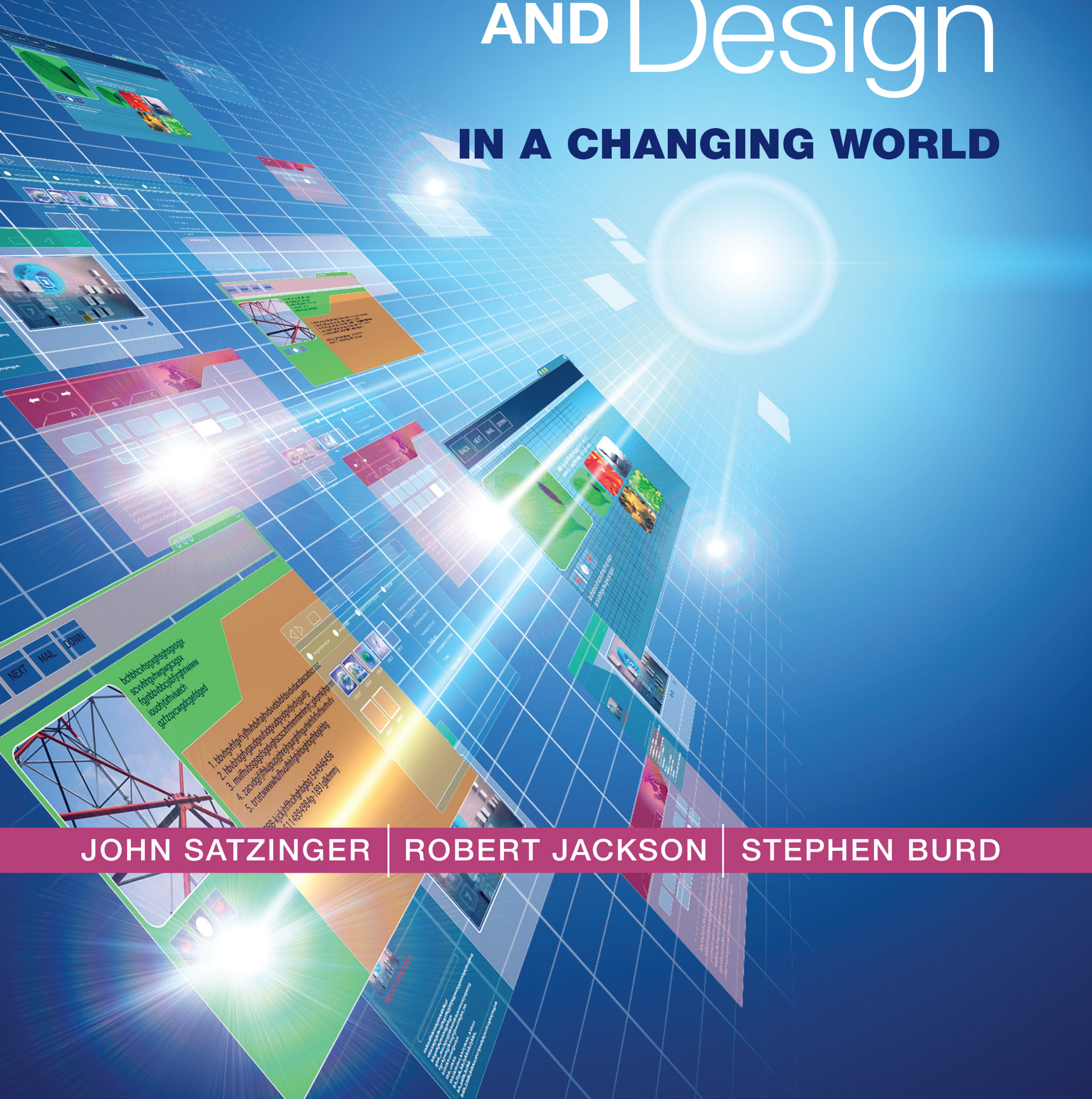
7e

Systems Analysis and Design

IN A CHANGING WORLD



JOHN SATZINGER | ROBERT JACKSON | STEPHEN BURD

SYSTEMS ANALYSIS AND DESIGN

In a Changing World

Seventh Edition

John W. Satzinger
Missouri State University

Robert B. Jackson
RBJ and Associates

Stephen D. Burd
University of New Mexico

CENGAGE Learning

Australia • Brazil • Mexico • Singapore • United Kingdom • United States

Systems Analysis and Design in a Changing World, Seventh Edition
John W. Satzinger, Robert B. Jackson, Stephen D. Burd

Vice President, General Manager, Social Science & Qualitative Business: Balraj Kalsi

Product Director: Joe Sabatino

Product Manager: Jason Guyler

Content Developer: Lori Bradshaw, S4Carlisle

Senior Product Assistant: Brad Sullender

Senior Market Manager: Eric La Scola

Marketing Coordinator: Will Guiliani

Art and Cover Direction, Production Management, and Composition:
Lumina Datamatics, Inc.

Intellectual Property Analyst: Christina Ciaramella

Senior Project Manager: Kathryn Kucharek

Manufacturing Planner: Ron Montgomery

Cover & Internal Image: Image Werks/Corbis

Uncredited figures are created by the authors.

Library of Congress Control Number: 2014958278

ISBN: 978-1-305-11720-4

Cengage Learning
20 Channel Center Street
Boston, MA 02210
USA

Printed in the United States of America
Print Number: 01 Print Year: 2014

DEDICATION

To my wife JoAnn—JWS

To my immediate and extended family—RBJ

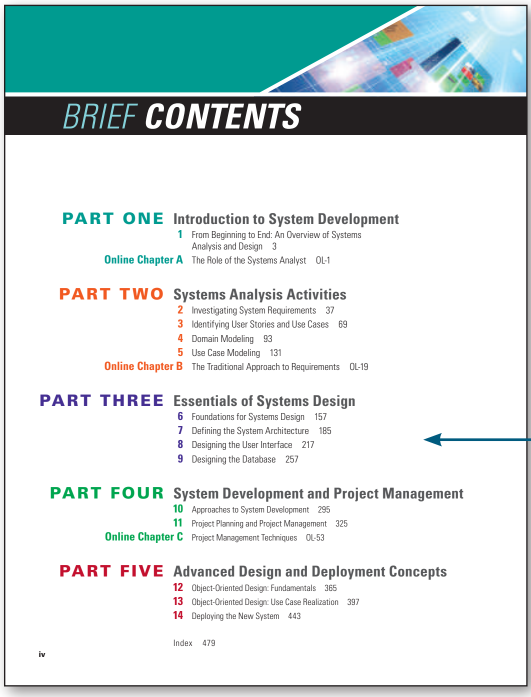
To Dee, Amelia, and Alex—SDB

BRIEF CONTENTS

CONTENTS

FEATURES

Systems Analysis and Design in a Changing World, Seventh Edition, was written and developed with instructor and student needs in mind. Here is just a sample of the unique and exciting features that help bring the field of systems analysis and design to life.



BRIEF CONTENTS

iv

The **innovative text organization** starts with a complete beginning-to-end system development example, moves immediately to systems analysis models and techniques, and then moves to systems design concepts emphasizing system architecture, user-interface design, and database design. Analysis and much of design is covered in the first nine chapters. Next, the text focuses on managing system development projects, including project planning and project management, after the student has a chance to learn what is involved in system development. Finally, the text covers detailed object-oriented design techniques and deployment topics.

The text uses a **completely updated** integrated case study of moderate complexity—**Ridgeline Mountain Outfitters (RMO)**—to illustrate key concepts and techniques. In addition, a smaller RMO application—the Tradeshow System—is used in Chapter 1 to introduce the entire system development process.

10 PART 1 ■ Introduction to System Development

FIGURE 1-6 RMO winter catalog

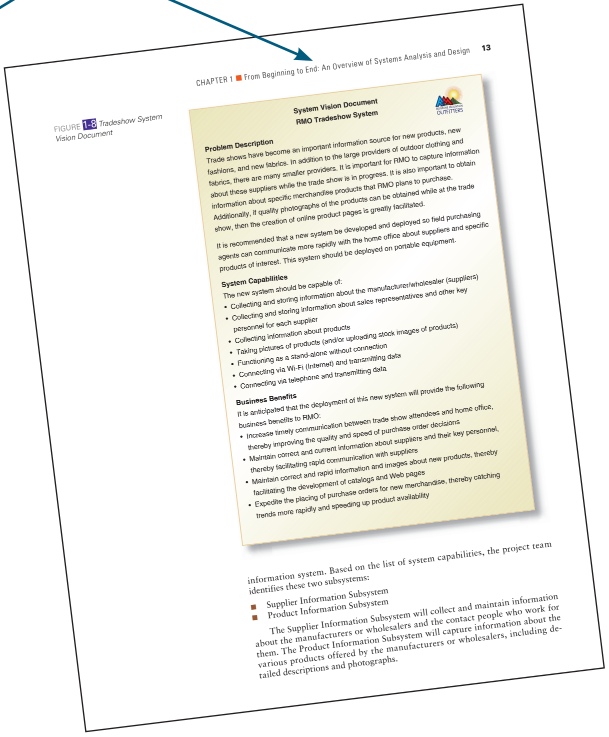
FIGURE 1-7 RMO online ordering home page

customers to go online to make purchases, so RMO continues to produce and mail abbreviated versions. **Figure 1-7** shows the RMO online ordering home page.

RMO produces its own line of outdoor clothing and sportswear. However, to offer a complete range of clothing in its retail outlets, it also sells brands of clothing sourced from other vendors. Furthermore, most accessories sold are sourced through vendors.

■ **Trade Shows**

To keep its product line innovative and responsive to consumer demand, RMO's purchasing agents attend apparel and accessory trade shows around the world



CHAPTER 1 ■ From Beginning to End: An Overview of Systems Analysis and Design 13

FIGURE 1-8 Tradeshow System Vision Document

System Vision Document
RMO Tradeshow System

Problem Description
Trade shows have become an important information source for new products, new fashions, and new fabrics. In addition to the large providers of outdoor clothing and fabrics, there are many smaller providers. It is important for RMO to capture information about these suppliers while the trade show is in progress. It is also important to obtain information about specific merchandise products that RMO plans to purchase. Additionally, if quality photographs of the products can be obtained while at the trade show, then the creation of online product pages is greatly facilitated.

It is recommended that a new system be developed and deployed so field purchasing agents can communicate more rapidly with the home office about suppliers and specific products of interest. This system should be deployed on portable equipment.

System Capabilities
The new system should be capable of:
- Collecting and storing information about the manufacturer/wholesaler (suppliers)
- Collecting and storing information about sales representatives and other key personnel for each supplier
- Collecting information about products
- Taking pictures of products (and/or uploading stock images of products)
- Functioning as a stand-alone without connection
- Connecting via Wi-Fi (Internet) and transmitting data
- Connecting via telephone and transmitting data

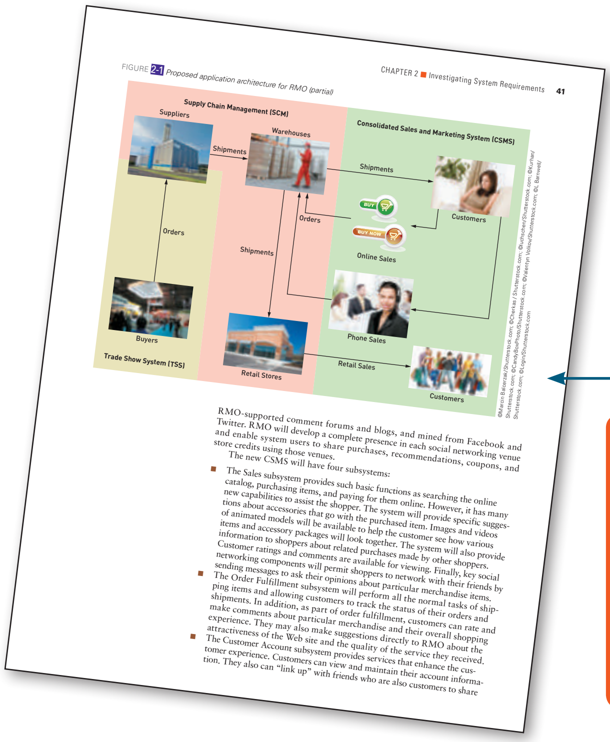
Business Benefits
It is anticipated that the deployment of this new system will provide the following business benefits to RMO:
- Increase timely communication between trade show attendees and home office, thereby improving the quality and speed of purchase order decisions
- Maintain correct and current information about suppliers and their key personnel, thereby facilitating rapid communication with suppliers
- Maintain correct and rapid information and images about new products, thereby facilitating the development of catalogs and Web pages
- Expedite the placing of purchase orders for new merchandise, thereby catching trends more rapidly and speeding up product availability

information system. Based on the list of system capabilities, the project team identifies these two subsystems:

- ■ Supplier Information Subsystem
- ■ Product Information Subsystem

The Supplier Information Subsystem will collect and maintain information about the manufacturers or wholesalers and the contact people who work for them. The Product Information Subsystem will capture information about the various products offered by the manufacturers or wholesalers, including detailed descriptions and photographs.

