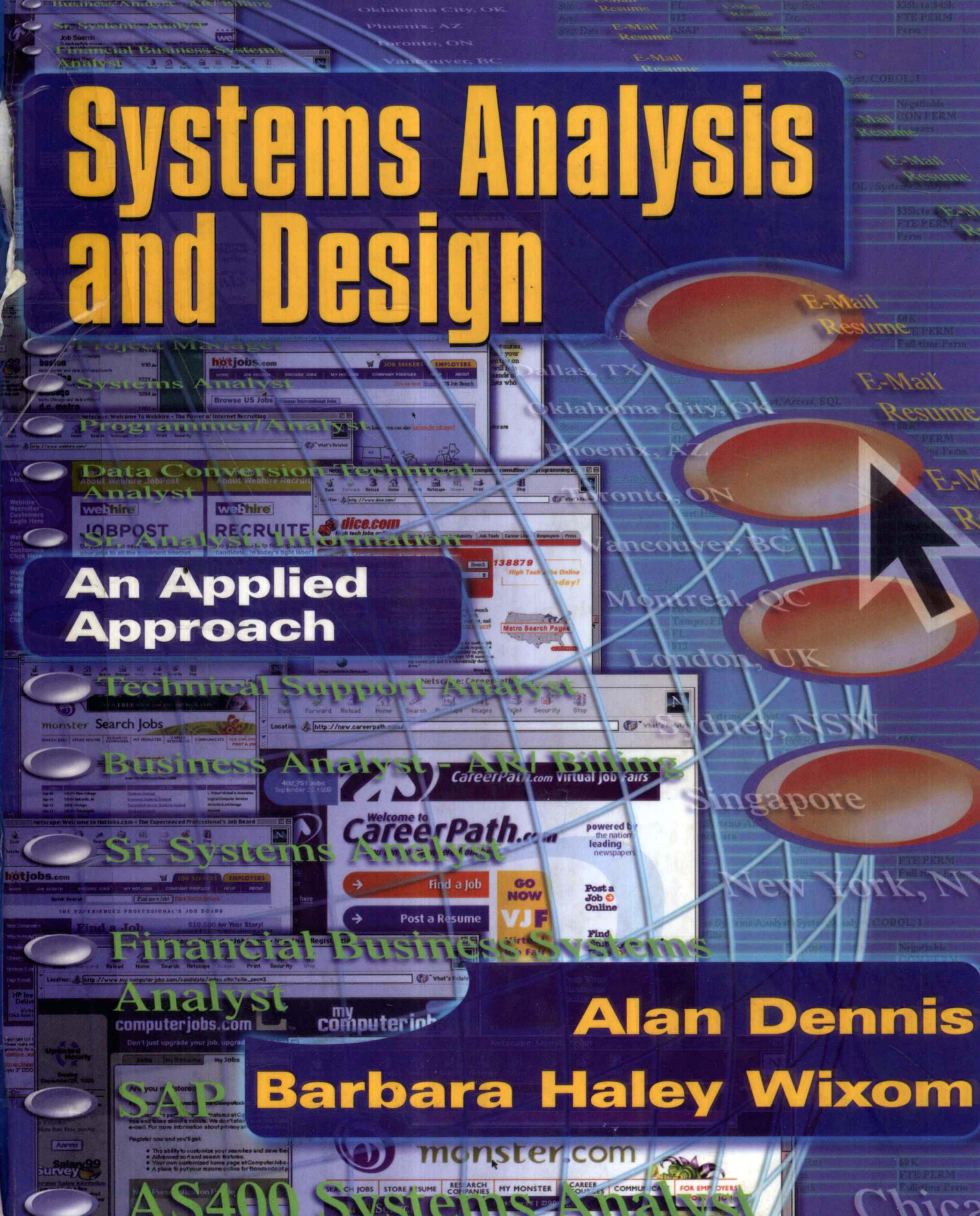
Systems Analysis and Design

An Applied Approach

Alan Dennis
Barbara Haley Wixom



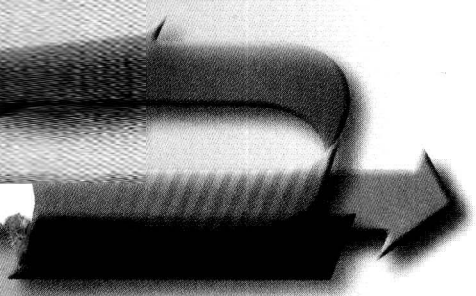
SYSTEMS ANALYSIS AND DESIGN

ALAN DENNIS

University of Georgia
Terry College of Business

BARBARA HALEY WIXOM

University of Virginia
McIntyre School of Business



John Wiley & Sons, Inc.

