information affect the design of the training material?

Projects

1. In this chapter, you learned about the importance of testing. Design a generic test plan that describes the testing for an imaginary system.
2. Design a generic post-implementation evaluation form. The form should consist of questions that you could use to evaluate any information system. The form should evaluate the training received and any problems associated with the program.
3. Create a one-page questionnaire to distribute to users in a post-implementation evaluation of a recent information system project. Include at least 10 questions that cover the important information you want to obtain.
4. Using the material in this chapter and your own Internet research, prepare a presentation on the pros and cons of agile development methods.

Apply Your Knowledge

The Apply Your Knowledge section contains four mini-cases. Each case describes a situation, explains your role in the case, and asks you to respond to questions. You can answer the questions by applying knowledge you learned in the chapter.

1 Sand and Surf Retailers

Situation:

Sand and Surf Retailers recently acquired several smaller companies to expand its chain of clothing outlets. To establish consistency for the current organization and future acquisitions, Sand and Surf decided to develop an in-house application called SPS (Standard Purchasing System). The SPS system would standardize purchasing practices for each Sand and Surf subsidiary and manage all purchasing information. System testing will be completed by the end of the week.

1. What types of documentation are needed for this application?
2. During application development, what steps should the IT staff follow to develop a structure chart?
3. What suggestions do you have for help screens and online tutorials?
4. What types of testing should be performed? What types of test data should be used?

2 Albatross Airfreight

Situation:

Albatross Airfreight specializes in shipping cargo via air across North America. In an effort to modernize, the company has begun to computerize manual business processes. The new IT infrastructure includes a computer system that tracks cargo from departure point to destination. At this point, the new system is ready for implementation. Systems analysts are modularizing the system, and programmers are ready to start coding the first modules.

1. What issues should systems analysts and programmers discuss before they proceed with the project?
2. As a systems analyst on this project, how would you describe your primary responsibilities, and how could you contribute to the quality of the finished product?
3. As a programmer, how would you describe your primary responsibilities, and how could you contribute to the quality of the finished product?
4. Will the use of structure charts be beneficial during this stage of development? Discuss the advantages of structure charts compared with flowcharts and pseudocode.

3 Victorian Creations

Situation:

Victorian Creations is a growing business that specializes in the reproduction of furniture from the Victorian era. Since 2006, sales have increased steadily. The original accounting system was a package from Peachtree Software, which initially ran on a stand-alone PC and later on a LAN. Now, the firm is preparing to install a powerful, scalable accounting package that can support the company's current and future operations. You have been asked to develop a training plan for users.

1. Who should receive training on the new software, and what topics should the training cover?
2. Investigate an accounting package such as Peachtree to learn if the product can convert data from other accounting programs.
3. What changeover strategy would you suggest for the new accounting system? Explain your answer.
4. When should a post-implementation review be scheduled? Explain your answer.

4 Calico Prints

Situation:

Calico Prints creates a wide range of fabrics and wallpapers. Recently the company updated its payroll software as an in-house development project. Users and IT staff members have completed a comprehensive training curriculum. The system has been in the operational environment for approximately six weeks, and no major problems have occurred. You are responsible for the post-implementation evaluation.

1. What are some techniques you might use to obtain an accurate evaluation?
2. What are some specific questions you would include in a questionnaire? Who should receive the questionnaire?
3. Are there general guidelines that apply to the timing of a post-implementation evaluation? What are they?
4. In your view, should the company schedule a post-implementation evaluation for the payroll system at this time? Give specific reasons for your answer.

Case Studies

Case studies allow you to practice specific skills learned in the chapter. Each chapter contains several case studies that continue throughout the textbook, and a chapter capstone case.

New Century Health Clinic

New Century Health Clinic offers preventive medicine and traditional medical care. In your role as an IT consultant, you will help New Century develop a new information system.

Background

You completed the systems design for the insurance system at New Century Health Clinic. The associates at the clinic have approved the design specification, and you hired two programmers, Bill Miller and Celia Goldring, to assist you with the programming and testing of the insurance system.

For Assignments 3 and 4, assume that a server and six client workstations have been purchased, installed, and networked in the clinic offices. You now are ready to begin installation and evaluation of the system.