FEATURES



CHAPTER 2 ■ Investigating System Requirements 41
FIGURE 2-1 Proposed application architecture for RMO (partial)
Supply Chain Management (SCM)
Suppliers
Shipments
Warehouses
Orders
Shipments
Consolidated Sales and Marketing System (CSMS)
Shipments
Customers
Online Sales
Phone Sales
Retail Sales
Customers
Buyers
Trade Show System (TSS)
Retail Stores

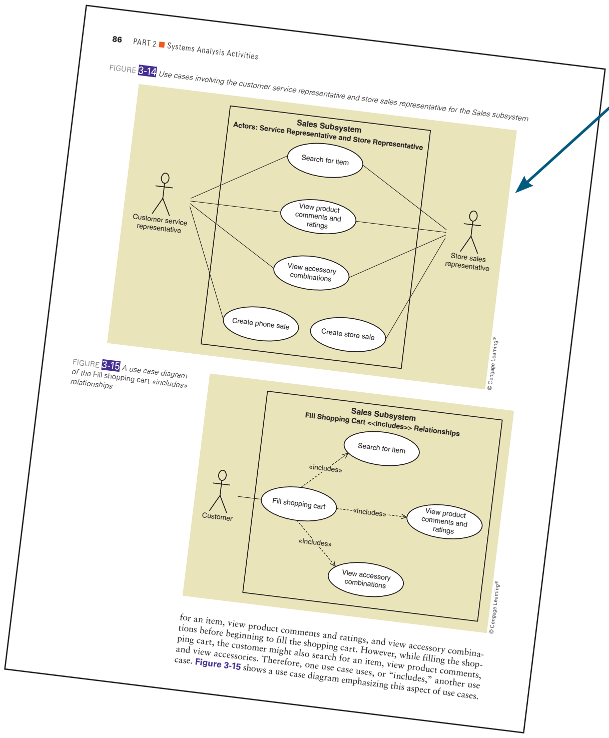
RMO-supported comment forums and blogs, and mined from Facebook and Twitter. RMO will develop a complete presence in each social networking venue and enable system users to share purchases, recommendations, coupons, and store credits using those venues.

The new CSMS will have four subsystems:

- The Sales subsystem provides such basic functions as searching the online catalog, purchasing items, and paying for them online. However, it has many new capabilities to assist the shopper. The system will provide specific suggestions about accessories that go with the purchased item. Images and videos of animated models will be available to help the customer see how various items and accessory packages will look together. The system will also provide information to shoppers about related purchases made by other shoppers. Customer ratings and comments are available for viewing. Finally, key social networking components will permit shoppers to network with their friends by sending messages to ask their opinions about particular merchandise items.
- The Order Fulfillment subsystem will perform all the normal tasks of shipping items and allowing customers to track the status of their orders and shipments. In addition, as part of order fulfillment, customers can rate and make comments about particular merchandise and their overall shopping experience. They may also make suggestions directly to RMO about the attractiveness of the Web site and the quality of the service they received.
- The Customer Account subsystem provides services that enhance the customer experience. Customers can view and maintain their account information. They also can "link up" with friends who are also customers to share

The planned **RMO application architecture** provides for rich examples—a Web-based component, a wireless smartphone/tablet application, and a client/server Windows-based component. All RMO applications described are integrated and strategically planned. The **Supply Chain Management System** already exists, ready for integrating the **Tradeshow System** and the new **Consolidated Sales and Marketing System.**

The new **Consolidated Sales and Marketing System (CSMS)** is the system development project described in Chapter 2 and used throughout the text for examples and explanations. It is strategically important to RMO, and the company must integrate the new system with legacy systems and other planned systems. There are **four subsystems,** and the requirements and design models are shown in detail. UML diagrams are used throughout for examples and exercises.



86 PART 2 ■ Systems Analysis Activities
FIGURE 3-14 Use cases involving the customer service representative and store sales representative for the Sales subsystem
Sales Subsystem
Actors: Service Representative and Store Representative
Search for item
View product comments and ratings
View accessory combinations
Create phone sale
Create store sale
Customer service representative
Store sales representative
FIGURE 3-15 A use case diagram of the Fill shopping cart «includes» relationships
Sales Subsystem
Fill Shopping Cart <<includes>> Relationships
Search for item
«includes»
Fill shopping cart
View product comments and ratings
View accessory combinations
Customer

for an item, view product comments and ratings, and view accessory combinations before beginning to fill the shopping cart. However, while filling the shopping cart, the customer might also search for an item, view product comments, and view accessories. Therefore, one use case uses, or "includes," another use case. Figure 3-15 shows a use case diagram emphasizing this aspect of use cases.

240 PART 3 ■ Essentials of Systems Design
FIGURE 8-20 RMO product detail search screen
FIGURE 8-21 Online retailer checkout page
Source: Amazon

Figure 8-21 is a sample screen of a Web-based application, where an item was recently added to the shopping cart. Generally, this is a good design, well organized and easily understood. However, it is somewhat limited in its use of navigation buttons. Frequently, a user would like to return to the search page to purchase a related item. Unfortunately, the only way to return to the original list of search results is with the Back button on the browser. A better design would be to include a Continue Shopping link back to the search page.

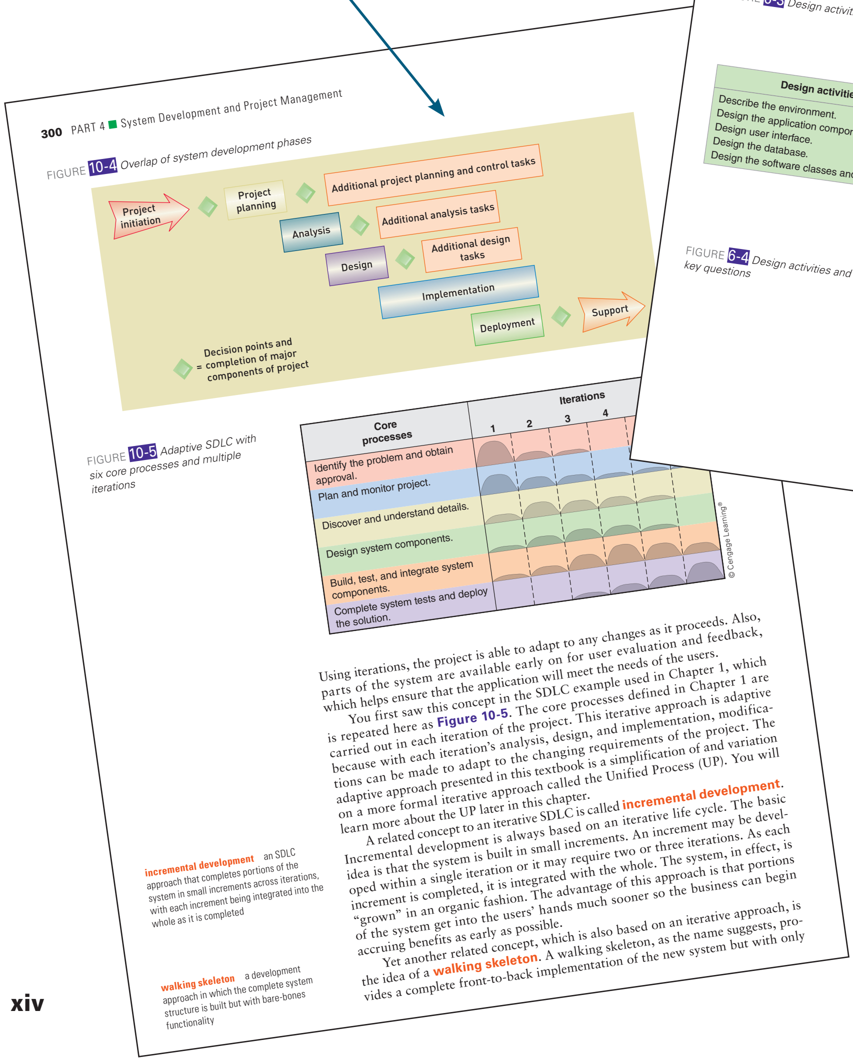
■ Considerations for Web-Based Applications

When designing the user interface for Web-based applications, all the preceding guidelines still apply. There are also two other important issues to be considered.

The first issue is that every aspect of the user interface must be transmitted over the Internet. With the speeds and bandwidth available today, even large

FEATURES

The text describes both **predictive and adaptive approaches to the SDLC** and recommends **Agile, iterative development** for most projects. The SDLC used in the text features a generic, condensed version of the Unified Process SDLC taught as an Agile approach that emphasizes iterations and core development processes. Core development processes and iterations are emphasized over phases to reduce the confusion that ordinarily occurs when students are taught "phases" and then told to use iterations. **Project planning and project management** are emphasized throughout, and the book focuses more on systems analysis and systems design as development disciplines rather than phases.



300 PART 4 ■ System Development and Project Management

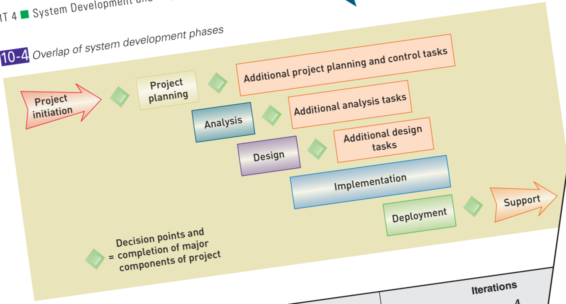

Project initiation
Project planning
Additional project planning and control tasks
Analysis
Additional analysis tasks
Design
Additional design tasks
Implementation
Deployment
Support
= Decision points and completion of major components of project

FIGURE 10-4 *Overlap of system development phases*

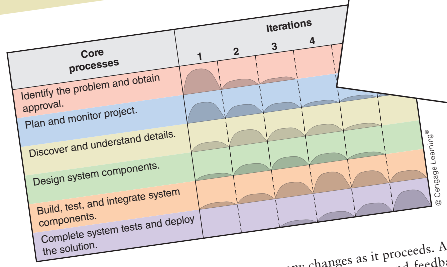


FIGURE 10-5 *Adaptive SDLC with six core processes and multiple iterations*

Using iterations, the project is able to adapt to any changes as it proceeds. Also, parts of the system are available early on for user evaluation and feedback, which helps ensure that the application will meet the needs of the users.

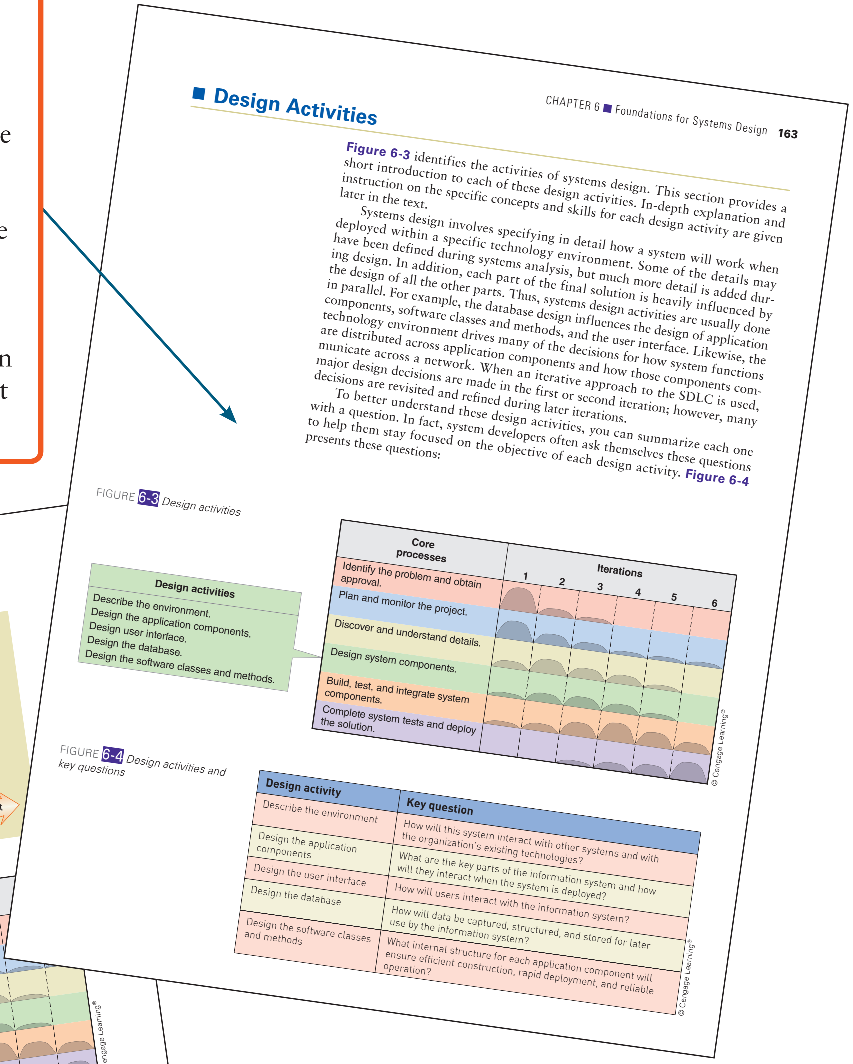
You first saw this concept in the SDLC example used in Chapter 1, which is repeated here as **Figure 10-5**. The core processes defined in Chapter 1 are carried out in each iteration of the project. This iterative approach is adaptive because with each iteration's analysis, design, and implementation, modifications can be made to adapt to the changing requirements of the project. The adaptive approach presented in this textbook is a simplification of and variation on a more formal iterative approach called the Unified Process (UP). You will learn more about the UP later in this chapter.