New York Chichester Weinheim Brisbane Singapore Toronto

http://www.wiley.com/college

ACQUISITIONS EDITOR:	Beth Golub
MARKETING MANAGER:	Peter Rytell
SENIOR PRODUCTION EDITOR:	Patricia McFadden
ART DIRECTION:	Dawn L. Stanley
TEXT DESIGNER:	Joan O'Connor
ILLUSTRATOR:	Norman Christensen
ILLUSTRATION EDITOR:	Anna Melhorn
COVER DESIGN:	Norman Christensen
PRODUCTION MANAGEMENT:	Hermitage Publishing Services

This book was set in 10.5/12 Times New Roman by Hermitage Publishing Services and printed and bound by Von Hoffman Press. The cover was printed by Phoenix Color, Inc.

Cover website screens provided courtesy of CareerPath.com, ComputerJobs.com, Dice.com, HotJobs.com, Monster.com, Netscape, and Webhire.com. Reproduced with permission.

This book is printed on acid-free paper. ♾

ISBN 0-471-24100-8

Printed in the United States of America

10 9 8 7 6 5 4

PREFACE

PURPOSE OF THIS BOOK

Systems Analysis and Design (SAD) is an exciting, active field in which analysts continually learn new techniques and approaches to develop systems more effectively and efficiently. However, there is a core set of skills that all analysts need to know—no matter what approach or methodology is used. All information systems projects move through the four phases of planning, analysis, design, and implementation; all projects require analysts to gather requirements, model the business needs, and create blueprints for how the system should be built; and all projects require an understanding of organizational behavior concepts like change management and team building.

This book captures the dynamic aspects of the field by keeping students focused on *doing* SAD while presenting the core set of skills that we feel every systems analyst needs to know today and in the future. This book builds on our professional experience as systems analysts and on our experience in teaching SAD in the classroom.

This book will be of particular interest to instructors who have students do a major project as part of their course. Each chapter describes one part of the process, provides clear explanations on how to do it, gives a detailed example, and then has exercises for the students to practice. In this way, students can leave the course with experience that will form a rich foundation for further work as a systems analyst.

OUTSTANDING FEATURES

A Focus on Doing SAD

The goal of this book is to enable students to *do* SAD—not just read about it, but understand the issues so they can actually analyze and design systems. The book introduces each major technique, explains *what* it is, explains *how* to do it, presents an *example*, and provides opportunities for students to *practice* before they do it for real in a project. After reading each chapter, the student will be able to perform that step in the system development life cycle (SDLC) process.

Rich Examples of Success and Failure

The book includes a running case about a fictitious company called CD Selections. Each chapter shows how the concepts are applied in situations at CD Selections. Unlike running cases in other books, we have tried to focus these examples on planning, managing, and executing the activities described in the chapter, rather than on detailed dialogue between fictious actors. In this way, the running case serves as a template that students can apply to their own work. Each chapter also includes numerous *Concepts in Action* boxes that describe how real companies succeeded—and failed—in performing the activities in the chapter. Many of these examples are drawn from our own experiences as systems analysts.

Incorporation of Object Oriented Concepts and Techniques

The field is moving towards object oriented concepts and techniques, both through UML, the new standard for object oriented analysis and design, as well as by gradually incorporating object oriented concepts into traditional techniques. We have taken two approaches to incorporating object oriented analysis and design into the book. First, we have integrated several object oriented concepts into our discussion of traditional techniques, although this may not be noticed by the students because few concepts are explicitly labeled as object oriented concepts. For example, we include the development of use cases as the first step in process modeling (Chapter 6), the use (and reuse) of standard interface templates (Chapter 10), and the development of use scenarios for interface design (Chapter 11). Second, and more obvious to students, we include a final chapter on UML that can be used as an introduction to object oriented analysts and design. This chapter can be used at the end of a course—while students are busy working on projects—or can be introduced after or instead of Chapter 7.