Assignments

1. Plan the testing required for the system. You should consider unit, integration, and system testing in your test plan and determine who should participate in the testing. Also design the test data that you will use. Prepare a structure chart that shows the main program functions for the New Century system.
2. You have asked Anita Davenport, New Century's office manager, to contribute to the user manual for the insurance system. She suggested that you include a section of frequently asked questions (FAQs), which you also could include in the online documentation. Prepare 10 FAQs and answers for use in the printed user manual and context-sensitive Help screens. Also identify the specific people who will require training on the new system. Describe the type and level of training you recommend for each person or group.
3. Recommend a changeover method for New Century's system and justify your recommendation. If you suggest phased operation or pilot operation, specify the order in which you would implement the modules or how you would select a pilot workstation or location.
4. Should the associates perform a post-implementation evaluation? If an assessment is done, who should perform it? What options are available and which would you recommend?

PERSONAL TRAINER, INC.

Personal Trainer, Inc., owns and operates fitness centers in a dozen Midwestern cities. The centers have done well, and the company is planning an international expansion by opening a new "supercenter" in the Toronto area. Personal Trainer's president, Cassia Umi, hired an IT consultant, Susan Park, to help develop an information system for the new facility. During the project, Susan will work closely with Gray Lewis, who will manage the new operation.

Background

Susan finished work on system architecture issues, and her system design specification was approved. Now she is ready to address system implementation tasks, including quality assurance, structure charts, testing, training, data conversion, system changeover, and post-implementation evaluation.

Assignments

1. Identify the specific groups of people who need training on the new system. For each group, describe the type of training you would recommend and list the topics you would cover.
2. Suggest a changeover method for the new billing system and provide specific reasons to support your choice. If you recommend phased operation, specify the order in which you would implement the modules. If your recommendation is for pilot operation, specify the department or area you would select as the pilot site and justify your choice.
3. Develop a data conversion plan that specifies which data items must be entered, the order in which the data should be entered, and which data items are the most time-critical.
4. You decide to perform a post-implementation evaluation to assess the quality of the system. Who would you involve in the process? What investigative techniques would you use and why?

FANCIFUL CRYSTAL

Fanciful Crystal has produced fine crystal products for many years. The company once dominated the global market, but sales have declined recently. Last year, Fanciful Crystal rushed to implement a new Web-based system to boost sagging sales. Unfortunately, the online system was not tested thoroughly and experienced start-up problems. For example, customers complained about order mix-ups and overcharges for deliveries. You are a new system analyst with the company, and your supervisor asked you to investigate the problems.

1. Based on what you know about e-commerce, how would you have tested a new Web-based system?
2. Should ISO standards have been considered? Explain your answer.
3. What should Fanciful Crystal do in the future to avoid similar problems when developing new systems?
4. Three months after the system changeover, you perform a post-implementation evaluation. Prepare three evaluation forms for the new information system: one for users, one for managers, and one for the IT operations staff.

CHAPTER CAPSTONE CASE: SoftWear, Limited

SoftWear, Limited (SWL), is a continuing case study that illustrates the knowledge and skills described in each chapter. In this case study, the student acts as a member of the SWL systems development team and performs various tasks.

Background

The ESIP development team of Jane Rossman, Tom Adams, and Becky Evans started work on the new system, which they would develop as a Microsoft Access application in a client/server environment. Jane and Tom scheduled additional meetings with the consulting firm, True Blue Systems, while Becky started designing the user interface.

The ESIP system design included a server to interact with various SWL client workstations and an interface to the new payroll package from Pacific Software. The payroll package was implemented successfully on SWL's mainframe, and several payroll cycles were completed without any processing problems.

When the ESIP development team met on Monday morning, the members studied the overview that True Blue submitted, shown in Figure 11-45. Jane said they would use a top-down design approach. Their first step was to partition the system and break it down into a set of modules on a structure chart. Each module would represent a program or function to be performed by macros or Visual Basic procedures.