A related concept to an iterative SDLC is called **incremental development**. Incremental development is always based on an iterative life cycle. The basic idea is that the system is built in small increments. An increment may be developed within a single iteration or it may require two or three iterations. As each increment is completed, it is integrated with the whole. The system, in effect, is "grown" in an organic fashion. The advantage of this approach is that portions of the system get into the users' hands much sooner so the business can begin accruing benefits as early as possible.

Yet another related concept, which is also based on an iterative approach, is the idea of a **walking skeleton**. A walking skeleton, as the name suggests, provides a complete front-to-back implementation of the new system but with only

incremental development an SDLC approach that completes portions of the system in small increments across iterations, with each increment being integrated into the whole as it is completed

walking skeleton a development approach in which the complete system structure is built but with bare-bones functionality



■ Design Activities

CHAPTER 6 ■ Foundations for Systems Design 163

Figure 6-3 identifies the activities of systems design. This section provides a short introduction to each of these design activities. In-depth explanation and instruction on the specific concepts and skills for each design activity are given later in the text.

Systems design involves specifying in detail how a system will work when deployed within a specific technology environment. Some of the details may have been defined during systems analysis, but much more detail is added during design. In addition, each part of the final solution is heavily influenced by the design of all the other parts. Thus, systems design activities are usually done in parallel. For example, the database design influences the design of application components, software classes and methods, and the user interface. Likewise, the technology environment drives many of the decisions for how system functions are distributed across application components and how those components communicate across a network. When an iterative approach to the SDLC is used, major design decisions are made in the first or second iteration; however, many decisions are revisited and refined during later iterations.

To better understand these design activities, you can summarize each one with a question. In fact, system developers often ask themselves these questions to help them stay focused on the objective of each design activity. **Figure 6-4** presents these questions:

FIGURE 6-3 *Design activities*

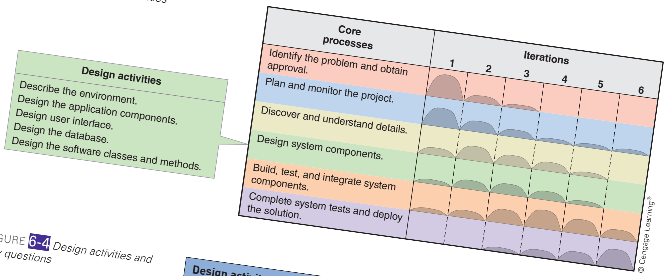


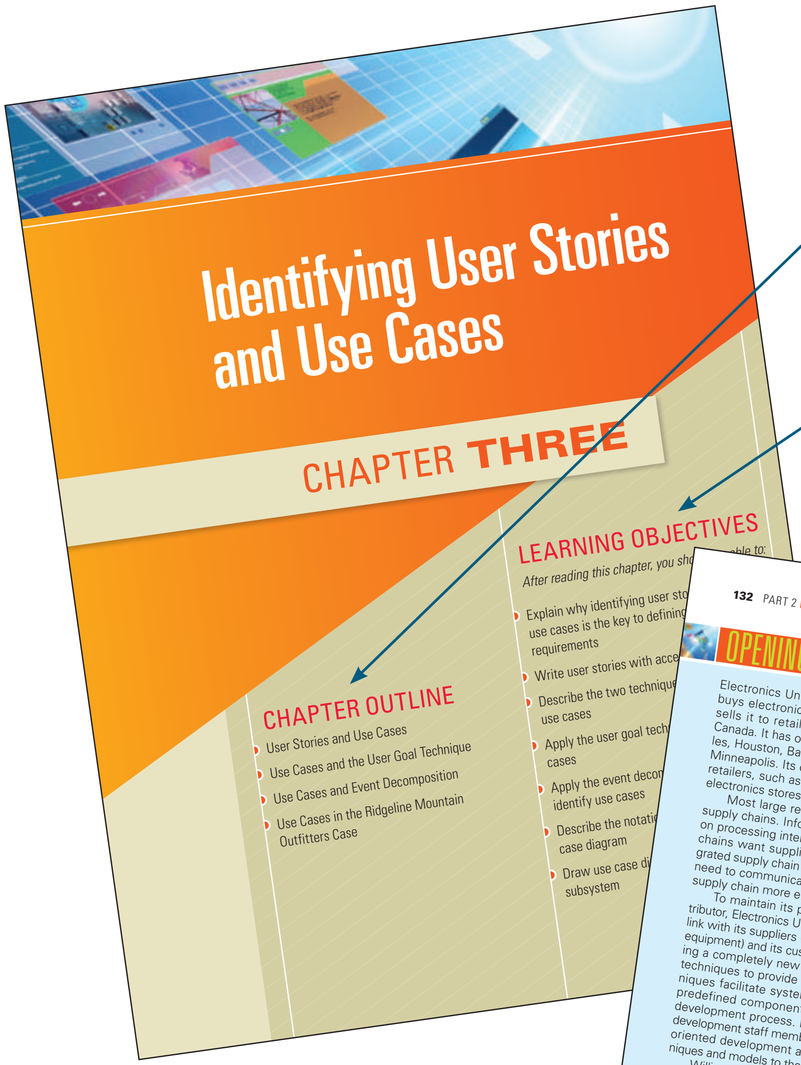
FIGURE 6-4 *Design activities and key questions*

Design activity	Key question
Describe the environment	How will this system interact with other systems and with the organization's existing technologies?
Design the application components	What are the key parts of the information system and how will they interact when the system is deployed?
Design the user interface	How will users interact with the information system?
Design the database	How will data be captured, structured, and stored for later use by the information system?
Design the software classes and methods	What internal structure for each application component will ensure efficient construction, rapid deployment, and reliable operation?

© Cengage Learning®

FEATURES

Each chapter provides a **chapter outline,** states clear **learning objectives,** and includes an **opening case study.**



Identifying User Stories and Use Cases

CHAPTER THREE

CHAPTER OUTLINE

- User Stories and Use Cases
- Use Cases and the User Goal Technique
- Use Cases and Event Decomposition
- Use Cases in the Ridgeline Mountain Outfitters Case

LEARNING OBJECTIVES

After reading this chapter, you sho… able to:

- Explain why identifying user sto… use cases is the key to defining… requirements
- Write user stories with acce…
- Describe the two technique… use cases
- Apply the user goal tech… cases
- Apply the event decom… identify use cases
- Describe the notati… case diagram
- Draw use case di… subsystem

132 PART 2 ■ Systems Analysis Activities

OPENING CASE **Electronics Unlimited: Integrating the Supply Chain**

Electronics Unlimited is a warehousing distributor that buys electronic equipment from various suppliers and sells it to retailers throughout the United States and Canada. It has operations and warehouses in Los Angeles, Houston, Baltimore, Atlanta, New York, Denver, and Minneapolis. Its customers range from large nationwide retailers, such as Target, to medium-sized independent electronics stores.

Most large retailers have moved toward integrated supply chains. Information systems used to be focused on processing internal data; however, today, these retail chains want suppliers to become part of a totally integrated supply chain system. In other words, the systems need to communicate between companies to make the supply chain more efficient.

To maintain its position as a leading wholesale distributor, Electronics Unlimited has to convert its system to link with its suppliers (the manufacturers of the electronic equipment) and its customers (the retailers). It is developing a completely new system that uses object-oriented techniques to provide these links. Object-oriented techniques facilitate system-to-system interfaces by using predefined components and objects to accelerate the development process. Fortunately, many of the system development staff members have experience with object-oriented development and are eager to apply the techniques and models to the system development project.

William Jones is explaining object-oriented development to the group of systems analysts who are being trained in this approach.

"We're developing most of our new systems by using object-oriented principles," he tells them. "The complexity of the new system, along with its interactivity, makes the object-oriented approach a natural way to develop requirements. The object-oriented models track very closely with the new object-oriented programming languages and frameworks."

William is just getting warmed up.

"This way of thinking about a system in terms of objects is very interesting," he adds. "It is also consistent with the object-oriented programming techniques you learned in your programming classes. You probably first learned to think about objects when you developed screens for the user interface. All the controls on the screen, such as buttons, text boxes, and drop-down boxes, are objects. Each has its own set of trigger events that activate its program functions."

"How does this apply to our situation?" one of the analysts asks.

"You just extend that thought process," William explains." You think of such things as purchase orders and employees as objects, too. We can call them the problem domain objects to differentiate them from user-interface objects, such as windows and buttons. During analysis, we have to find out all the trigger events and methods associated with each business object."

"And how do we do that?" another analyst asks.

"You continue with your fact-finding activities and build a better understanding of each use case," William says. "The way the problem domain objects interact with each other in the use case determines how you identify the initiating activity. We refer to those activities as the messages between objects. The tricky part is that you need to think in terms of objects instead of just processes. Sometimes, it helps me to pretend I am an object. I will say, 'I am a purchase order object. What functions and services are other objects going to ask me to do?' After you get the hang of it, it works very well, and it is enlightening to see how the system requirements unfold as you develop the diagrams."

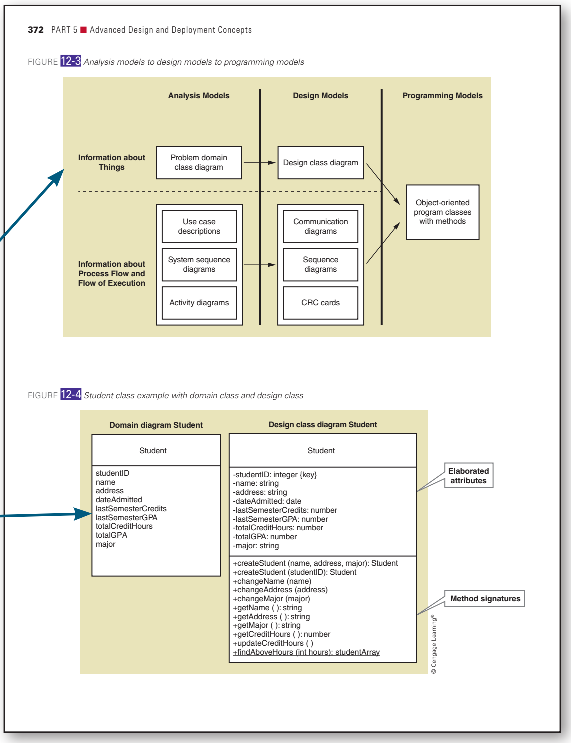
■ **Overview**

The main objective of defining requirements in system development is understanding users' needs, how the business processes are carried out, and how the system will be used to support those business processes. As indicated in Chapter 2, system developers use a set of models to discover and understand the requirements for a new system. This activity is a key part of systems analysis in the system development process. The first step in the process for developing this understanding requires the fact-finding skills you learned in Chapter 2. Fact-finding activities are also called *discovery activities*, and obviously, discovery must precede understanding.