Real World Focus

The skills that students learn in a systems analysis and design course should mirror the work that they ultimately will do in real organizations. We have tried to make this book as "real" as possible by building extensively on our experience as professional systems analysts for organizations such as Arthur Andersen, IBM, the U.S. Department of Defense, and the Australian Army. We have also worked with a diverse industry advisory board of IS professionals and consultants in developing the book and have incorporated their stories, feedback, and advice throughout. Many students who use this book will eventually use the skills on the job in a business environment, and we believe they will have a competitive edge in understanding what successful practitioners feel is relevant in the real world.

Project Approach

We have presented the topics in this book in the SDLC order in which an analyst encounters them in a typical project. Although the presentation is necessarily linear (because students have to learn concepts in the way in which they build on each other), we emphasize the iterative, complex nature of SAD as the book

unfolds. The presentation of the material should align well with courses that encourage students to work on projects because it presents topics as students need to apply them.

Graphic Organization

The underlying metaphor for the book is doing SAD through a project. We have tried to emphasize this graphically throughout the book so that students better understand how the major elements in the SDLC are related to each other. First, at the start of every major phase of the system development life cycle, we have a graphic illustration that shows the major deliverables that will be developed and added to the "project binder" during that phase. Second, at the start of each chapter, we present a checklist of key tasks that will be performed to produce the deliverables associated with this chapter. These two graphic elements—the binder of deliverables tied to each phase, and the task checklist tied to the chapter—can help students better understand how the tasks, deliverables, and phases are related and flow from one to another. Finally, we have highlighted important practical aspects throughout the book by marking boxes and illustrations with a  "pushpin." These topics are particularly important in the practical day-to-day life of systems analysts and are the kind of topics that junior analysts should pull out of the book and post on the bulletin board in their office to help them avoid costly mistakes!

ORGANIZATION OF THIS BOOK

This book is organized by the phases of the Systems Development Life Cycle (SDLC). Each chapter has been written to teach students specific *tasks* that analysts need to accomplish over the course of a project, and the *deliverables* that will be produced from those tasks. As students complete the book, tasks will be "checked off" and deliverables will be completed and filed in a *Project Binder.* Along the way, students will be reminded of their progress using *roadmaps* that indicate where their current task fits into the larger context of SAD.

Chapter 1 introduces the SDLC and describes the roles and skills needed for a project team. Part 1 contains Chapters 2 and 3, which describe the first phase of the SDLC, the Planning Phase. Chapter 2 presents Project Initiation, with a focus on the System Request and Feasibility Analysis. In Chapter 3, students learn about Project Management, with emphasis on the Workplan, Staffing Plan, Project Charter, and Risk Assessment that are used to help manage and control the project.

Part Two presents techniques needed during the Analysis Phase. In Chapter 4, students create an Analysis Plan after learning a variety of analysis techniques to help with Business Automation, Business Improvement, and Business Process Reengineering. Chapter 5 focuses on information gathering techniques that are used to create Use Cases and Process Models (Chapter 6) and Data Models (Chapter 7).

The Design Phase is covered in Part 3 of the textbook. In Chapter 8, students learn how to convert existing process and data models into physical representations of the To-Be system. They create an Alternative Matrix that compares custom, packaged, and outsourcing alternatives. Chapter 9 focuses on architecture design, which includes the Infrastructure Design, Network Model, Hardware/Software Specification, and Security Plan. Chapters 10 and 11 present

interface design, and students learn how to create the Interface Structure, Interface Standards, User Interface Template, and User Interface Design. Finally, the data storage and program designs are illustrated in Chapters 12 and 13, which contain information regarding the Data Storage Design, Program Structure Chart, and Program Specification.

The Implementation Phase is presented in Chapters 14 and 15. Chapter 14 focuses on system construction, and students learn how to build and test the system. It includes information about the Test Plan and User Documentation. Installation is covered in Chapter 15, and students learn about the Conversion Plan, Change Management Plan, Support Plan, and the Project Assessment.