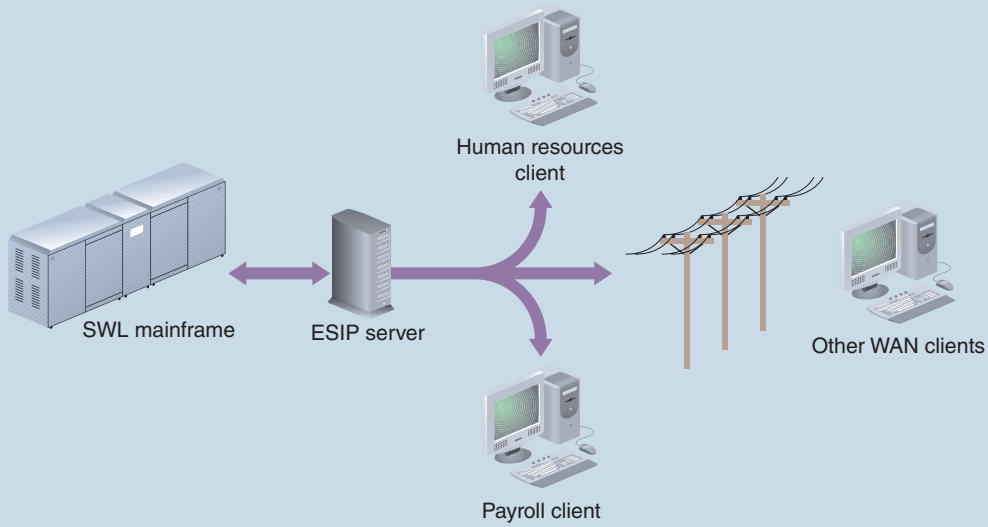


FIGURE 11-45 The ESIP system plan developed by True Blue Systems.

The team reviewed the documentation and DFDs carefully, using the DFDs they prepared during the systems analysis phase. The team determined that the ESIP system would perform five main tasks: Extract the ESIP deductions during payroll processing; apply the extracted deductions to specific ESIP options; update employee deduction selections; update ESIP option choices; and handle fund transfers to internal and external ESIP entities. To accomplish those tasks, the system would need a variety of reports, controls, query and display capabilities, input screens, security provisions, and other features.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

Of the five main ESIP processes, only the extracting of payroll deductions would be done on SWL's mainframe. Jane said they would need to develop an interface program to control the extraction processing, but all the other functions would run on the ESIP server and clients. By afternoon, they created a structure chart that showed a top-down model of all functions and processes.

Next, Jane estimated the time needed to design, code, unit test, and document each module. She also estimated the time needed for integration and system testing, completing the ESIP system documentation, and receiving management approval. Jane used Microsoft Project software to create a work breakdown structure and a critical path for the project. The development team decided to meet daily to review their progress.

Mainframe Interface

Jane met with Rick to discuss the ESIP deduction extraction program that executes when the mainframe runs the weekly payroll. Jane learned that she could write the extract module in Visual Basic.

Working together, Jane and Becky prepared a design, wrote the commands, and unit tested the ESIP program modules with test files. They used stubs to indicate the extraction program files. After verifying the results, Jane created a procedure for downloading the deduction file from the mainframe to the ESIP server in the payroll department. She tested the download procedure and updated the documentation.

ESIP Server

The team started developing the Access database application that would handle the other ESIP functions. The plan was for Becky to finish the basic switchboard and screen designs, then add features that users had requested, including custom menus and icons for frequently used functions. Becky also would design all the reports documented in the data dictionary.

Meanwhile, Jane and Tom reviewed the ERDs developed in the systems design phase to confirm that the entities and normalized record designs still were valid. Then they started to create objects, including tables, queries, macros, and code modules, using the application design tools in Access.

Jane and Tom defined the data tables, identified primary keys, and linked the tables into a relational structure, as shown in Figure 11-46. They reviewed the designs to ensure that the tables would work properly with Becky's input screens. Jane and Tom used an agreed-upon naming convention for all objects to ensure consistency among the systems.

After they loaded test data, Jane and Tom developed queries that would allow users to retrieve, display, update, and delete records. Some queries affected only individual records, such as ESIP options, while they designed other queries for specific reports, such as the ESIP Deduction Register query shown