The models introduced in Chapters 3 and 4 focus on two primary aspects of functional requirements: the use cases and the problem domain classes involved in users' work. User stories are sometimes used in place of use cases with Agile development. Use cases are identified by using the user goal technique and the

FEATURES

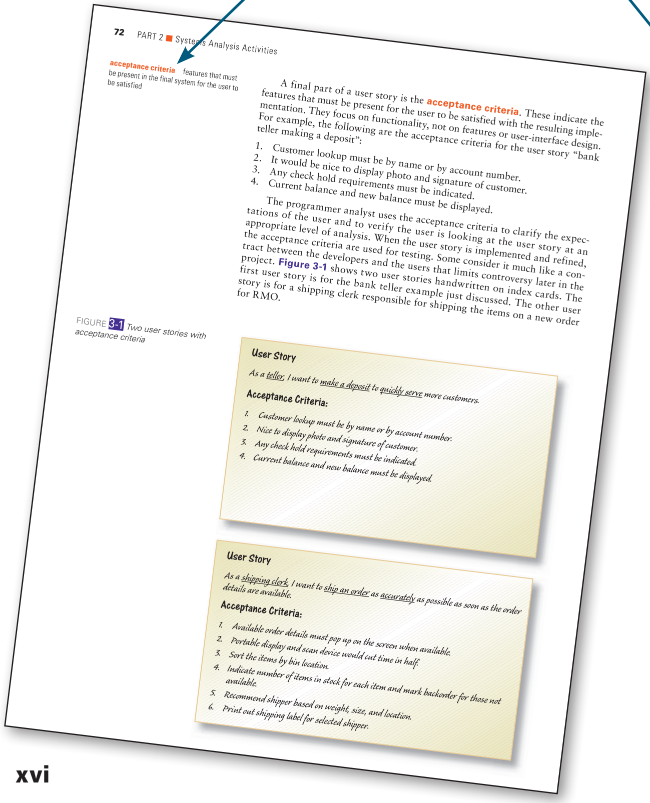
Margin definitions of key terms are placed in the text when a term is first used. Each chapter includes extensive **figures** and **illustrations** designed to clarify and summarize key points and to provide examples of **UML diagrams and other deliverables** produced by an analyst.



372 PART 5 ■ Advanced Design and Deployment Concepts

FIGURE 12-3 *Analysis models to design models to programming models*

FIGURE 12-4 *Student class example with domain class and design class*



72 PART 2 ■ Systems Analysis Activities

acceptance criteria features that must be present in the final system for the user to be satisfied

A final part of a user story is the **acceptance criteria**. These indicate the features that must be present for the user to be satisfied with the resulting implementation. They focus on functionality, not on features or user-interface design. For example, the following are the acceptance criteria for the user story "bank teller making a deposit":

1. Customer lookup must be by name or by account number.
2. It would be nice to display photo and signature of customer.
3. Any check hold requirements must be indicated.
4. Current balance and new balance must be displayed.

The programmer analyst uses the acceptance criteria to clarify the expectations of the user and to verify the user is looking at the user story at an appropriate level of analysis. When the user story is implemented and refined, the acceptance criteria are used for testing. Some consider it much like a contract between the developers and the users that limits controversy later in the project. **Figure 3-1** shows two user stories handwritten on index cards. The first user story is for the bank teller example just discussed. The other user story is for a shipping clerk responsible for shipping the items on a new order for RMO.

FIGURE 3-1 *Two user stories with acceptance criteria*

User Story

As a teller, I want to make a deposit to quickly serve more customers.

Acceptance Criteria:

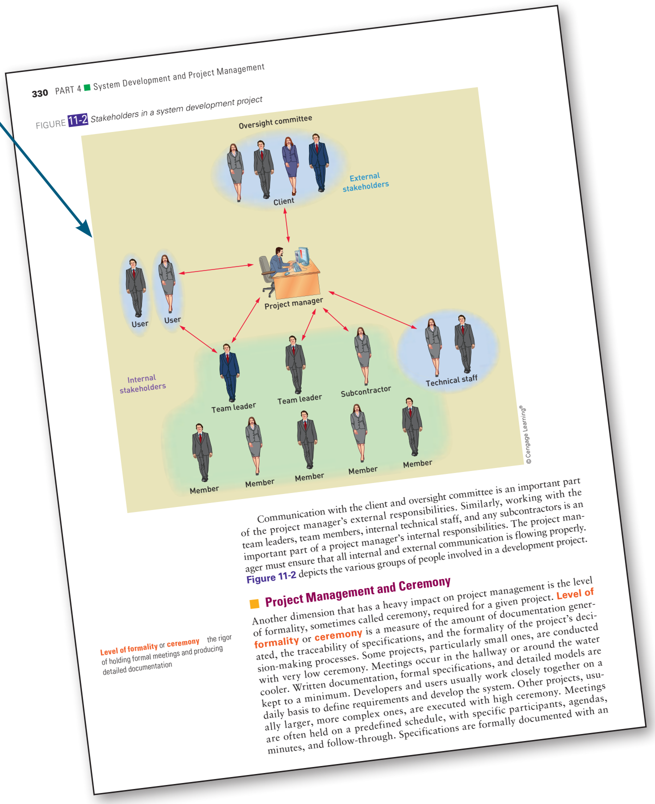
1. *Customer lookup must be by name or by account number.*
2. *Nice to display photo and signature of customer.*
3. *Any check hold requirements must be indicated.*
4. *Current balance and new balance must be displayed.*

User Story

As a shipping clerk, I want to ship an order as accurately as possible as soon as the order details are available.

Acceptance Criteria:

1. *Available order details must pop up on the screen when available.*
2. *Portable display and scan device would cut time in half.*
3. *Sort the items by bin location.*
4. *Indicate number of items in stock for each item and mark backorder for those not available.*
5. *Recommend shipper based on weight, size, and location.*
6. *Print out shipping label for selected shipper.*



330 PART 4 ■ System Development and Project Management

FIGURE 11-2 *Stakeholders in a system development project*

Communication with the client and oversight committee is an important part of the project manager's external responsibilities. Similarly, working with the team leaders, team members, internal technical staff, and any subcontractors is an important part of a project manager's internal responsibilities. The project manager must ensure that all internal and external communication is flowing properly. **Figure 11-2** depicts the various groups of people involved in a development project.

■ Project Management and Ceremony

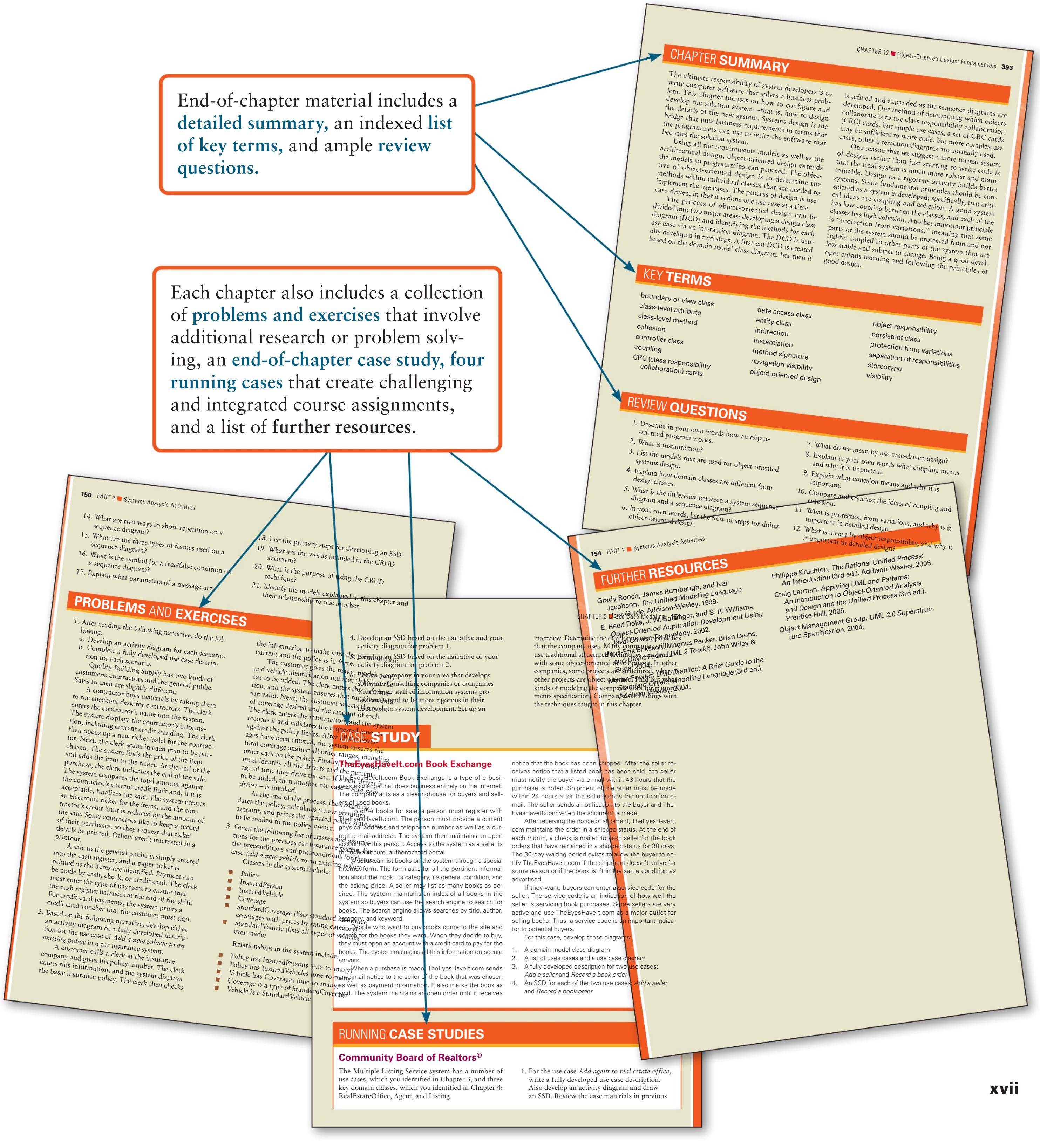
Level of formality or **ceremony** the rigor of holding formal meetings and producing detailed documentation

Another dimension that has a heavy impact on project management is the level of formality, sometimes called ceremony, required for a given project. **Level of formality** or **ceremony** is a measure of the amount of documentation generated, the traceability of specifications, and the formality of the project's decision-making processes. Some projects, particularly small ones, are conducted with very low ceremony. Meetings occur in the hallway or around the water cooler. Written documentation, formal specifications, and detailed models are kept to a minimum. Developers and users usually work closely together on a daily basis to define requirements and develop the system. Other projects, usually larger, more complex ones, are executed with high ceremony. Meetings are often held on a predefined schedule, with specific participants, agendas, minutes, and follow-through. Specifications are formally documented with an

FEATURES

End-of-chapter material includes a **detailed summary,** an indexed **list of key terms,** and ample **review questions.**

Each chapter also includes a collection of **problems and exercises** that involve additional research or problem solving, an **end-of-chapter case study, four running cases** that create challenging and integrated course assignments, and a list of **further resources.**



CHAPTER 12 ■ Object-Oriented Design: Fundamentals 393

CHAPTER SUMMARY

The ultimate responsibility of system developers is to write computer software that solves a business problem. This chapter focuses on how to configure and develop the solution system—that is, how to design the details of the new system. Systems design is the bridge that puts business requirements in terms that the programmers can use to write the software that becomes the solution system.

Using all the requirements models as well as the architectural design, object-oriented design extends the models so programming can proceed. The objective of object-oriented design is to determine the methods within individual classes that are needed to implement the use cases. The process of design is use-case-driven, in that it is done one use case at a time.

The process of object-oriented design can be divided into two major areas: developing a design class diagram (DCD) and identifying the methods for each use case via an interaction diagram. The DCD is usually developed in two steps. A first-cut DCD is created based on the domain model class diagram, but then it is refined and expanded as the sequence diagrams are developed. One method of determining which objects collaborate is to use class responsibility collaboration (CRC) cards. For simple use cases, a set of CRC cards may be sufficient to write code. For more complex use cases, other interaction diagrams are normally used.

One reason that we suggest a more formal system of design, rather than just starting to write code is that the final system is much more robust and maintainable. Design as a rigorous activity builds better systems. Some fundamental principles should be considered as a system is developed; specifically, two critical ideas are coupling and cohesion. A good system has low coupling between the classes, and each of the classes has high cohesion. Another important principle is "protection from variations," meaning that some parts of the system should be protected from and not tightly coupled to other parts of the system that are less stable and subject to change. Being a good developer entails learning and following the principles of good design.

KEY TERMS

boundary or view class	data access class	object responsibility
class-level attribute	entity class	persistent class
class-level method	indirection	protection from variations
cohesion	instantiation	separation of responsibilities
controller class	method signature	stereotype
coupling	navigation visibility	visibility
CRC (class responsibility collaboration) cards	object-oriented design	

REVIEW QUESTIONS

1. Describe in your own words how an object-oriented program works.
2. What is instantiation?
3. List the models that are used for object-oriented systems design.
4. Explain how domain classes are different from design classes.
5. What is the difference between a system sequence diagram and a sequence diagram?
6. In your own words, list the flow of steps for doing object-oriented design.
7. What do we mean by use-case-driven design?
8. Explain in your own words what coupling means and why it is important.
9. Explain what cohesion means and why it is important.
10. Compare and contrast the ideas of coupling and cohesion.
11. What is protection from variations, and why is it important in detailed design?
12. What is meant by object responsibility, and why is it important in detailed design?