Chapter 16 provides a background in object orientation and explains several key object concepts supported by the standard set of object modeling techniques used by systems analysts and developers. Then, we explain how to draw four of the most effective models in UML: the use-case diagram, the sequence diagram, the class diagram, and the statechart diagram.

SUPPLEMENTS

Web Site (*www.wiley.com/college/dennis/sad*)

The Web site, developed by Daniel Mittleman of DePaul University, includes a variety of resources that will help in the instruction and learning processes:

- Short experiential exercises that instructors can use to help students experience and understand key topics in each chapter
- Word and RTF templates for the project deliverables that students can use as starting points for building their own project binders
- PowerPoint slides, prepared by Fred Niederman of St. Louis University, that instructors can tailor to their classroom needs and that students can use to guide their reading and studying activities
- Relevant Internet links are listed by chapter so that students can experience some concepts from the textbook using the Web environment

Instructors Manual

The Instructors manual, prepared by Roberta M. Roth of The University of Northern Iowa, provides resources to support the instructor both inside and outside of the classroom:

- Short experiential exercises that instructors can use to help students experience and understand key topics in each chapter
- Short stories have been provided by people working in both corporate and consulting environments for instructors to insert into lectures to make concepts more colorful and real
- Additional mini-cases for every chapter which allow students to practice using key concepts developed in the chapter.
- A test bank of different kinds of questions ranging from multiple choice and short answer to essay style
- Answers to end of chapter questions are provided

Cases in Systems Analysis and Design

A separate CD-based Case Book, edited by Jonathan Trower of Baylor University, provides a set of more than a dozen cases that can be used to supplement the book and provide exercises for students to practice with. The cases are a mixture of shorter cases designed to be used to support one phase within the SDLC, and longer cases than can be used to support an entire semester-long project. The cases are primarily drawn from the U.S. and Canada, but also include a number of international cases. We are always looking for new cases, so if you have a case that might be appropriate please contact us directly (or your local Wiley sales representative).

CASE Software

Two CASE (Computer-Aided Software Engineering) tools can be purchased with the text:

1. Oracle's Enterprise Development Suite, comprising of Oracle8i Personal Edition 8.1.5, Oracle Developer 6.0, and Oracle Designer 6.0. This software is available under a "Development License" for personal development purposes only, and has ***no*** time restrictions or limitations..
2. Visible Systems Corporation's Visible Analyst Student Edition.

Contact your local Wiley sales representative for details, including pricing and ordering information. Please note that you will need a specific ISBN (order number) for the different packaging options above.

Project Management Software

A 120-Day Trial Edition of Microsoft Project 98 can be purchased with the textbook. Contact your local Wiley sales representative for details.

ACKNOWLEDGMENTS

Our thanks to the many people who contributed to the preparation of this first edition. We are indebted to the staff at John Wiley & Sons for their support, including Beth Lang Golub, Information Systems Editor, Samantha Alducin, Editorial Program Assistant, Cindy Rhoads, Supplements Editor, Larry Meyer of Hermitage Publishing Services, Dawn L. Stanley, Art Director, and Anna Melhorn, Illustration Editor.

We would like to thank the following reviewers and focus group participants for their helpful and insightful comments:

Murugan Anandarajan	*Drexel University*
John Baron	*University of Illinois, Ph.D. Student*
Meral Binbasioglu	*Hofstra University*
Thomas Case	*Georgia Southern University*
Manoj Choudhary	*De Vry Institute of Technology, Scarborough, CA*

