

# 用例建模

影印版

# USE CASE MODELING

[美] Kurt Bittner 著  
Ian Spence

Foreword by Ivar Jacobson

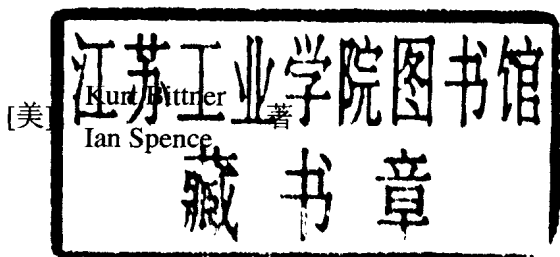


清华大学出版社

Addison-Wesley 对象技术丛书

# 用例建模

影印版



清华大学出版社

北京

## 内 容 简 介

本书提供了识别和描述用例的实践细节和用例的详细说明, 是对 Ivar Jacobson 著作的扩展和补充, 完整地介绍了确定用例以及用例发展情况的细节。

本书可作为软件学院及大学计算机等专业相关课程的教材, 也可以作为软件开发人员参考。

**EISBN: 0-201-70913-9**

**Use Case Modeling**

**Kurt Bittner, Ian Spence**

**Copyright © 2003 by Pearson Education, Inc.**

**Original English language edition published by Pearson Education, Inc.**

**All right reserved.**

For sale and distribution in the People's Republic of China exclusively (except Taiwan, Hong Kong SAR and Macau SAR). 仅限于中华人民共和国境内 (不包括中国香港、澳门特别行政区和中国台湾地区) 销售发行。

本书封面贴有 Pearson Education (培生教育出版集团) 激光防伪标签, 无标签者不得销售。

北京市版权局著作权合同登记号: 图字 01-2002-4652 号

### 图书在版编目 (CIP) 数据

用例建模 Use Case Modeling / (美) 比特纳, (美) 斯芬兹著. — 影印版.

— 北京: 清华大学出版社, 2003

ISBN 7-302-06558-6

I. 用... II. ①比... ②斯... III. ①软件工程—系统分析—英文②软件工程—系统设计—英文 IV. TP311

中国版本图书馆 CIP 数据核字 (2003) 第 027160 号

出 版 者: 清华大学出版社 (北京清华大学学研大厦, 邮编 100084)

<http://www.tup.com.cn>

<http://www.tup.tsinghua.edu.cn>

责任编辑: 尤晓东

印 刷 者: 清华大学印刷厂

发 行 者: 新华书店总店北京发行所

开 本: 880×1230 1/32 印张: 11.625

版 次: 2003 年 5 月第 1 版 2003 年 5 月第 1 次印刷

书 号: ISBN 7-302-06558-6 / TP·4915

印 数: 0001~3000

定 价: 28.00 元

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Addison-Wesley was aware of a trademark claim, the designations have been printed in initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers discounts on this book when ordered in quantity for bulk purchases and special sales. For more information, please contact:

U.S. Corporate and Government Sales  
(800) 382-3419  
corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact:

International Sales  
(317) 581-3793  
international@pearsontechgroup.com

Visit Addison-Wesley on the Web: [www.awprofessional.com](http://www.awprofessional.com)

*Library of Congress Cataloging-in-Publication Data*

Bittner, Kurt.

Use case modeling / Kurt Bittner, Ian Spence.  
p. cm.

Includes bibliographical references and index.

ISBN 0-201-70913-9 (pbk. : alk. paper)

1. System design. 2. Use cases (Systems engineering) I. Spence, Ian, 1961- II. Title.

QA76.9.S88 .B575 2003  
004.2'1--dc21

2002074790

Copyright © 2003 by Pearson Education, Inc.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher. Printed in the United States of America. Published simultaneously in Canada.