154 PART 2 ■ Systems Analysis Activities

FURTHER RESOURCES

Grady Booch, James Rumbaugh, and Ivar Jacobson, *The Unified Modeling Language User Guide*. Addison-Wesley, 1999.

E. Reed Doke, J. W. Satzinger, and S. R. Williams, *Object-Oriented Application Development Using Java*. Course Technology, 2002.

Hans-Erik Eriksson, Magnus Penker, Brian Lyons, and David Fado, *UML 2 Toolkit*. John Wiley & Sons, 2004.

Martin Fowler, *UML Distilled: A Brief Guide to the Standard Object Modeling Language* (3rd ed.). Addison-Wesley, 2004.

Philippe Kruchten, *The Rational Unified Process: An Introduction* (3rd ed.). Addison-Wesley, 2005.

Craig Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process* (3rd ed.). Prentice Hall, 2005.

Object Management Group, *UML 2.0 Superstructure Specification*. 2004.

150 PART 2 ■ Systems Analysis Activities

14. What are two ways to show repetition on a sequence diagram?
15. What are the three types of frames used on a sequence diagram?
16. What is the symbol for a true/false condition on a sequence diagram?
17. Explain what parameters of a message are.
18. List the primary steps for developing an SSD.
19. What are the words included in the CRUD acronym?
20. What is the purpose of using the CRUD technique?
21. Identify the models explained in this chapter and their relationship to one another.

PROBLEMS AND EXERCISES

1. After reading the following narrative, do the following:
   a. Develop an activity diagram for each scenario.
   b. Complete a fully developed use case description for each scenario.

Quality Building Supply has two kinds of customers: contractors and the general public. Sales to each are slightly different.

A contractor buys materials by taking them to the checkout desk for contractors. The clerk enters the contractor's name into the system. The system displays the contractor's information, including current credit standing. The clerk then opens up a new ticket (sale) for the contractor. Next, the clerk scans in each item to be purchased. The system finds the price of the item and adds the item to the ticket. At the end of the purchase, the clerk indicates the end of the sale. The system compares the total amount against the contractor's current credit limit and, if it is acceptable, finalizes the sale. The system creates an electronic ticket for the items, and the contractor's credit limit is reduced by the amount of the sale. Some contractors like to keep a record of their purchases, so they request that ticket details be printed. Others aren't interested in a printout.

A sale to the general public is simply entered into the cash register, and a paper ticket is printed as the items are identified. Payment can be made by cash, check, or credit card. The clerk must enter the type of payment to ensure that the cash register balances at the end of the shift. For credit card payments, the system prints a credit card voucher that the customer must sign.

2. Based on the following narrative, develop either an activity diagram or a fully developed description for the use case of *Add a new vehicle to an existing policy* in a car insurance system.

A customer calls a clerk at the insurance company and gives his policy number. The clerk enters this information, and the system displays the basic insurance policy. The clerk then checks the information to make sure the policy is in force.

The customer gives the make, model, and vehicle identification number (VIN) of the car to be added. The clerk enters this information, and the system ensures that the data are valid. Next, the customer selects the types of coverage desired and the amount of each. The clerk enters the information and the system records it and validates the requested amounts against the policy limits. After all coverages have been entered, the system ensures the total coverage against all other ranges, including other cars on the policy. Finally, the customer must identify all the drivers and the percentage of time they drive the car. If a new driver is to be added, then another use case—*Add new driver*—is invoked.

At the end of the process, the system updates the policy, calculates a new premium amount, and prints the updated policy statement to be mailed to the policy owner.

3. Given the following list of classes and associations for the previous car insurance system, identify the preconditions and postconditions for the use case *Add a new vehicle to an existing policy*. Classes in the system include:
   - Policy
   - InsuredPerson
   - InsuredVehicle
   - Coverage
   - StandardCoverage (lists standard coverages with prices by rating category)
   - StandardVehicle (lists all types of vehicles ever made)

   Relationships in the system include:
   - Policy has InsuredPersons (one-to-many)
   - Policy has InsuredVehicles (one-to-many)
   - Vehicle has Coverages (one-to-many)
   - Coverage is a type of StandardCoverage
   - Vehicle is a StandardVehicle

CHAPTER 5 ■ Use Case Modeling

4. Develop an SSD based on the narrative and your activity diagram for problem 1.
5. Develop an SSD based on the narrative or your activity diagram for problem 2.
6. Find a company in your area that develops software. Consulting companies or companies with large staff of information systems professionals tend to be more rigorous in their approach to system development. Set up an interview. Determine the development practices that the company uses. Many companies still use traditional structured techniques combined with some object-oriented development. In other companies, some projects are structured, while other projects are object-oriented. Find out what kinds of modeling the company does for requirements specification. Compare your findings with the techniques taught in this chapter.

CASE STUDY

TheEyesHaveIt.com Book Exchange

TheEyesHaveIt.com Book Exchange is a type of e-business exchange that does business entirely on the Internet. The company acts as a clearinghouse for buyers and sellers of used books.

To offer books for sale, a person must register with TheEyesHaveIt.com. The person must provide a current physical address and telephone number as well as a current e-mail address. The system then maintains an open account for this person. Access to the system as a seller is through a secure, authenticated portal.

A seller can list books on the system through a special Internet form. The form asks for all the pertinent information about the book: its category, its general condition, and the asking price. A seller may list as many books as desired. The system maintains an index of all books in the system so buyers can use the search engine to search for books. The search engine allows searches by title, author, category, and keyword.

People who want to buy books come to the site and search for the books they want. When they decide to buy, they must open an account with a credit card to pay for the books. The system maintains all this information on secure servers.

When a purchase is made, TheEyesHaveIt.com sends an e-mail notice to the seller of the book that was chosen as well as payment information. It also marks the book as sold. The system maintains an open order until it receives notice that the book has been shipped. After the seller receives notice that a listed book has been sold, the seller must notify the buyer via e-mail within 48 hours that the purchase is noted. Shipment of the order must be made within 24 hours after the seller sends the notification e-mail. The seller sends a notification to the buyer and TheEyesHaveIt.com when the shipment is made.

After receiving the notice of shipment, TheEyesHaveIt.com maintains the order in a shipped status. At the end of each month, a check is mailed to each seller for the book orders that have remained in a shipped status for 30 days. The 30-day waiting period exists to allow the buyer to notify TheEyesHaveIt.com if the shipment doesn't arrive for some reason or if the book isn't in the same condition as advertised.

If they want, buyers can enter a service code for the seller. The service code is an indication of how well the seller is servicing book purchases. Some sellers are very active and use TheEyesHaveIt.com as a major outlet for selling books. Thus, a service code is an important indicator to potential buyers.

For this case, develop these diagrams:

1. A domain model class diagram
2. A list of uses cases and a use case diagram
3. A fully developed description for two use cases: *Add a seller* and *Record a book order*
4. An SSD for each of the two use cases: *Add a seller* and *Record a book order*

RUNNING CASE STUDIES

Community Board of Realtors®

The Multiple Listing Service system has a number of use cases, which you identified in Chapter 3, and three key domain classes, which you identified in Chapter 4: RealEstateOffice, Agent, and Listing.

1. For the use case *Add agent to real estate office*, write a fully developed use case description. Also develop an activity diagram and draw an SSD. Review the case materials in previous

PREFACE

When we wrote the first edition of this textbook, the world of system development was in a major transition period—from structured methodologies to object-oriented methodologies. We were among the first to introduce a comprehensive treatment of object-oriented methodologies, and ***Systems Analysis and Design in a Changing World, Seventh Edition***, continues to be the leader in teaching UML and object-oriented techniques.

However, change continues. Today, many new initiatives and trends have become firmly embedded in the world of system development. First and foremost is the ubiquitous access to the Internet throughout the global economy. The resulting explosion of connectivity means that project teams are now distributed around the world. In addition, large providers (such as Microsoft) and a proliferation of small providers now contribute to a wonderfully rich and varied software development environment.

In order to manage system development teams in today's distributed, fast-paced, connected, ever-changing environment, the techniques for software development and the approach to project management have expanded. Along with the foundational project management principles, additional approaches and philosophies provide new, success-oriented methodologies, such as Agile iterative, incremental development approaches. These are thoroughly covered in this edition.

Even though ***Systems Analysis and Design in a Changing World, Seventh Edition***, continues to be the leader in its field, with thorough treatment of such topics as user stories, use cases, object-oriented modeling, comprehensive project management, the Unified Modeling Language, and Agile techniques, it was time to take another step forward in textbook design. This edition uses an innovative approach to teaching systems analysis and design, taking advantage of the new teaching tools and techniques that are now available. As a result, not only is systems analysis and design easier to learn by using this approach, it is also easier to teach. It brings together the best approaches for teachers *and* students.

In this edition, we accomplish four major new objectives. First, we teach all the essential principles of system development—principles that must be followed in today's connected environment. Second, we teach and explain the new methodologies and techniques that are now available because of widespread connectivity. Third, we have organized and revamped the textbook so that it teaches these new concepts in a new way. Fourth, we created a set of short videos that explain key concepts and walk the reader through UML diagrams to help with understanding complex modeling.

For example, Chapter 1 presents a complete iteration in the development of a new system. Students get to see that complete iteration—from beginning to end (through implementation and testing)—before having to learn abstract principles or memorize terms. Also, the newly written running cases throughout the book focus on current issues of communication and connectedness and take the students through all aspects of system development. We have also expanded the *Instructor's Materials* and enhanced the aids available through CourseMate, our online resource. Additional online chapters are also available to enhance and extend the learning experience.

Finally, we updated and enhanced the set of ***over 30 short videos*** that explain key concepts in the text. These videos have been very well received and are even better with the new edition. These videos are useful for blended and online classes as well as traditional classes. The videos range from 3 to 10 minutes, and provide just-in-time explanations for often difficult to understand concepts, such as iterative development and Agile development, and illustrate important techniques such as identifying user stories and use cases. Most importantly, the videos show by demonstration how to read and interpret important UML models such as the domain model class diagrams, use case diagrams, sequence diagrams, and package diagrams. Understanding detailed UML models is finally possible in a way no other text can match.

We are excited about this new approach. The time is right for new materials and new tools for teaching systems analysis and design. Instructors will find this textbook intuitive, powerful, and easy to use. Students will find it engaging and empowering. Many concepts are presented so the students can teach themselves, with coaching and direction provided by the professor. It will be an rewarding experience to teach and learn with this textbook.

Innovations

This edition is innovative in many respects. It includes key concepts from traditional and object-oriented approaches, covers the use case-driven approach (with UML modeling being detailed in depth), emphasizes Agile and iterative development, and incorporates the latest concepts in Agile project management. Also, the material is completely reorganized to better support learning systems analysis and design.

Coverage of Object Orientation and Traditional Analysis and Design

This textbook is unique in its integration of key systems-modeling concepts that apply to the traditional structured approach and the object-oriented approach—user goals and events that trigger system use cases, plus classes of objects/data entities that are part of the system's problem domain. We devote one chapter to identifying user stories and use cases and another chapter to modeling key objects/entities, including coverage of entity-relationship diagrams, while emphasizing UML domain model class diagrams. After completing these chapters, instructors can cover structured analysis and design by including an online chapter, or they can focus on object-oriented analysis and design by using the chapters in this textbook. It is assumed from the beginning that everyone should understand the key object-oriented concepts. The traditional approach isn't discarded; key structured concepts are still included. But these days, most instructors are emphasizing the object-oriented approach.

Full Coverage of UML and the Object-Oriented Approach

The object-oriented approach presented in this textbook is based on the Unified Modeling Language (UML 2.0) from the Object Management Group, as originated by Grady Booch, James Rumbaugh, and Ivar Jacobson. A model-driven approach to analysis starts with user stories and use cases and then defines problem domain classes involved in the users' work. We include requirements modeling with use case diagrams, domain modeling, use case descriptions, activity diagrams, and system sequence diagrams. The FURPS+ model is used to emphasize functional and nonfunctional requirements.

Design principles and design patterns are discussed in depth, and system architecture is modeled by using UML component diagrams and package diagrams. Detailed design models are also discussed in detail, with particular attention given to use case realization with CRC cards, sequence diagrams, and design class diagrams.