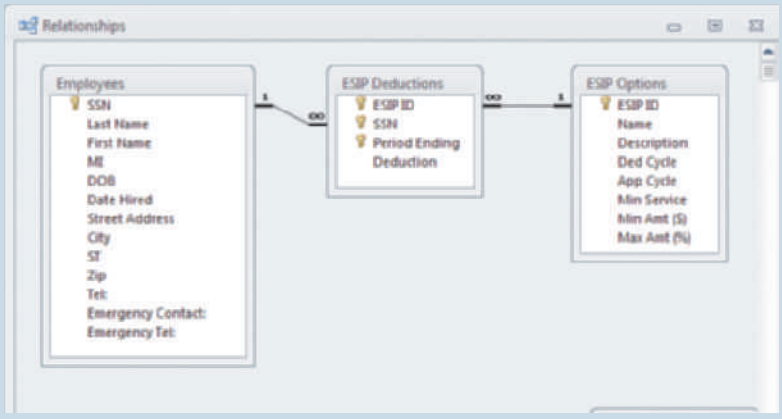


FIGURE 11-46 Relational structure for the ESIP system tables. Notice that the field or fields that comprise the primary key are shown in bold. Common fields link the tables, and referential integrity is indicated by the 1 (one) and ∞ (many) symbols.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

in Figure 11-47, which displays deductions first in date order, then by employee SSN. This query also produces SQL commands shown in the lower screen that will be transmitted to the ESIP server.

Jane and Tom also developed and tested macros that performed specific actions when certain events occurred. The macros later would be linked to various buttons and menus that Becky was designing into her switchboard and input screens. To save coding time, Tom converted several macros to Visual Basic in order to work on additional features and capabilities.

Jane and Tom also completed work on various security features, including password protection and several levels of permission required to view, modify, or delete specific objects. Later, when the system became operational, management would authorize specific permission levels, and Jane would designate a system administrator to maintain security and system policies.

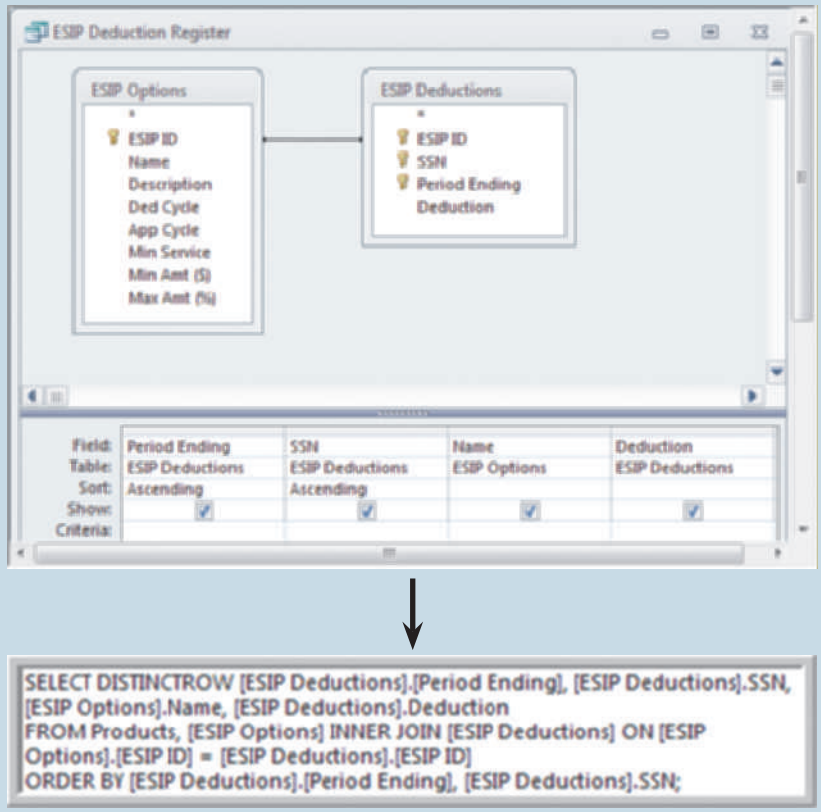


FIGURE 11-47 Example of a query that will provide data for the ESIP Deduction Register. Notice that the query sorts deductions by date, then by employee SSN. The query also produces SQL commands shown in the lower screen that will be transmitted to the ESIP server.

Completing Application Development

In three weeks, the ESIP development team finished unit testing. Becky tested the switchboard, macros, queries, screen forms, menus, submenus, and code modules to ensure that they functioned correctly. Next, they linked the modules and performed integration testing. The testing ran smoothly, and they encountered no significant problems.

After integration testing, the analysts asked several principal users to participate in system testing. During this process, the users suggested some minor changes to screens and reports, which the team implemented.

The team members prepared user documentation as they completed each task. To produce a clear, understandable user manual, they decided to ask Amy Calico, SWL's payroll director, to review their notes and help them write a draft for current and future users. They wanted to explain the system in nontechnical terms, with screen shots and a set of frequently asked questions. Jane said that the entire manual could be put online after SWL's intranet was developed.