For information on obtaining permission for use of material from this work, please submit a written request to:

Pearson Education, Inc.  
Rights and Contracts Department  
75 Arlington Street, Suite 300  
Boston, MA 02116  
Fax: (617) 848-7047

ISBN 0-201-70913-9  
Text printed on recycled paper  
1 2 3 4 5 6 7 8 9 10—MA—0605040302  
First printing, August 2002

## The Addison-Wesley Object Technology Series

Grady Booch, Ivar Jacobson, and James Rumbaugh, Series Editors

For more information, check out the series Web site [<http://www.awprofessional.com/otseries/>].

Ahmed/Umrish, *Developing Enterprise Java Applications with J2EE™ and UML*

Arlow/Neustadt, *UML and the Unified Process: Practical Object-Oriented Analysis and Design*

Armour/Miller, *Advanced Use Case Modeling: Software Systems*  
Bellin/Simone, *The CRC Card Book*

Binder, *Testing Object-Oriented Systems: Models, Patterns, and Tools*  
Bittner/Spence, *Use Case Modeling*

Blakley, *CORBA Security: An Introduction to Safe Computing with Objects*

Booch, *Object Solutions: Managing the Object-Oriented Project*

Booch, *Object-Oriented Analysis and Design with Applications, Second Edition*

Booch/Bryan, *Software Engineering with ADA, Third Edition*

Booch/Rumbaugh/Jacobson, *The Unified Modeling Language User Guide*  
Box/Brown/Ewald/Sells, *Effective COM: 50 Ways to Improve Your COM and MTS-based Applications*

Carlson, *Modeling XML Applications with UML: Practical e-Business Applications*

Cockburn, *Surviving Object-Oriented Projects: A Manager's Guide*

Collins, *Designing Object-Oriented User Interfaces*

Conallen, *Building Web Applications with UML, Second Edition*

D'Souza/Wills, *Objects, Components, and Frameworks with UML: The Catalysis Approach*

Douglass, *Doing Hard Time: Developing Real-Time Systems with UML, Objects, Frameworks, and Patterns*

Douglass, *Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems*

Douglass, *Real-Time UML, Second Edition: Developing Efficient Objects for Embedded Systems*

Eeles/Houston/Kozaczynski, *Building J2EE™ Applications with the Rational Unified Process*

Fontoura/Preze/Rumpe, *The UML Profile for Framework Architectures*

Fowler, *Analysis Patterns: Reusable Object Models*

Fowler/Beck/Brant/Opdyke/Roberts, *Refactoring: Improving the Design of Existing Code*

Fowler/Scott, *UML, Distilled, Second Edition: A Brief Guide to the Standard Object Modeling Language*

Gomaa, *Designing Concurrent, Distributed, and Real-Time Applications with UML*

Graham, *Object-Oriented Methods, Third Edition: Principles and Practice*

Heinckens, *Building Scalable Database Applications: Object-Oriented Design, Architectures, and Implementations*

Hofmeister/Nord/Dilip, *Applied Software Architecture*

Jacobson/Booch/Rumbaugh, *The Unified Software Development Process*

Jordan, *C++ Object Databases: Programming with the ODMG Standard*

Kruchten, *The Rational Unified Process, An Introduction, Second Edition*

Lau, *The Art of Objects: Object-Oriented Design and Architecture*

Leffingwell/Widrig, *Managing Software Requirements: A Unified Approach*

Marshall, *Enterprise Modeling with UML: Designing Successful Software through Business Analysis*

McGregor/Sykes, *A Practical Guide to Testing Object-Oriented Software*

Mellor/Balcer, *Executable UML: A Foundation for Model-Driven Architecture*

Mowbray/Ruh, *Inside CORBA: Distributed Object Standards and Applications*

Naiburg/Maksimchuk, *UML for Database Design*

Oestereich, *Developing Software with UML: Object-Oriented Analysis and Design in Practice, Second Edition*

Page-Jones, *Fundamentals of Object-Oriented Design in UML*

Pohl, *Object-Oriented Programming Using C++, Second Edition*

Quatrani, *Visual Modeling with Rational Rose 2002 and UML*

Rector/Sells, *ATL Internals*

Reed, *Developing Applications with Visual Basic and UML*

Rosenberg/Scott, *Applying Use Case Driven Object Modeling with UML: An Annotated e-Commerce Example*