Mark Dishaw	*University of Wisconsin, Oshkosh*
Mark N. Frolick	*University of Memphis*
Rick Gibson	*American University*
Dale Goodhue	*University of Georgia*
Bill Hardgrave	*University of Arkansas*
Fred G. Harold	*Florida Atlantic University*
Jeffrey S. Harper	*Indiana State University*
Albert Harris	*Appalachian State University*
Monica C. Holmes	*Central Michigan University*
Peter C. Johnson	*California State University, Sacramento*
Ron Kelly	*Nova Scotia Community College, Burridge Campus*
Deepak Khazanchi	*Northern Kentucky University*
Chung S. Kim	*Southwest Missouri State University*
George M. Marakas	*Indiana University*
Vicki McKinney	*University of Wisconsin, Milwaukee*
Fred Niederman,	*University of Baltimore*
Richard O'Lander	*St. John's University-St. Vincent's College*
Tom Pettay	*De Vry Institute of Technology, Columbus, OH*
Alan M. Przyworski	*De Vry Institute of Technology, Decatur, GA*
Thomas C. Richards	*University of North Texas*
Cynthia Ruppel	*University of Toledo*
Nancy L. Russo	*Northern Illinois University*
Linda Salchenberger	*Loyola University Chicago*
Ulrike Schultze	*Southern Methodist University*
Tony Scime	*State University of New York, College at Brockport*
John B. Schwartz	*University of Maryland, Baltimore County*
Ted Strickland	*University of Louisville*
James Suleiman	*University of Colorado, Colorado Springs*
Ron Thompson	*University of Vermont*
Jonathan Trower	*Baylor University*
Duane P. Truex III	*Georgia State University*
William J. Vachula	*University of Pennsylvania*
David Vance	*Southern Illinois University*
Bruce White	*Dakota State University*
Rosann Webb Collins	*University of South Florida*

We would like to thank the following practitioners for helping us add a *real world* component to this project:

Matthew Anderson	*Andersen Consulting*
James Chapman	*Bell Atlantic*
Jeff Elgin	*Capital One*
Graig Galef	*American Management Systems*
Jane Griffin	*Arthur Andersen & Co.*
Mary Haffey	*Arthur Anderson & Co.*
Rixey Jones	*Home Depot*
Brice Marsh	*Computer Sciences Corporation*
Darrell Piatt	*First USA Bank*

David Price — *Connect Knowledge*
Troy Venis — *Check Free Corporation*
Gillen Young — *Ernst & Young*

Thanks also to our families and friends for their patience and support along the way, especially Christopher Wixom, Eileen Dennis, and Alec Dennis.

Alan Dennis
adennis@uga.edu

Barb Haley Wixom
bhaley@mindspring.com

CONTENTS

CHAPTER 1

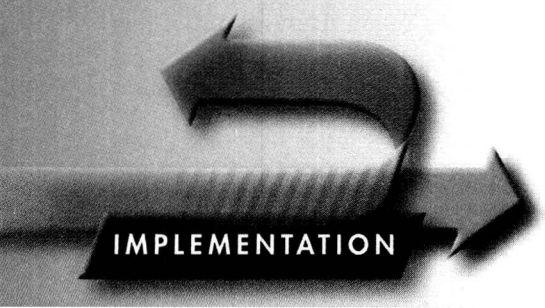
INTRODUCTION

This chapter introduces the systems development life cycle, the fundamental four-phase model (planning, analysis, design, and implementation) that is common to all information system development projects. It then examines five commonly used methodologies that differ in their focus and approach to each of these phases. The chapter closes with a discussion of the roles and skills within the project team.

OBJECTIVES

- Understand the fundamental systems development life cycle and its four phases.
- Understand five different types of methodologies and how to choose among them.
- Be familiar with the different roles on the project team.

CHAPTER OUTLINE



INTRODUCTION

The *systems development life cycle* (SDLC) is the process of understanding how an information system (IS) can support business needs, designing the system, building it, and delivering it to users. If you have taken a programming class or have programmed on your own, this probably sounds pretty simple. Unfortunately, it is not. A 1996 survey by the Standish Group found that 42% of all corporate IS projects were abandoned before completion. A similar study done in 1996 by the General Accounting Office found 53% of all U.S. government IS projects were abandoned. Unfortunately, many of the systems that aren't abandoned are delivered to the users significantly late, end up costing far more than planned, and have fewer features than originally planned.

Most of us would like to think that these problems only occur to other people or other organizations, but they happen in most companies. Even Microsoft has a history of failures and overdue projects (e.g., Windows 1.0, Windows 95).[1]

We would like to promote this book as a silver bullet that will keep you from IS failures, but a silver bullet that guarantees IS development success does not exist. Instead, this book will provide you with several fundamental concepts and many practical techniques that you can use to improve the probability of success.