Project Management Coverage

Many undergraduate programs depend on their systems analysis and design course to teach project management principles. To satisfy this need, we cover project management by taking a four-pronged approach. First, specific project management techniques, skills, and tasks are included and highlighted throughout this book. This integration teaches students how to apply specific project management tasks to the various activities of the system development life cycle (SDLC), including iterative development. Second, complete coverage of project planning and project management is included in a separate chapter. Third, we include a 120-day trial version of Microsoft Project Professional in the back of this book so students can obtain hands-on experience with this important tool. Fourth, a more in-depth treatment of project management techniques and principles is provided in an online chapter on this book's Web site. This information is based on the Project Management Body of Knowledge (PMBOK), as developed by the Project Management Institute—the primary professional organization for project managers in the United States.

Organized for More Effective Learning

This edition's innovative and entirely new organization starts with a complete beginning-to-end example of system development, moves immediately to systems analysis models and techniques, and then proceeds to system design concepts, emphasizing system architecture, user interfaces, and database design. The student sees analysis and much of design covered in the first nine chapters. Next, the text focuses on managing system development projects, including Agile development, after the student has had a chance to understand what is actually involved in system development. Finally, the text covers detailed design topics and deployment topics, going into more depth about such contemporary approaches as the Unified Process, Extreme Programming, and Scrum.

CourseMate Companion Web Site

Cengage Learning's ***Systems Analysis and Design in a Changing World, Seventh Edition***, CourseMate brings course concepts to life with interactive learning, study, and exam preparation tools that support the printed textbook. Watch student comprehension soar as your class works with the printed textbook and the textbook-specific Web site. CourseMate goes beyond the book to deliver what you need! Learn more at cengage.com/coursemate.

Engagement Tracker

How do you assess your students' engagement in your course? How do you know your students have read the material or viewed the resources you have assigned? How can you tell if your students are struggling with a concept? With CourseMate, you can use the included Engagement Tracker to assess student preparation and engagement. Use the tracking tools to see progress for the class as a whole or for individual students. Identify students at risk early in the course. Uncover which concepts are most difficult for your class. Monitor time on task. Keep your students engaged.

Interactive Teaching and Learning Tools

CourseMate includes interactive teaching and learning tools:

- Quizzes
- Case projects
- Flash cards
- Short videos on concepts, techniques, and models
- PowerPoint presentations

These assets enable students to review for tests, prepare for class, and address the needs of students' varied learning styles.

Interactive E-Book

In addition to interactive teaching and learning tools, CourseMate includes an interactive e-book. Students can take notes, highlight, search for, and interact with embedded media specific to their book. Use it as a supplement to the printed text or as a substitute—the choice is your students' with CourseMate.

Organization and Use

Systems Analysis and Design in a Changing World, Seventh Edition, includes this printed textbook, a complete e-book, and supporting online chapters. The current printed textbook provides a focused presentation of those topics that are essential and most important for information systems developers. The online chapters extend those concepts and provide a broader presentation of several topics. The online chapters may be integrated into the course or simply used as additional readings as prescribed by the instructor.

There are three major subject areas discussed in this book: systems analysis, systems design, and project management. There are additional subject areas, which are no less important but aren't discussed in as much depth. These include systems implementation, testing, and deployment. In addition, we have taken an approach that is quite different from other texts. Because students already have a basic understanding of systems analysis and design from Chapter 1, we immediately present in-depth concepts related to systems analysis and design. We present approaches to development and project management topics later in the text. This allows students to learn those project management concepts after understanding the elements of systems analysis and design. We think it will be more meaningful for students at that point in the course.

Part 1: Introduction to System Development

Part 1, comprising Online Chapter A and Chapter 1, presents an overview of system development. Online Chapter A, "The Role of the Systems Analyst," describes basic systems concepts and the role of the systems analyst in system development projects. Chapter 1 begins by briefly explaining the objectives of systems analysis and systems design. Then, it provides a detailed, concrete example of what is required in a typical software development project. Many students who take a programming class think that programming is all you need to develop software and deploy a system. This chapter and the rest of the book should dispel that myth.

Part 2: Systems Analysis Tasks

Chapters 2 through 5 cover systems analysis in detail. Chapter 2 discusses system requirements, analysis activities, and techniques for gathering information about the business problem. Developing the right system solution is possible only if the problem is accurately understood. Chapter 2 also explains how to identify and involve the stakeholders and introduces the concept of models and modeling. Chapters 3 and 4 teach modeling techniques for capturing the detailed requirements for the system in a useful form. When discussing an information system, two key concepts are particularly useful: the user stories/use cases that define what the end users need the system to do and the data entities/domain classes that users work with while carrying out their work tasks. These two concepts—user stories/use cases and data entities/domain classes—are important no matter what approach to system development is being used. Chapter 5 presents more in-depth requirements models, such as use case descriptions, activity diagrams, system sequence diagrams, and CRUD analysis.

Online Chapter B, "The Traditional Approach to Requirements," presents the traditional, structured approach to developing systems. To those instructors and students who desire to learn about data flow diagrams and structured English, this chapter provides an in-depth presentation.

All these modeling techniques provide in-depth analysis of user needs and allow the analyst to develop requirements and specifications. Again, the purpose of systems analysis is to thoroughly understand and specify the user's needs and requirements.

Part 3: Essentials of Systems Design

Part 3 provides the fundamental concepts related to systems design and designing the user experience. Chapter 6 provides broad and comprehensive coverage of important principles of systems design, including design activities and the crucial issues of system controls and security that all students should understand. It serves not only as a broad overview of design principles but also as a foundation for later chapters that explain the detailed techniques, tasks, skills, and models used to carry out design.

Chapter 7 provides a comprehensive overview of system architecture and is a new chapter that consolidates material previously spread out in multiple chapters. Chapter 8 presents additional design principles related to the user experience. Designing the user interface is a combination of analysis and design. It is related to analysis because it requires heavy user involvement and includes specifying user activities and desires. On the other hand, it is a design activity because it is creating specific final components that are used to drive the programming effort. The screens and reports and other user interaction components must be precisely designed so they can be programmed as part of the final system. Chapter 9 provides a compact and integrated coverage of designing the database.

Part 4: Projects and Project Management

By this point, students will have a basic understanding of all the elements of system development. Part 4 brings together all these concepts by explaining more about the process of organizing and managing development projects. Chapter 10 describes different approaches to system development in today's environment, including Agile development and several widely used development methodologies—the Unified Process, Extreme Programming, and Scrum. It is an important chapter to help you understand how projects actually get executed.

Chapter 11 extends these concepts by teaching foundation principles of project planning and project management. Every systems analyst is involved in helping organize, coordinate, and manage software development projects. In addition, most good students will eventually become team leaders and project managers. The principles presented in Chapter 11 are essential to a successful career.

Online Chapter C, “Project Management Techniques,” goes into more detail regarding the tools and techniques used by systems analysts and project managers to plan and monitor development projects. For those instructors and students who would like to learn specific project management skills, this is an important chapter.

Part 5: Advanced Design and Deployment Concepts

Part 5 goes into more depth with respect to systems design, specifically object-oriented software design, and other important issues related to effective and successful system development and deployment.

Chapters 12 and 13 explain in detail the models, skills, and techniques used to design software systems. As mentioned earlier, systems design is a fairly complex activity, especially if it is done correctly. The objective of these two chapters is to teach the student the various techniques—from simple to complex—that can be used to effectively design software systems.

Chapter 14 describes the final elements in system development: final testing, deployment, maintenance, and version control.

Designing Your Analysis and Design Course

There are many approaches to teaching analysis and design courses, and the objectives of the course differ considerably from college to college. In some academic information systems departments, the analysis and design course is a capstone course in which students apply the material learned in prior database, networking, and programming courses to a real analysis and design project. In other information systems departments, analysis and design is used as an introduction to the field of system development and is taken prior to more specialized courses. Some information systems departments offer a two-course sequence emphasizing analysis in the first semester and design and implementation in the second semester. Some information systems departments have only one course that covers analysis and design.

The design of the analysis and design course is complicated even more by the choice of emphasizing some traditional and some object-oriented content—again, depending on local curriculum priorities. Additionally, the more iterative approach to development in general has made choices about sequencing the analysis and design topics more difficult. For example, with iterative development, a two-course sequence can’t be divided into analysis and then design as easily.

The objectives, course content, assignments, and projects have many variations. What we offer below are some suggestions for using this textbook in various approaches to the course.

UML and Object-Oriented Analysis and Design Course

This is the course we designed the printed textbook to support, so all the printed chapters but none of the online chapters are included. Note that object-oriented design is included in detail. The course covers object-oriented analysis and design, user and system interface design, database design, controls and security, and implementation and testing. It is usually assumed that the projects will use custom development, including Web development. The course emphasizes iterative development with three-layer architecture, project management, information gathering, and management reporting. One-semester courses are usually limited to completing some prototypes of the user interface to give students closure. Sometimes, this course is spread over two semesters, with some implementation of an actual system in the second semester for a more complete development experience. Iterative development is emphasized.

A suggested outline for a course emphasizing object-oriented development is:

Online Chapter A: The Role of the Systems Analyst (optional)
Chapter 1: From Beginning to End: An Overview of Systems Analysis and Design
Chapter 2: Investigating System Requirements
Chapter 3: Identifying User Stories and Use Cases
Chapter 4: Domain Modeling
Chapter 5: Use Case Modeling
Chapter 6: Foundations for Systems Design
Chapter 7: Defining the System Architecture
Chapter 8: Designing the User Interface
Chapter 9: Designing the Database
Chapter 10: Approaches to System Development
Chapter 11: Project Planning and Project Management
Chapter 12: Object-Oriented Design: Fundamentals
Chapter 13: Object-Oriented Design: Use Case Realization
Chapter 14: Deploying the New System

Traditional Analysis and Design Course

A traditional systems analysis and design course provides coverage of activities and tasks by using structured analysis, user and system interface design, database design, controls and security, and implementation and testing. It is usually assumed that the project will use custom development, including Web development. The course emphasizes the SDLC, project management, information gathering, and management reporting. One-semester courses are usually limited to completing some prototypes of the user interface to give students closure. Sometimes, this course is spread over two semesters, with some implementation of an actual system in the second semester for a more complete development experience.

For this approach to the analysis and design course, a reasonable outline would omit chapters and sections detailing object orientation but include the online chapters on the role of the systems analyst and on traditional structured analysis. However, object-oriented concepts are introduced throughout the text, so students will still be familiar with them. Additionally, because of the amount of material to cover, the online chapter detailing project management, financial feasibility, and scheduling might be omitted.

A suggested outline for a course emphasizing the traditional structured approach is:

Online Chapter A: The Role of the Systems Analyst
Chapter 1: From Beginning to End: An Overview of Systems Analysis and Design
Chapter 2: Investigating System Requirements
Chapter 3: Identifying User Stories and Use Cases
Chapter 4: Domain Modeling
Online Chapter B: The Traditional Approach to Requirements
Chapter 6: Foundations for Systems Design
Chapter 8: Designing the User Interface
Chapter 9: Designing the Database
Chapter 10: Approaches to System Development
Chapter 11: Project Planning and Project Management
Chapter 14: Deploying the New System

In-Depth Analysis and Project Management

Some courses cover object-oriented systems analysis methods in more depth and briefly survey structured analysis—with not much about object-oriented design—while emphasizing project management. Sometimes, these courses are graduate courses; sometimes, they assume design and implementation are covered in more technical courses. In some cases, it might be assumed that packages are likely solutions rather than custom development, so defining requirements and managing the process are more important than design activities. The online chapters covering the role of the systems analyst, the traditional approach to structured analysis, and project management would be included.

A suggested outline for a course emphasizing object-oriented analysis, with in-depth coverage of project management, is:

Online Chapter A: The Role of the Systems Analyst
Chapter 1: From Beginning to End: An Overview of Systems Analysis and Design
Chapter 2: Investigating System Requirements
Chapter 3: Identifying User Stories and Use Cases
Chapter 4: Domain Modeling
Chapter 5: Use Case Modeling
Online Chapter B: The Traditional Approach to Requirements
Chapter 6: Foundations for Systems Design
Chapter 8: Designing the User Interface
Chapter 10: Approaches to System Development
Chapter 11: Project Planning and Project Management
Online Chapter C: Project Management Techniques
Chapter 14: Deploying the New System

Available Support

Systems Analysis and Design in a Changing World, Seventh Edition, includes teaching tools to support instructors in the classroom. The ancillary materials that accompany the textbook include an *Instructor's Manual*, solutions, test banks and test engine, PowerPoint presentations, and figure files. Please contact your Cengage Course Technology sales representative to request the Teaching Tools CD-ROM if you haven't already received it. Or go to the Web page for this book at login.cengage.com to download all these items.