Rosenberg/Scott, *Use Case Driven Object Modeling with UML: A Practical Approach*

Royce, *Software Project Management: A Unified Framework*

Rumbaugh/Jacobson/Booch, *The Unified Modeling Language Reference Manual*

Schneider/Winters, *Applying Use Cases, Second Edition: A Practical Guide*

Shan/Earle, *Enterprise Computing with Objects: From Client/Server Environments to the Internet*

Smith/Williams, *Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software*

Stevens/Pooley, *Using UML, Updated Edition: Software Engineering with Objects and Components*

Unhelkar, *Process Quality Assurance for UML-Based Projects*  
van Harmelen, *Object Modeling and User Interface Design: Designing Interactive Systems*

Warmer/Kleppe, *The Object Constraint Language: Precise Modeling with UML*

White, *Software Configuration Management Strategies and Rational ClearCase®: A Practical Introduction*

## The Component Software Series

Clemens Szyperski, Series Editor

For more information, check out the series Web site

[<http://www.awprofessional.com/csseries/>].

Allen, *Realizing eBusiness with Components*

Atkinson/Bayer/Bunse/Karnsties/Laitenberger/Laqua/Muthig/Paech/Wust/Zettel, *Component-Based Product Line Engineering with UML*

Cheesman/Daniels, *UML Components: A Simple Process for Specifying Component-Based Software*

Szyperski, *Component Software, Second Edition: Beyond Object-Oriented Programming*

Whitehead, *Component-Based Development: Principles and Planning for Business Systems*

# Foreword

---

Use cases have come a long way since I first proposed them in 1986. Their value and power were clearly revealed by Object-Oriented programming. Use cases both contributed to and benefited from the development of the Object-Oriented paradigm. Today, knowledge of use cases is critical to one's understanding and application of UML and other modern software processes, such as the Rational Unified Process (RUP).

When used effectively, use cases have proven particularly valuable as part of the requirements activities of the software process. They have vastly improved communication between development teams and stakeholders and have made the determination of requirements far easier and more precise.

Use cases are unique in their ability to help teams understand the value the system must provide for its stakeholders. Because use cases describe how users use the system and what the system does for those users, they provide a unique way to build consensus about what the system must do. Building consensus is essential to a project's success: If the stakeholders cannot agree on the value the system must deliver, it is unlikely that the project can be successful.

Because use cases help create this understanding, they naturally provide an excellent principle around which to structure project activities. Use cases play an important role for analysts, who work with the requirements of the system; developers, who apply use cases to design and develop the system; testers, who verify that the system delivers the value demanded by the stakeholders; technical writers, who document how the system is used; and user-experience professionals, who help to make the system easy to use. All these project team members must understand use cases in order to develop better solutions.

To date, there has been something missing from the literature of use-case modeling: a description of the practical, day-to-day details of identifying and describing use cases. This book provides those details, defining the use-case model and fleshing out use-case descriptions. It's a perfect extension and complement to my earlier works, finishing the story of how the use cases are identified and how they evolve.

*Use Case Modeling* builds on the basic concepts by leveraging the practical experience that Kurt and Ian have gained through their many years of work in various industries—working with development teams either as consultants or as team members themselves. They have nicely distilled that experience into this very practical and insightful work. For people new to the field, this book provides an excellent tutorial. For use-case veterans, it provides an excellent reference that can be called upon on a daily basis.

This is the very best book on use cases ever written. Read it to understand use-case ideas and to apply those ideas with common sense based on the kind of system you are building and the maturity of your team members.

—Ivar Jacobson  
July 2002

## Preface

---

# Why Bother with Use Cases?

## WHAT ARE “USE CASES” ALL ABOUT?

In a world where it seems we already have too much to do, and too many things to think about, it seems the last thing we need is something new that we have to learn. As Eric Sevareid observed, the chief cause of problems is solutions.