The key person in the SDLC is the systems analyst, who analyzes the business situation, identifies opportunities for improvements, and designs an IS to implement them. Being a systems analyst is one of the most interesting, exciting, and challenging jobs around. As a systems analyst, you will work with a variety of people and learn how they conduct business. You will work with a team of systems analysts, programmers, and others on a common mission. You will feel the satisfaction of seeing systems that you designed and developed make a significant business impact and you will know that you contributed your unique skills to make that happen.

It is important to remember that the primary objective of the systems analyst is not to create a wonderful system. The primary goal is to create value for the organization, which for most companies means increasing profits (government agencies and not-for-profit organizations measure value differently). Many failed systems were abandoned because the analysts tried to build a wonderful system without clearly understanding how the system would fit with the organization's goals, current business processes, and other information systems to provide value. An investment in an information system is like any other investment, such as a new machine tool. The goal is not to acquire the tool, because the tool is simply a means to an end; the goal is to enable the organization to perform work better so it can earn greater profits or serve its constituents more effectively.

This book will introduce you to the fundamental skills you need to be a systems analyst. This is a pragmatic book that discusses best practices in systems development; it does not present a general survey of systems development that exposes you to everything about the topic. By definition, systems analysts *do things* and challenge the current way that organizations work. To get the most out of this

[1] For more information on the problem, see Capers Jones, *Patterns of Software System Failure and Success* London: International Thompson Computer Press, 1996; Capers Jones, *Assessment and Control of Software Risks,* Englewood Cliffs, NJ: Yourdon Press, 1994; Julia King, "IS reins in runaway projects," *Computerworld,* February 24, 1997.

CONCEPTS IN ACTION **1-A WHOLESALER SHELVES PROJECT**

Nash Finch Co., a $4 billion Minnesota-based supermarket and food wholesaler operator, shelved a major 100-person project to improve its retail information systems. Nash Finch had purchased SAP's R/3 retail system with the intention of improving operations, providing better data analysis, and preventing Y2K problems, but the R/3 system required so much additional programming to make it meet Nash Finch's needs that it became clear that the system would not be ready by January 2000. The company suspended the project after spending $50 million and was unable to estimate the additional cost needed to make its existing systems Y2K compliant.

Source: "Big retail SAP project put on ice," *Computerworld,* November 2, 1998

QUESTION

Why did this system fail? Why would a company start building a system that ultimately would not meet its needs?

book, you will need to actively apply the ideas and concepts in the examples, those in the "Your Turn" exercises that are presented throughout, and, ideally, from those you develop in your own systems development project. This book will guide you through all the steps for delivering a successful information system. Also, we will illustrate how one organization (which we call CD Selections) applies the steps in one project (developing a Web-based CD sales system). By the time you finish the book, you won't be an expert analyst, but you will be ready to start building systems for real.

In this chapter, we first introduce the basic SDLC that IS projects follow. This life cycle is common to all projects, although the focus and approach to each phase of the life cycle may differ. In the next section, we discuss two fundamentally different types of methodologies (structured design and rapid application development). Finally, we discuss one of the most challenging aspects of systems development—the depth and breadth of skills that are required. Today, most organizations use project teams that contain members with unique but complementary skills. This chapter closes with a discussion of the key roles played by members of the systems development team.

THE SYSTEMS DEVELOPMENT LIFE CYCLE

In many ways, building an information system is similar to building a house. First, the house (or the information system) starts with a basic idea. Second, this idea is transformed into a simple drawing that is shown to the customer and refined (often through several drawings, each improving on the other) until the customer agrees that the picture depicts what he or she wants. Third, a set of blueprints is designed that present much more detailed information about the house (e.g., the type of water faucets, where the telephone jacks will be placed). Finally, the house is built following the blueprints—and often with some changes and decisions made by the customer as the house is erected.