The Instructor's Manual

The *Instructor's Manual* includes suggestions and strategies for using the text, including course outlines for instructors that emphasize the traditional structured approach or the object-oriented approach. The manual is also helpful for those teaching graduate courses on analysis and design.

Solutions

We provide instructors with answers to review questions and suggested solutions to chapter exercises and cases. Detailed traditional and UML object-oriented models are included for all exercises and cases that ask for modeling solutions.

ExamView

This objective-based test generator lets the instructor create paper, LAN, or Web-based tests from test banks designed specifically for this Course Technology text. Instructors can use the QuickTest Wizard to create tests in fewer than five minutes by taking advantage of Course Technology's question banks or instructors can create customized exams.

WebTUTOR™ Plug and Play!

Jump-start your course with customizable, text-specific content within your Course Management System!

- **Jump-start**—Instructors simply load a WebTutor cartridge or e-Pack into their Course Management System.
- **Content**—Students have access to text-specific content, media assets, quizzing, Web links, discussion topics, interactive games and exercises, and more.
- **Customizable**—Instructors can easily blend, add, edit, reorganize, or delete content.

Whether you want to Web-enable your class or put an entire course online, WebTutor delivers! Visit academic.cengage.com/webtutor to learn more.

Product Description

WebTutor and WebTutor Toolbox products are Course Cartridges and e-Packs that provide content natively on a Course Management System (WebCT, BlackBoard, Angel, D2L, and eCollege). The purpose of the product is to provide electronic solutions in an easy-to-use format with little up-front costs to instructors.

- For more information on how to bring WebTutor to your course, instructors should contact their Cengage Learning sales representative.

PowerPoint Presentations

Microsoft PowerPoint slides are included for each chapter. Instructors might use the slides in a variety of ways, such as teaching aids during classroom presentations or as printed handouts for classroom distribution. Instructors can add their own slides for additional topics they introduce to the class.

Figure Files

Figure files allow instructors to create their own presentations by using figures taken directly from this text.

Credits and Acknowledgments

We have been very gratified as authors to receive so many supportive and enthusiastic comments about ***Systems Analysis and Design in a Changing World***. Students and instructors in the United States and Canada have found our text to be the most up-to-date and flexible book available. The book has also been translated into many languages and is now used productively in Europe, Australia, New Zealand, India, China, and elsewhere. We truly thank everyone who has been involved in all the editions of our textbook, particularly Lori Bradshaw who managed the development of the seventh edition.

We also want to thank all the reviewers who worked so hard for us—beginning with an initial proposal and continuing throughout the completion of all seven editions of this text. We were lucky enough to have reviewers with broad perspectives, in-depth knowledge, and diverse preferences. We listened very carefully, and the text is much better as a result of their input. Reviewers for the various editions include:

Rob Anson, *Boise State University*
Marsha Baddeley, *Niagara College*
Teri Barnes, *DeVry Institute—Phoenix*
Robert Beatty, *University of Wisconsin—Milwaukee*

James Buck, *Gateway Technical College*
Anthony Cameron, *Fayetteville Technical Community College*
Genard Catalano, *Columbia College*
Paul H. Cheney, *University of Central Florida*
Kim Church, *Oklahoma State University*
Jung Choi, *Wright State University*
Jon D. Clark, *Colorado State University*
Mohammad Dadashzadeh, *Oakland University*
Lawrence E. Domine, *Milwaukee Area Technical College*
Gary Garrison, *Belmont University*
Cheryl Grimmett, *Wallace State Community College*
Jeff Hedrington, *University of Phoenix*
Janet Helwig, *Dominican University*
Susantha Herath, *St. Cloud State University*
Barbara Hewitt, *Texas A&M University*
Ellen D. Hoadley, *Loyola College in Maryland*
Jon Jasperson, *Texas A&M University*
Norman Jobes, *Conestoga College—Waterloo, Ontario*
Gerald Karush, *Southern New Hampshire University*
Robert Keim, *Arizona State University*
Michael Kelly, *Community College of Rhode Island*
Rajiv Kishore, *The State University of New York—Buffalo*
Rebecca Koop, *Wright State University*
Hsiang-Jui Kung, *Georgia Southern University*
James E. LaBarre, *University of Wisconsin—Eau Claire*
Ingyu Lee, *Troy University*
Terrence Linkletter, *Central Washington University*
Tsun-Yin Law, *Seneca College*
David Little, *High Point University*
George M. Marakas, *Indiana University*
Roger McHaney, *Kansas State University*
Cindi A. Nadelman, *New England College*
Bruce Neubauer, *Pittsburgh State University*
Michael Nicholas, *Davenport University—Grand Rapids*
Mary Prescott, *University of South Florida*
Alex Ramirez, *Carleton University*
Eliot Rich, *The State University of New York—Albany*
Robert Saldarini, *Bergen Community College*
Laurie Schatzberg, *University of New Mexico*
Deborah Stockbridge, *Quincy College*
Jean Smith, *Technical College of the Lowcountry*
Peter Tarasewich, *Northeastern University*
Craig VanLengen, *Northern Arizona University*
Bruce Vanstone, *Bond University*
Haibo Wang, *Texas A&M University*
Terence M. Waterman, *Golden Gate University*

Introduction to System Development

PART **ONE**

From Beginning to End: An Overview of Systems Analysis and Design

CHAPTER ONE

LEARNING OBJECTIVES

After reading this chapter, you should be able to:

- Describe the purpose of systems analysis and design when developing information systems
- Explain the purpose of the system development life cycle (SDLC) and identify its six core processes
- Explain how information system methodologies provide guidelines for completing the six core processes of the SDLC
- Describe the characteristics of Agile methodologies and iterative system development
- Based on the Ridgeline Mountain Outfitters Tradeshow System example:
  - Describe how the six core processes of the SDLC are used in each iteration
  - Identify key documents used in planning a project
  - Identify key diagrams used in systems analysis and systems design

CHAPTER OUTLINE

Software Development and Systems Analysis and Design

You have grown up in a world of ubiquitous computing, where computers are everywhere and are increasingly characterized by mobility, communication, and connectivity. You use smartphones, laptops, notepads, and wearable devices throughout the day. Some of you have already developed your own application software or have friends who have written applications for these devices. Some of you have taken programming classes; others have taught yourself how to write computer application programs. In one way or another, you are certainly interested in building computer applications and information systems.

Although you are most likely more familiar with your mobile devices, there is much more to building information systems than just that. Information systems exist to support all aspects of business organizations and have done so for centuries. The ancient Mesopotamians conducted business and had accounting information systems 3,000 years ago—using clay tablet technology. Electronic computers have been a part of these information systems only for the last 50 years. The technology changes, but information systems have a long history.

information system a set of interrelated components that collect, process, store, and provide as output the information needed to complete business tasks

An **information system** is a set of interrelated components that collect, process, store, and provide as output the information needed to complete business tasks. The information system always includes people who operate the system and carry out some of the work. In Mesopotamia, people did just about all of the work required. Now, of course, electronic computing devices do most of the work, although not all. If you are at the library typing in some search terms using the online catalog, you are part of the information system—the part that supplies the input and consumes the output. If you are using your bank's online information system, you are part of the information system—the part that selects which account to use to pay a specific bill.

computer application or **app** a computer software program that executes on a computing device to carry out a specific function or set of related functions

More recently, another term has been used to refer to an information system—a computer application. A **computer application** is a computer software program that executes on a computing device to carry out a specific function or set of related functions. Sometimes, *computer application* is shortened to **app** (such as an iPhone app or an Android app). Many people use the terms information system and computer application interchangeably, but remember that an information system includes people and their manual procedures and an application usually refers just to the software.

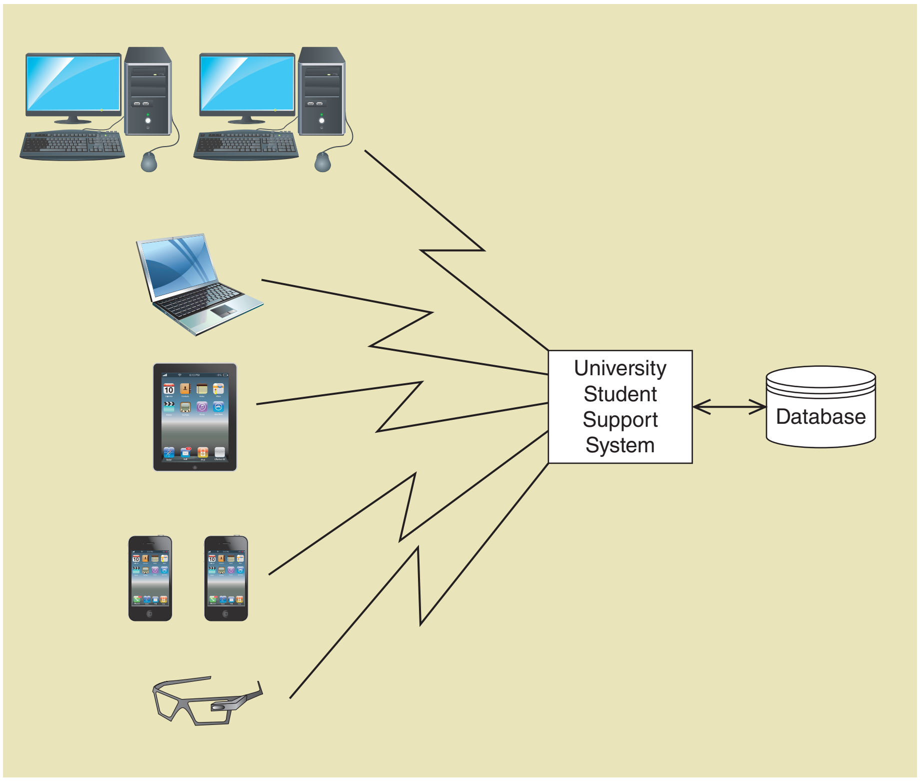
Consider the information system your university or college uses to support students. It is an elaborate system that likely integrates admissions, financial aid, course scheduling, and even individual course support. You probably access this information system through the network using a desktop workstation at home or in a computer lab, a wireless notebook computer, an iPad or tablet, an iPhone or an Android phone, and even a wearable device such as a smartwatch or Google Glass. There might be an app that connects to the system seamlessly from your device, or you might connect through a browser on your desktop, notebook, or other devices. **Figure 1-1** shows a variety of devices all connecting to the same University Student Support System.

Each information system (or app) was conceived and built to satisfy some need. When the information system is completed, it is used productively to satisfy that need. Our purpose here is to describe the process by which an information system is created from perceived need through actual use. As noted in this chapter's title, systems analysis and systems design are key components of this process.

systems analysis those system development activities that enable a person to understand and specify what the new system should accomplish

Systems analysis consists of those activities that enable a person to understand and specify what the new system should accomplish. The operative words here are *understanding* and *specifying*. Systems analysis is more than a brief

FIGURE 1-1 *A variety of devices all connected to the same information system*



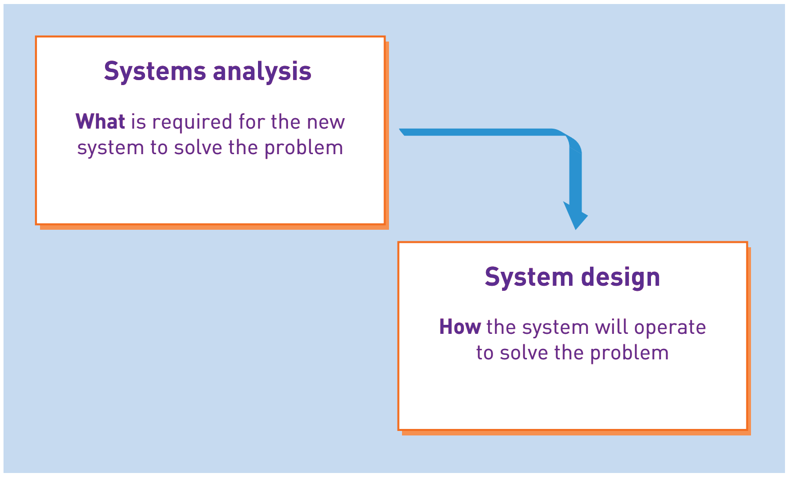
statement of the problem. For example, a customer management system must track customers, register products, monitor warranties, and track service levels, among many other functions—all of which have many details. Systems analysis describes in detail *what* a system must do to satisfy the need or solve the problem.

systems design those system development activities that enable a person to describe in detail how the resulting information system will actually be implemented

Systems design consists of those activities that enable a person to describe in detail how the information system will actually be implemented to provide the needed solution. In other words, systems design describes *how* the system will actually work. It specifies in detail all the components of the solution system and how they work together. See **Figure 1-2** to help distinguish between analysis and design.