But use cases do solve a problem with requirements: with strict declarative requirements it's hard to describe steps and sequences of events. To see why, let's consider a simple example:

### Example

Some requirements that must be satisfied by an automated teller system:

1. The system shall allow customers to withdraw cash from their accounts.
2. The system shall ensure that the customer's account is never overdrawn.
3. If the customer attempts to overdraw the account, the system will allow the account to be overdrawn, up to a specified amount, for a transaction fee.
4. If the customer is using an automated teller machine (ATM) that is owned by a financial institution other than the one to which the account belongs, an additional fee will be charged to the account.

Simple enough, you say. Or is it?

In what order should these things be done? Does it matter? If the ATM is not one that is owned by the customer's financial institution, should the ATM usage fee be charged before or after checking for overdraft? If the customer's



account balance is less than the ATM usage fee, charging the ATM usage fee before checking for overdraft will automatically result in an overdraft charge being applied, even if the customer decides to cancel the transaction. Is this the right behavior? With only declarative requirements, which is all that many projects have, it's impossible to say.

Use cases, stated simply, allow description of sequences of events that, taken together, lead to a system doing something useful. As simple as this sounds, this is important. When confronted only with a *pile of requirements*, it's often impossible to make sense of what the authors of the requirements really wanted the system to do. In the preceding example, use cases reduce the ambiguity of the requirements by specifying exactly when and under what conditions certain behavior occurs; as such, the sequence of the behaviors can be regarded as a requirement. Use cases are particularly well suited to capturing these kind of requirements. Although this may sound simple, the fact is that conventional requirement capture approaches, with their emphasis on declarative requirements and "shall" statements, completely fail to capture the dynamics of the system's behavior. Use cases are a simple yet powerful way to express the behavior of the system in way that all stakeholders can easily understand.

But, like anything, use cases come with their own problems, and as useful as they are, they can be misapplied. The result is something that is as bad, if not worse, than the original problem. Therein lies the central theme of this book—how to utilize use cases effectively without creating a greater problem than the one you started with.

## WHO SHOULD BE INTERESTED IN USE CASES?

The short answer to this question is "just about everyone," or at least everyone involved in some aspect of delivering a system that satisfies the needs of the customer. To be more specific about who should be interested in use cases, the following roles can benefit from the use-case technique of describing system behavior:

- **Customers**, who need to be sure that the system that is getting built is the one that they want
- **Managers**, who need to have an overall understanding of what the system will do in order to effectively plan and monitor the project
- **Analysts**, who need to describe and document what the system is going to do
- **Developers**, who need to understand what the system needs to do in order to develop it

- **Testers**, who need to know what the system is supposed to do so that they can verify that it does it
- **Technical writers**, who need to know what the system is supposed to do so that they can describe it
- **User-experience designers**, who need to understand the users' goals and how they will use the system to achieve these goals.
- And anyone else who wants to better understand what needs to be built *before* it is actually constructed

## HOW TO READ THIS BOOK

This book is fundamentally about creating use-case models and, more importantly, about writing detailed descriptions of use cases. To remain focused on this task, we have intentionally left out the parts of the project life cycle that use the use cases but are not directly involved in writing them. These areas include user-interface design, analysis, design, technical writing, testing, and project management. Other authors have covered a number of these areas adequately, and we felt that you, the reader, were best served if we focused narrowly on the use cases themselves. We hope you will agree.

This book is intended to be a ready reference for the practitioner, the person who is actually doing the work and grappling with the unique problems of working with use cases. It can certainly be read cover to cover, but the real intent behind the book is to provide you with something that can continue to add value after the first reading, providing you with a “mentor” at your fingertips. The topics presented in the book have arisen from working with countless project teams who grappled with the same issues facing you.

The book is divided into two parts. In Part I, Getting Started with Use-Case Modeling, we introduce the basics concepts of use-case modeling that you will need to understand in order to be effective using use cases. We conclude Part I with a description of an excellent way to get started with use cases: with a workshop.