Installation of the ESIP System

After a successful period of parallel operation, SWL fully implemented the payroll package purchased from Pacific Software and was ready to start installation of the ESIP system. In preparation, the IT development team of Jane Rossman, Tom Adams, and Becky Evans confirmed that SWL's existing network could handle the additional traffic generated by the new system, but might need to be upgraded in the future.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

Tom's first task was to install the ESIP application on the server in the payroll department and to verify that the system could communicate properly with SWL's mainframe. Then, he installed and tested a new high-speed tape cartridge backup system for the ESIP system.

Next, Tom loaded the ESIP application on a client PC in the human resources department. He checked all hardware and system software settings and used several test files to ensure that the client communicated with the ESIP server in the payroll department.

Meanwhile, Becky Evans and Rick Williams worked together on the interface between the ESIP system and the mainframe. They previously created a module called an extract program that directed the mainframe payroll system to capture the ESIP payroll deductions, store them in a file, and transmit the file back to the ESIP system. They already tested the interface using stubs to represent actual input and output files. Next, they would use a test data file with examples of every possible combination of permissible deductions and several improper deductions that testing should detect.

As soon as Rick confirmed that the payroll package was ready for the interface test, Tom set up the ESIP server and sent the test file to the mainframe. Then, he ran the module that sent processing commands to the payroll system. Everyone was pleased to see that the mainframe handled the test data properly and generated an extracted deduction file.

Becky and Rick were ready to conduct hands-on training with the payroll group, so they arranged an early morning session with Amy Calico, Nelson White, Britton Ellis, and Debra Williams. Becky walked them through the steps, which were described clearly in the user manual, and then answered several questions. The payroll employees seemed pleased with the explanation and commented on how much easier the new system would be for them to use.

Next, Becky went to see Mike Feiner, director of human resources. Mike would be the only person allowed to add, change, or delete any of the ESIP options. Based on written authorization from Rob King, vice president of human resources, Mike would have a special password and permission level to allow him to perform those actions. Becky described to Mike how the system worked and then showed him how to enter, modify, and delete a test option she prepared. For security reasons, the special documentation for those functions would not be printed in the user manual itself, but would be retained in the IT department files.

Becky and Tom met again with the payroll group to show users how to enter the deduction authorizations for individual employees. Although the new payroll system was operational, the company still handled ESIP deductions manually. Using the ESIP server and another networked payroll PC, the payroll clerks were able to enter actual payroll data during a three-day test period. The built-in edit and validation features detected the errors that the team purposely inserted as test data and even identified several invalid authorizations that they had not noticed previously. The group produced printed reports and asked other payroll department members to review and verify the output.

Jane suggested that they could send a notice to all SWL employees to describe the new ESIP system, remind employees how to select options, and invite their questions or comments. Michael Jeremy, vice president of finance, agreed with her suggestion.

Up to that point, the company had made no final decision about the changeover method for the new system. Because the ESIP system replaced a series of manual processing steps, the main question was whether they should run the manual system in parallel operation for a specified period. Managers in the payroll, human resources, and accounting departments all wanted the new system operational as soon as possible, so Jane decided to use a direct cutover method. The cutover was scheduled for the first Friday in May, when the first weekly payroll in May was processed.

Starting on April 23, the IT team met again with each of the users and reviewed a final checklist. No problems appeared, and the system was ready to interface with the mainframe and handle live data in an operational environment. On Friday morning, May 4, the payroll

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

department ran the ESIP module that sent the processing commands to the payroll system. Later that morning, during the weekly processing cycle, the payroll package created a file with the extracted deductions and passed it back to the ESIP server.

With the ESIP system using real input data, IT department members visited each of the recently trained users to make sure they were experiencing no difficulties. They received good reports — users in the payroll and human resources departments seemed pleased with the new system. They were able to access the ESIP data, enter new deductions, and had no problems with screen output or printed reports.

The direct cutover to the ESIP system occurred without major problems. By the end of June, the system had completed nine weekly payroll cycles, produced all required reports and outputs, and properly handled the monthly transfer of funds to the credit union and the SWL stock purchase plan.