The SDLC has a similar set of four fundamental *phases*: planning, analysis, design, and implementation (Figure 1-1). Different projects may emphasize differ-

Phase	Chapter	Step	Sample Techniques	Deliverable
Planning (Why build the system?)	2	Identifying business value	System request	System request
		Analyze feasibility	Technical feasibility Economic feasibility Organizational feasibility	Feasibility study
	3	Develop work plan	Task identification Time estimation	Work plan
	3	Staff the project	Creating a staffing plan Creating a project charter	Staffing plan Project charter
	3	Control and direct project	Refine estimates Track tasks Coordinate project Manage scope Mitigate risk	Gantt chart[a] CASE tool Standards list Project binder(s) Risk assessment
Analysis (Who, what, when, where will the system be?)	4	Analysis	Problem analysis Benchmarking Reengineering	Analysis plan
	5	Information gathering	Interviews Questionnaires	Information
	6	Process modeling	Data flow Diagramming	Process model
	7	Data modeling	Entity relationship modeling	Data model
Design (How will the system work?)	8	Physical design	Custom development, package development outsourcing	Design plan
	9	Architecture design	Hardware design Network design	Architecture design Infrastructure design
	10, 11	Interface design	Interface structure chart User interface design	Interface design
	12	Database and file design	Selecting a data storage format Optimizing data storage	Data storage design
	13	Program design	Program structure chart Program specifications	Program design
Implementation (System delivery)	14	Construction	Programming Testing	Test plan Programs Documentation
	15	Installation	Direct conversion Parallel conversion Phased conversion	Conversion plan Training plan

CASE = computer-aided software engineering.
[a] The Gantt chart is named for its inventor, Henry Laurence Gantt.

FIGURE 1-1
System Development Life Cycle Phases

ent parts of the SDLC or approach the SDLC phases in different ways, but all projects have elements of these four phases. Each phase is itself composed of a series of *steps,* which rely on *techniques* that produce *deliverables* (specific documents and files that provide understanding about the project).

For example, when you apply for admission to a university, there are several phases that all students go through: information gathering, applying, and accepting. Each of these phases has steps; information gathering includes such steps as searching for schools, requesting information, and reading brochures. Students then use techniques (e.g., Internet searching) that can be applied to steps (e.g., requesting information) to create deliverables (e.g., evaluations of different aspects of universities).

Figure 1-1 suggests that the SDLC phases and steps proceed in a logical path from start to finish. In some projects, this is true, but in many projects, the project teams move through the steps consecutively, iteratively, or in other patterns. In this section, we describe the phases and steps and some of the techniques that are used to accomplish the steps at a very high level. We should emphasize that not all organizations follow the SDLC in exactly the way described below. As we shall shortly see, there are many variations on the overall SDLC.

For now, there are two important points to understand about the SDLC. First, you should get a general sense of the phases and steps that IS projects move through and some of the techniques that produce certain deliverables. Second, it is important to understand that the SDLC is a process of *gradual refinement.* The deliverables produced in the analysis phase provide a general idea of the shape of the new system. These deliverables are used as input to the design phase, which then refines them to produce a set of deliverables that describe in much more detailed terms exactly how the system will be built. These deliverables in turn are used in the implementation phase to produce the actual system. Each phase refines and elaborates on the work done previously.

Planning

The *planning phase* is the fundamental process of understanding *why* an information system should be built and determining how the project team will go about building it. The first step is called project initiation, during which the system's business value to the organization is identified—how will it lower costs or increase profits? Most ideas for new systems come from outside the IS area (from the marketing department, accounting department, etc.) in the form of a *system request.* A system request presents a brief summary of a business need and it explains how a system that supports the need will create business value. The IS department works together with the person or department that generated the request (called the *project sponsor*) to conduct a *feasibility analysis* that examines the idea's technical feasibility (i.e., can we build it?), the economic feasibility (i.e., will it provide business value?), and the organizational feasibility (i.e., if we build it, will it be used?).

The system request and feasibility analysis are presented to an information systems *approval committee* (sometimes called a steering committee), which decides whether the project should be undertaken. If the committee approves the project, then the second step of project initiation occurs—*project management.* During project management, the *project manager* creates a *work plan,* staffs the project, and puts techniques in place to help him or her control and direct the project through the entire SDLC. The deliverable for project management is a *project*