- The first chapter, A Brief Introduction to Use-Case Modeling, provides practical background for people who are unfamiliar with use cases, or for people who have read other books and articles and still find themselves wrestling with the basic ideas. The purpose of the chapter is to provide a brief overview of the use-case approach without getting into a lot of formal details.
- The second chapter, Fundamentals of Use-Case Modeling, presents the foundations underlying the use-case modeling technique. The concepts presented here will provide the basis for the subsequent chapters in the book.

- The third chapter, *Establishing the Vision*, provides the essential tools for determining the business problem to be solved, for identifying the stakeholders in the solution, and for deciding what the system should do for those stakeholders to solve the business problem. This information is essential if we are to define the right solution when we develop our use-case model.
- The fourth chapter, *Finding Actors and Use Cases*, describes the process and subtleties of identifying the key elements of the use-case model. The purpose of this content is to help you through the sometimes-confusing task of getting started by providing a sound understanding of the basic concepts of actors and use cases.
- The fifth chapter, *Getting Started with a Use-Case Modeling Workshop*, describes the practicalities of getting started using use cases, including how to run a use-case workshop and how to deal with the practical details of starting to work with use cases.

In Part II, *Writing and Reviewing Use-Case Descriptions*, we explore the finer details of working with use cases, including the anatomy of a use case, how to write use-case descriptions (instead of the simple but incomplete descriptions presented in Part I), and what it means to work with use cases in practice. In these chapters, we explore in-depth how to write detailed use-case descriptions.

- The sixth chapter, *The Life Cycle of a Use Case*, describes the transitions that a use case undergoes as it evolves from concept to complete description. This chapter establishes context for the remaining chapters and places the content of Part I into a larger context.
- The seventh chapter, *The Structure and Contents of a Use Case*, describes the various constituent parts of a use case—the basic flow, preconditions, postconditions, and the alternate flows, as well as related topics.
- The eighth chapter, *Writing Use-Case Descriptions: An Overview*, describes the objectives and challenges related to writing detailed descriptions of use cases and presents strategies for successfully mastering this challenging task.
- The ninth chapter, *Writing Use-Case Descriptions: Revisited*, discusses the mechanics of how to go about writing use-case descriptions, how to handle details, and how to structure the descriptions for readability. This is done using an evolving example in which a variety of techniques are progressively and systematically applied to improve the quality of the use-case description.

- The tenth chapter, *Here There Be Dragons*, describes the problems that most teams encounter when using relationships between use cases (specifically the *include*, *extend*, and *generalization* relationships) and relationships between actors.
- The eleventh chapter, *Reviewing Use Cases*, describes how to organize and conduct reviews of the use-case model, including a summary of areas where particular focus is needed.

The final chapter, Chapter 12, *Wrapping Up*, touches on a number of topics related to how use cases are used in the larger context of the project, bringing our journey into the world of use cases to a close. In doing so, we provide the reader with a number of references to sources to consult for further information about how use cases are used in other disciplines.

## ACKNOWLEDGMENTS

We have had the pleasure over the years to work with many colleagues and customers who have helped shape the views that are presented here. A full enumeration of all of these people would be impossible, but we find ourselves especially indebted to a number of our colleagues for contributing to our views on use cases. We are in great debt to Ivar Jacobson, who originated the concepts of use-case modeling and initially defined their role in the modern software development process, for his support and encouragement on this project. We are also indebted to our colleague Dean Leffingwell for his work defining the role of use cases and traditional requirements-management approaches. We would also like to thank Bryon Baker, Chris Littlejohns, Anthony Kesterton, Gary Evans, Laurent Mondamert, Peter Eeles, Brian Kerr, and Susan August for their insightful suggestions at various points in the long evolution of this book. Special thanks go to Douglas Bush and Ida Audeh for their assistance in helping us to write clearly and concisely. We would also like to thank the many technical consultants at Rational whose experiences and questions have helped to shape this book. Finally, we would like to thank the customers with whom we and these consultants have worked, since their experiences and questions have ultimately made us realize that this book has been sorely needed. To all these people goes a great share of the credit for this book; any flaws or shortcomings are exclusively our own.