During the first part of July, the IT department conducted a post-implementation evaluation with a team that consisted of two people: Katie Barnes, a systems analyst who had not been involved in the ESIP system development, and Ben Mancuso, a member of the finance department designated by Michael Jeremy. The evaluation team reviewed system operations, conducted interviews, and asked users to complete a brief questionnaire. The results were favorable, and it appeared that users were very satisfied with the new ESIP system. When the evaluation was completed in mid-July, Ann Hon sent the e-mail message shown in Figure 11-48 to Michael Jeremy. The systems development effort for the ESIP system was completed successfully.

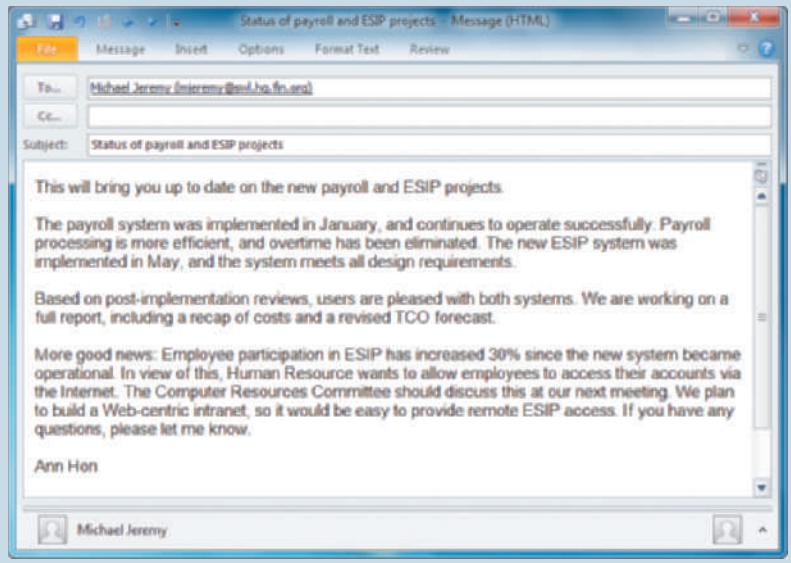


FIGURE 11-48 E-mail message from Ann Hon to Michael Jeremy regarding the payroll and ESIP systems.

SWL Team Tasks

1. Rick has asked you to help him develop a recommendation for future software testing. One of his concerns is that in systems testing, it is virtually impossible to simulate every system transaction and function. Specifically, Rick wants you to suggest some guidelines that will produce the most reliable test results. He suggested that you should consider who should be involved in the testing, what types of transactions should be tested, and when the testing should be done. Based on what you already have learned about the ESIP system in earlier chapters, what would you recommend?
2. Rick also wants you to prepare a list of interview questions for users and design a brief questionnaire that measures the effectiveness of the new system. Develop the interview questions and questionnaire by following the guidelines suggested in this chapter and in Chapter 4, which discusses fact-finding techniques.
3. Rick is interested in using agile development methods for the next SWL project. He asked you to write a "Guide to Agile Methods" for other members of the IT team. The guide should be thorough, easy to understand, and should include your own research on the Internet.

CHAPTER CAPSTONE CASE: SoftWear, Limited (continued)

4. You know that Jane was under pressure from SWL managers to get the ESIP system up and running. You also know that not everyone on the IT staff agreed with her decision to use a direct cutover method. What are some of the disadvantages of direct cutover? What other methods might have been used? What would you have done?

Manage the SWL Project

You have been asked to manage SWL's new information system project. One of your most important activities will be to identify project tasks and determine when they will be performed. Before you begin, you should review the SWL case in this chapter. Then list and analyze the tasks, as follows:

LIST THE TASKS Start by listing and numbering at least ten tasks that the SWL team needs to perform to fulfill the objectives of this chapter. Your list can include SWL Team Tasks and any other tasks that are described in this chapter. For example, Task 3 might be to Identify training needs, and Task 6 might be to Develop training solutions.

ANALYZE THE TASKS Now study the tasks to determine the order in which they should be performed. First identify all concurrent tasks, which are not dependent on other tasks. In the example shown in Figure 11-49, Tasks 1, 2, 3, 4, and 5 are concurrent tasks, and could begin at the same time if resources were available.

Other tasks are called dependent tasks, because they cannot be performed until one or more earlier tasks have been completed. For each dependent task, you must identify specific tasks that need to be completed before this task can begin. For example, you would want to identify training needs before you could develop training solutions, so Task 6 cannot begin until Task 3 is completed, as Figure 11-49 shows.