Systems analysis and design plays an integral role in the development of information systems. To illustrate, consider an analogous situation: the art and

FIGURE 1-2 *Systems analysis versus systems design*


Systems analysis
What is required for the new system to solve the problem
System design
How the system will operate to solve the problem

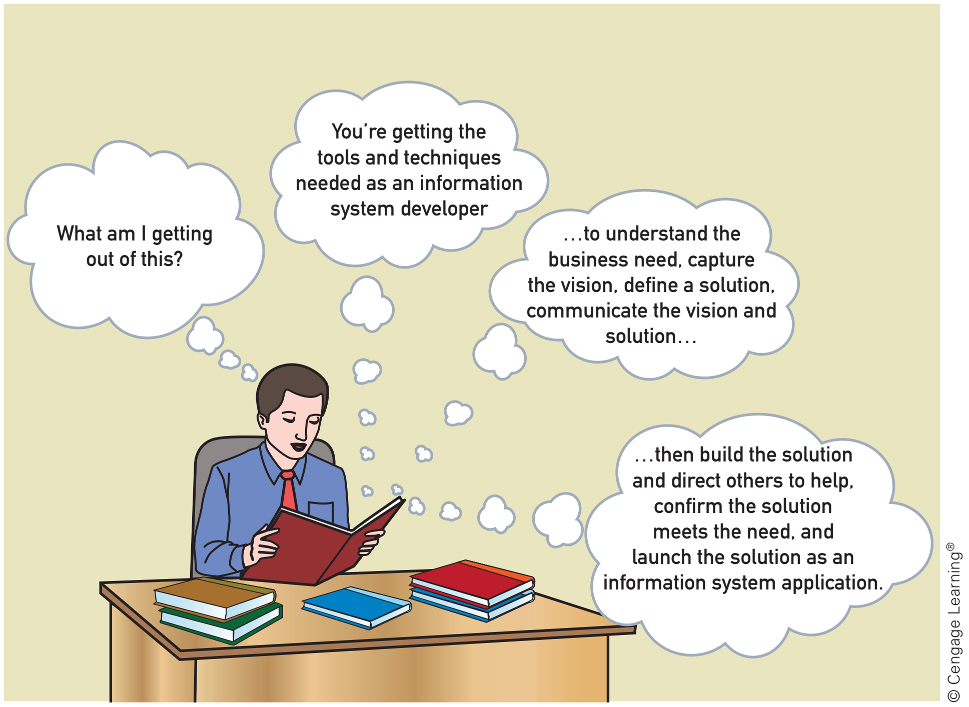
science of creating a new building. In this scenario, there is the owner of the land who has the vision, the builder who will construct the building, and the architect who serves as the bridge between the owner and the builder. The architect helps the owner develop the vision, but must also communicate the building's specifications to the builder. In doing so, the architect uses various tools to first capture the vision from the owner and to then provide the builder with instructions—including line drawings, blueprints, to-scale models, detail specifications, and even on-site inspection reports.

Just as a builder doesn't start construction without plans, programmers don't just sit down and start writing program code. They need someone (maybe themselves) to function like an architect—planning, capturing the vision, understanding details, specifying needs—before designing and writing the code that satisfies the vision. Usually, we call this person a *systems analyst*. In situations where you are the programmer as well as the analyst (often called a *programmer-analyst*), it might be possible to keep track of the details without many formal notes. However, in today's world, with system development teams often distributed worldwide, you might only be responsible for part of the programming, with the rest handled by team members in different locations. In a distributed team situation or with a complicated project, it is much more important to create formal requirements documents that capture each components' specifications.

In a nutshell, systems analysis and design provides the tools and techniques you need as an information system developer to complete the development process:

1. Understand the need (business need).
2. Capture the vision.
3. Define a solution.
4. Communicate the vision and the solution.
5. Build the solution or direct others in building the solution.
6. Confirm that the solution meets the need.
7. Launch the solution application.

FIGURE 1-3 *What analysis and design provides for the system developer*


What am I getting out of this?
You're getting the tools and techniques needed as an information system developer
...to understand the business need, capture the vision, define a solution, communicate the vision and solution...
...then build the solution and direct others to help, confirm the solution meets the need, and launch the solution as an information system application.
© Cengage Learning®

■ The System Development Life Cycle (SDLC)

project a planned undertaking that has a beginning and an end and produces some end result

Initial development of a new information system is usually done as a project. A **project** is a planned undertaking that has a beginning and an end and produces some end result. This means that the activities required to develop a new system are identified, planned, organized, and monitored. Some projects are very formal, whereas others are informal, usually depending on the project size.

system development life cycle (SDLC) a framework that identifies all the activities required to research, build, deploy, and often maintain an information system

To manage a project with analysis, design, and other development activities, you need a project management framework to guide and coordinate the work of the project team. The **system development life cycle (SDLC)** is a framework that identifies all the activities required to research, build, deploy, and often maintain an information system. Normally, the SDLC includes all activities needed for the planning, systems analysis, systems design, programming, testing, and user training stages of information systems development, as well as other project management activities that are required to successfully deploy the new information system.

There are many approaches to the SDLC, including variations specific to certain types of projects. However, every SDLC includes some core processes that are always required, though many different names are used. Here are the six core processes required in the development of any information system (see **Figure 1-4**):

- Identify the problem or need and obtain approval to proceed with the project.
- Plan and monitor the project—what to do, how to do it, and who does it.
- Discover and understand the details of the problem or the need—what is required?
- Design the system components that solve the problem or satisfy the need—how will it actually work?
- Build, test, and integrate system components—lots of programming and component integration.
- Complete system tests and then deploy the solution—the need now is satisfied.

system development process or **methodology** a set of comprehensive guidelines for carrying out all of the activities of each core process of the SDLC

As previously stated, most information systems you will develop are conceived and built to solve complex organizational problems, which are usually very complex, thus making it difficult to plan and manage a system development project. Fortunately, there are many ways to implement the six core processes of the SDLC to handle each project's complexity. An information systems development **methodology** is a set of comprehensive guidelines for carrying out all of the activities of each core process of the SDLC. An overall **system development process** is a more recent term for methodology. Each development

FIGURE 1-4 *Six core processes of the SDLC*

Core processes
Identify the problem and obtain approval.
Plan and monitor the project.
Discover and understand details.
Design system components.
Build, test, and integrate system components.
Complete system tests and deploy the solution.

© Cengage Learning®

methodology prescribes a way of carrying out the development project, and every organization develops its own system development methodology over time to suit its needs.

During the last 15 years, information system research efforts have resulted in many new information systems development methodologies/processes to improve the chance of project success. These are all based on what is called **Agile development**. The basic philosophy of Agile development is that neither team members nor the users completely understand the problems and complexities of a new system, so the project plan and the execution of the project must be responsive to unanticipated issues. The plan must be agile and flexible. It must have procedures in place to allow for, anticipate, and even embrace changes and new requirements that come up during the development process. The six core processes are still involved in Agile development, but they are carried out iteratively, as explained next.

Agile development an information system development process that emphasizes flexibility and rapid response to anticipate new and changing requirements during development

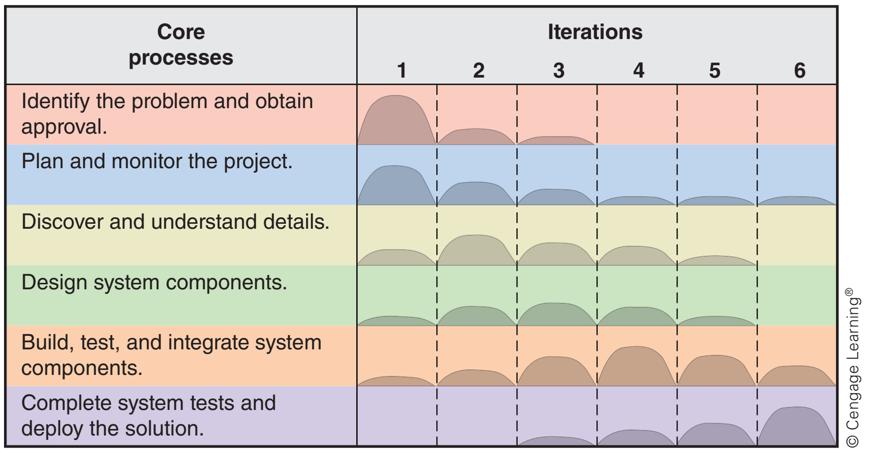
■ Iterative Development

Iterative development an approach to system development in which the system is "grown" piece by piece through multiple mini-projects called iterations

Iterative development is an approach to system development in which the system is "grown" in an almost organic fashion. Core components are developed first and then additional components are added. It is called *iterative* because the six core development processes are repeated for each component. In other words, there is one big project, which consists of a series of mini-projects, and the information system is grown piece by piece during these mini-projects. Iterative development makes Agile development possible, although Agile development includes additional techniques that help with project flexibility, too.

Figure 1-5 illustrates how an iterative project might be managed. Across the figure, you see six iterations as columns. Each iteration involves all six core processes, shown as rows in the table. At the end of each iteration, a working part of the system is completed and evaluated. An iteration lasts a fixed period of time, usually two to four weeks. The rounded mounds inside the graph represent the relative amount of effort for that core process during that iteration. For example, in Figure 1–5, Iteration 1 appears to primarily focus on identifying the problem and planning the project. Lesser amounts of discovery, design, and build/test may also be done. For this iteration, nothing is done with regard to deploying the system. In Iteration 2, there is less effort for identifying the problem and planning the project and more effort for discovery, design, and build/test. By Iteration 3, build/test gets the most effort, but all six core processes are still involved, including the beginnings of completing and deploying the system.

FIGURE 1-5 *The six core processes of the SDLC showing iterations*



There are several benefits to iterative development. For one, portions of the system can sometimes be deployed sooner. If there are core functions that provide basic support for users, these can be deployed in an early iteration. Second, by taking a small portion and developing it first, the most difficult problems can be identified and addressed early in the project. Many of today's systems are so large and complex that even with a formal process it is impossible to remember and understand everything. By focusing on only a small portion at a time, the requirements are fewer and easier to solve. Finally, developing a system in iterations makes the entire development process more flexible and able to address new requirements and issues that come up throughout the project.

A key element of iterative development is dividing system components into pieces that can be completed in two to four weeks. During one iteration, all the core development processes are involved, including programming and system-wide testing, so the result is a working part of the system, even though it may only have a portion of the functionality that is ultimately required. Developers choose components for each iteration based on priority, either the components most needed or riskiest to implement.

To better illustrate these concepts, we will walk through a complete example in the next sections concerning Ridgeline Mountain Outfitters (RMO). These sections use a fairly small information system to demonstrate all six core processes (as much as is feasible in a textbook, anyway). The example completes one iteration in detail, but the project actually requires multiple iterations. By going all the way through a very simple project, you will more easily understand the complex concepts provided in the rest of the text.

■ Introduction to Ridgeline Mountain Outfitters (RMO)

Ridgeline Mountain Outfitters (RMO) is a large retail company that specializes in clothing and related accessories for all types of outdoor and sporting activities. The mountain and western regions of the United States and Canada witnessed tremendous growth in recreation activities in recent years, including skiing, snowboarding, mountain biking, water skiing, jet skiing, river running, sailing, jogging, hiking, ATVing, cycling, camping, mountain climbing, and rappelling. With the increased interest in outdoor sports, the market for winter and summer sports clothing and accessories exploded, so RMO continually expanded its line of sportswear to respond to this market.

The company's growth charted an interesting history of mail-order, brick-and-mortar, and online sales. RMO got its start by selling to clothing stores in the Park City, Utah, area. In the late 1980s and early 1990s, it began selling directly to customers using catalogs with mail-in and telephone-order options. It opened its first store in 1994. After the Winter Olympics in Park City in 2002, business exploded and RMO quickly expanded to 10 retail outlets throughout the West and added Internet sales. Last year, retail store revenue was $67 million, telephone- and mail-order revenues were $10 million, and Internet sales were $200 million. Most sales continue to be in the West, although the market in several areas of the eastern United States and Canada is growing. By the Winter Olympics in Vancouver, British Columbia, in 2008, RMO's growth and profits resulted mainly from online sales and service, as with most specialty retailers; however, the brick-and-mortar and mail-order business remained important, too. After the Winter Olympics in Sochi in 2014, RMO negotiated with several Utah Olympic Medal Winners for endorsements. This provided additional interest throughout the West and instigated another period of rapid growth.

Figure 1-6 shows a sample of the catalog that RMO still mails out. Although mail-order and telephone sales are modest, receiving the catalog encourages