Kurt Bittner and Ian Spence  
April, 2002

# Contents

*Foreword*    *xiii*

**Preface**    **xv**

---

## **Why Bother with Use Cases?**

What Are “Use Cases” All About?    xv

Who Should Be Interested in Use Cases?    xvi

How to Read This Book    xvii

Acknowledgments    xix

**PART I   GETTING STARTED WITH USE-CASE MODELING**    **1**

---

**Chapter 1**    **3**

---

## **A Brief Introduction to Use-Case Modeling**

Actors and Use Cases    3

Use-Case Diagrams    4

**The Relationship Between Use Cases and Requirements**    **5**

*Types of Requirements*    5

*Functional and Nonfunctional Requirements*    8

*The Role of Use Cases*    9

*Use Cases Place Software Requirements in Context*    10

**To “Use Case” or Not to “Use Case”**    **12**

*When Are Use Cases Useful?*    13

*Use Cases Provide a Conceptual Model of the System*    13

*Use Cases Describe How the System Is Used and What It Does for Its Stakeholders*    14

*Does Everything the System Does Have to Be Described in a Use Case?*    14

**General Principles of Use-Case Modeling**    **15**

*Use Cases Do Not Exist in Isolation*    15

*Use Cases Are a Synthetic Rather Than an Analytic Technique*    17

*Rules of Thumb*    17

**Summary**    **18**

## Chapter 2

19

## Fundamentals of Use-Case Modeling

The Use-Case Model 20

The Basic Building Blocks of a Use-Case Model 21

*Actors* 21*Use Cases* 23*Connecting Actors and Use Cases* 25*Use-Case Diagrams* 28*Brief Descriptions* 29*Use-Case Descriptions* 30

Supporting Artifacts 39

*The Glossary and/or the Domain Model* 39*Supplementary Specifications* 43*Declarative and Special Requirements* 46

Summary 47

## Chapter 3

49

## Establishing the Vision

Introducing Stakeholders and Users 50

*What Are Stakeholders?* 51*The Role of Stakeholders and Stakeholder Representatives* 55*Users: A Very Important Class of Stakeholder* 58*Stakeholders and Use-Case Modeling* 61

Involving Stakeholders and Users in Your Project 63

*Step 1: Identify Stakeholder and User Types* 63*Step 2: Identify and Recruit the Stakeholder Representatives* 64*Step 3: Involve the Stakeholder Representatives in the Project* 67

Creating a Shared Vision 68

*Analyze the Problem* 69*Understand the Key Stakeholder and User Needs* 72*Describe the Features and Other High-Level Product Requirements* 74*Provide an Overview of the Product* 80

Bringing It All Together: The Vision Document 82

Do You Really Need to Do All of This? 85

Summary 85

## Chapter 4

87

**Finding Actors and Use Cases****Finding Actors 88**

- Start by Identifying the Primary Actors* 88
- Work from the Specific to the General* 89
- Don't Forget the Supporting Actors* 90
- Consider All Existing Requirements Information* 91
- Remember That Actors Are Not Always People* 91
- Focus on the System Boundary* 93
- Don't Preempt the Design* 94
- Don't Confuse the Actors with the Devices They Use* 95
- When you Can't Find the Actors, Start with the Use Cases* 96
- Focus First on the Familiar* 97
- Evolve the Set of Actors Alongside the Set of Use Cases* 97

**Documenting Actors 98**

- How to Name Actors* 98
- Don't Confuse Actors with Organizational Roles or Job Titles* 99
- Don't Overgeneralize* 100
- Give Every Actor a Brief Description* 101
- Characterize the Actors* 102
- Trace the Actors to the User Types, Stakeholders, and Stakeholder Roles* 103