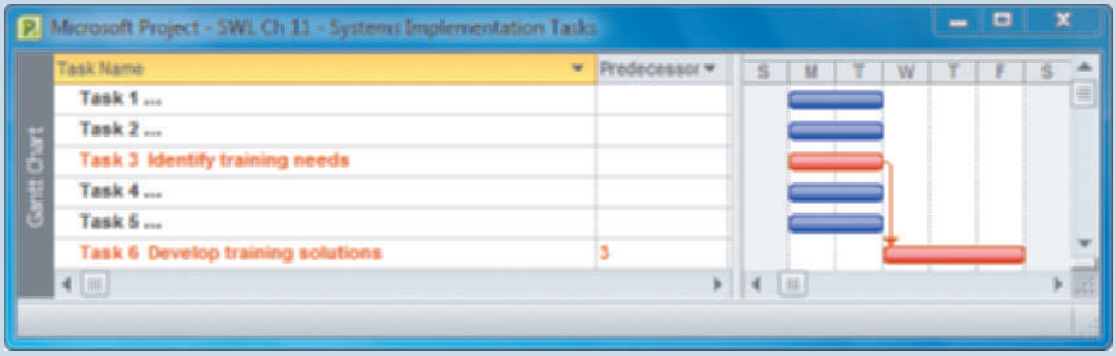


FIGURE 11-49 Tasks 1, 2, 3, 4, and 5 are concurrent tasks that could be performed at the same time. Task 6 is a dependent task that cannot be performed until Task 3 has been completed.

Chapter 3 describes project management tools, techniques, and software. To learn more, you can use the Features section on your Student Study Tool CD-ROM, or visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com** and locate the project management resources library for this book. On the Web, Microsoft offers demo versions, training, and tips for using Project 2010. You also can visit the OpenWorkbench.org site to learn more about this free, open-source software.

Ready for a Challenge?

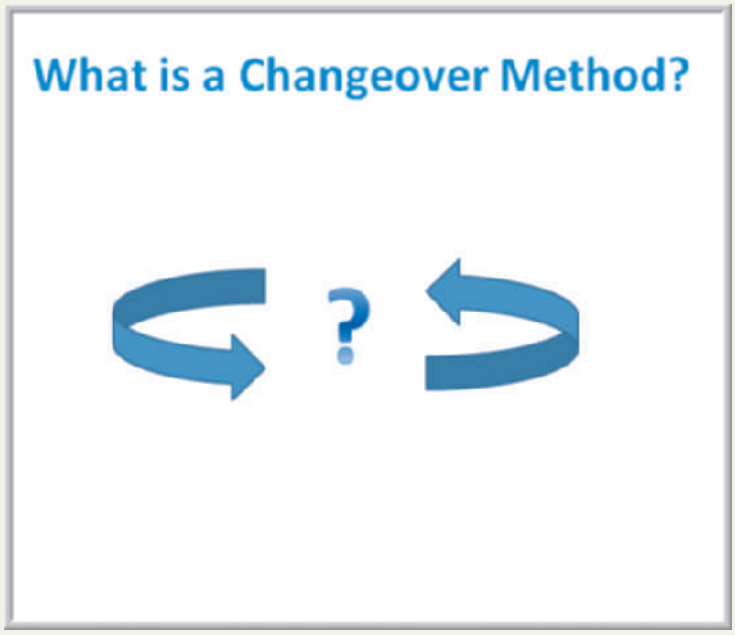
In additional to technical skills, IT professionals need critical thinking skills such as perception, organization, analysis, problem-solving, and decision-making. The Ready for a Challenge feature can help you learn, practice, and apply critical thinking skills that you can take to the workplace.