**Finding Use Cases 104**

- Start by Identifying the Actor Goals* 104
- Consider the Information Needs of the System and Its Users* 105
- Don't Worry About Commonality (at least at first)* 106
- Don't Confuse Use Cases with "Functions"* 106
- Focus on Value* 108
- Derive the Use Cases from the System's Vision* 109
- Don't Forget the Supporting and Operational Use Cases* 110
- Evolve the Set of Use Cases Alongside the Set of Actors and the Supplementary Specification* 111

**Documenting Use Cases 111**

- Associate the Use Cases to Their Actors* 111
- Name the Use Cases* 112
- Give Every Use Case a Brief Description* 113
- Outline the Use Cases* 113
- Trace the Use Cases to Stakeholders and Stakeholder Roles* 116
- Trace the Use Cases to the Features and Constraints* 116

**Summary 117**

## Chapter 5

119

**Getting Started with a Use-Case Modeling Workshop****Reasons for Having a Workshop 119**

- To Transfer Expertise 120*
- To Build a Team 120*
- To Create Shared Understanding 120*
- To Tap into the Creative Power of a Group 121*

**Preparing for the Workshop 121**

- Train the Participants 121*
- Understand the Vision 122*
- Keep the Group Small and Involved 122*
- Vary the Composition of the Group 123*
- Select a Facilitator 124*
- Set Objectives for the Workshop 125*
- Schedule the Workshop and Organize the Facilities 126*

**Finding a Mentor 127**

- Find an Effective Communicator 127*
- Find a Skilled Motivator and Manager 127*
- Find a Mentor with Full Life-Cycle Experience 128*
- Don't Use the Mentor as a Crutch 129*

**Structuring the Workshop 129**

- Define the Ground Rules for the Workshop 129*
- Understand the Problem 131*
- Define the Boundary of the System 131*
- Identify Actors 133*
- Identify Use Cases 134*
- Consolidate the Model and Validate the Results 134*
- Wrap Up the Workshop and Plan the Next Steps 135*

**Supporting Activities 136**

- Capture Terminology in a Glossary 136*
- Capture Nonfunctional Requirements 138*
- Capture Issues, Risks, and Assumptions 138*

**Handling Common Problems 138**

- Avoid Functional Decomposition and Dataflow Modeling 139*
- Maintain Focus 139*
- Synthesize, Don't Analyze 140*
- Don't Describe What Happens Outside the System 141*
- Don't Just Draw Pictures 141*
- Don't Mix Business Use Cases and System Use Cases 142*

**Summary 143**



---

**PART II WRITING AND REVIEWING USE-CASE DESCRIPTIONS 145**


---

**Chapter 6 147**


---

**The Life Cycle of a Use Case**
**The Software Development Life Cycle 148**
**The Authoring Life Cycle 152**
*State 1: Discovered 153*
*State 2: Briefly Described 154*
*State 3: Bulleted Outline 154*
*State 4: Essential Outline 155*
*State 5: Detailed Description 157*
*State 6: Fully Described 159*
**Team Working 160**
*The Use-Case Modeling Process 161*
*Establish the Vision 163*
*Produce an Overview of the System 164*
*Reach Agreement on System Scope 164*
*Package the Use-Case Model 165*
*Address Areas of Instability and Author Stable Use Cases and*
*Supplementary Specifications 168*
*Consolidate and Review the Use-Case Model 169*
**Summary 169**


---

**Chapter 7 173**


---

**The Structure and Contents of a Use Case**
**Use Cases and System State 175**
*The System and External Events 175*
*The System State: More about Preconditions and Postconditions 177*
*How Use Cases Interact 180*
*The Side Effects of Using Preconditions 181*
**The Nature of the Flow of Events 182**
*The Structure of the Flow of Events 183*
*Managing Scope Using Alternative Flows 190*
*The Complexity of the Use-Case Model Versus the Complexity of the Design 192*
*Visualizing the Flow of Events 193*
*What Is a Scenario? 196*
*What Is a Use-Case Realization? 197*
**Summary 199**