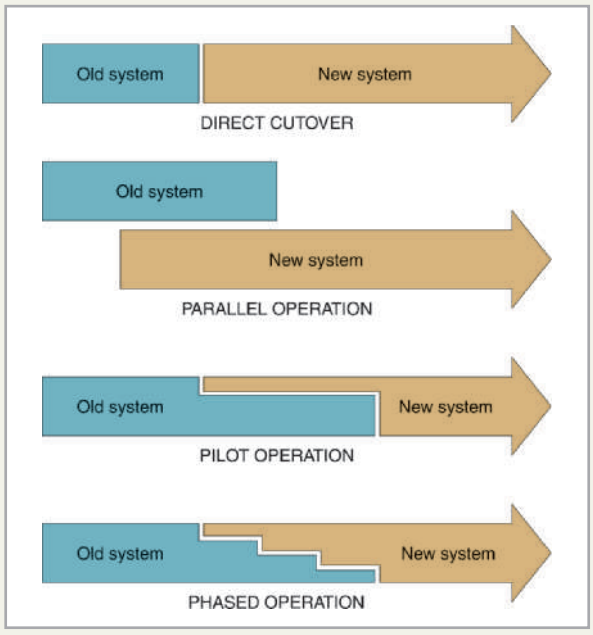
The IT team at Game Technology is working on implementation strategies for the new Customer Contact Care system (C^3). You are assigned to help with a structure chart, and to prepare a video tutorial about changeover methods. Before you start, you plan to review Chapter 11 of your textbook, including the Video Learning Session on structure charts.

First, you will work on the structure chart. You know that the system will rate contacts to identify the best sales prospects. The tasks are performed by a control module called Followup, which has three subordinate modules: List Contacts, Rate Contacts, and Produce Reports. Here are some details about the modules, data, and control couples:

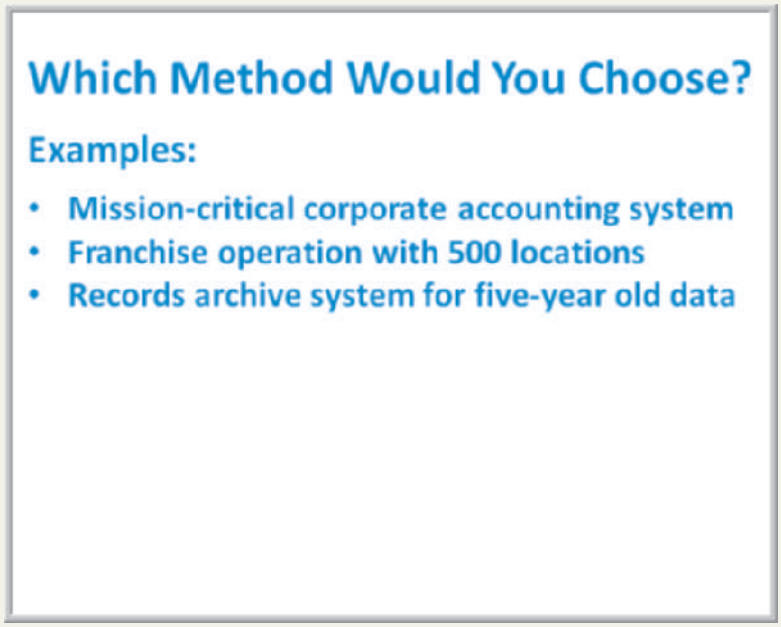
From	Data / Control Couple	To	Comments
Followup	Request Contact Data	List Contacts	Loop
List Contacts	Contact Data	Followup	
Followup	Contact Data	Rate Contacts	
Rate Contacts	Ratings	Followup	
Followup	Contact Data	Produce Reports	
List Contacts	End of File	Followup	Control Couple

You also need to work on the video tutorial. To get you started, your team prepared the first three slides, shown here and on the next page.





Ready for a Challenge? (Continued)



Practice Tasks

Study the table carefully and decide how to handle each of the data or control couples. Also review the material on video tutorial development. Now complete these tasks:

A. Create a structure chart using the information shown in the table.
B. Using the examples in the chapter, prepare a text narrative for each of the slides shown. Also include notes for the video developer and the narrator.

After you complete the Practice Tasks, to check your work and view sample answers, visit the Management Information Systems CourseMate Web site at **www.cengagebrain.com**, navigate to the resources for this chapter, and locate Ready for a Challenge?.

The Challenge

You just found out that the structure chart needs a change. The Produce Reports module will have a condition, and can send a Contact Rating to three of its own subordinate modules: High Potential, Medium Potential, and Low Potential.

Also, the team wants you to add a slide and narrative, similar to slide 3, with three additional examples.

Challenge Tasks

A. Add the necessary changes to the structure chart.
B. Create the additional slide and narrative. Show at least one example where phased changeover would be suitable.

This page intentionally left blank