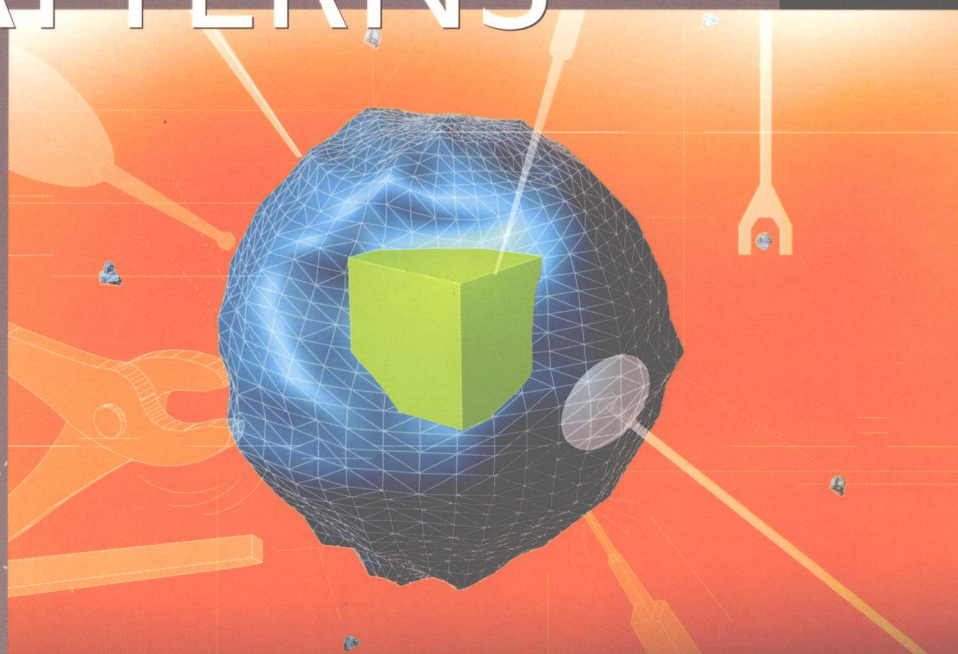


Microsoft

BEST PRACTICES

SOFTWARE REQUIREMENT PATTERNS



软件需求案例

China Edition
Published by Shanghai World Publishing Corporation

Stephen Withall

Foreword by Karl E. Wiegers
Author of Software Requirements

世界图书出版公司

Software Requirement Patterns

江苏工业学院图书馆
藏书章

Stephen Withall

图书在版编目(CIP)数据

软件需求案例:英文/(美)韦瑟尔著. -上海:上海世界图书出版公司,2008.10

ISBN 978-7-5062-9292-4

I. 软… II. 韦… III. 软件开发-英文 IV. TP311.52

中国版本图书馆 CIP 数据核字(2008)第 159885 号

© 2007 by Microsoft Corporation. All rights reserved.

Original English language edition © 2007 by Stephen Withall. All rights reserved. Published by arrangement with the original publisher, Microsoft Corporation, Redmond, Washington, U. S. A.

软件需求案例

[美] 史蒂文·韦瑟尔 著

上海世界图书出版公司 出版发行

上海市尚文路 185 号 B 楼

邮政编码 200010

(公司电话:021-63783016 转发行部)

上海竞成印务有限公司印刷

如发现印装质量问题,请与印刷厂联系

(质检科电话:021-56422678)

各地新华书店经销

开本:787×960 1/16 印张:24 字数:750 000

2008 年 10 月第 1 版 2008 年 10 月第 1 次印刷

ISBN 978-7-5062-9292-4/T·179

图字:09-2008-627 号

定价:188.00 元

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

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

Foreword

Requirements development is hard! Requirements analysts often are not adequately trained or experienced, so they do the best they can without necessarily knowing how to write high-quality requirements. Analysts struggle with questions such as “Where do I start?,” “How do I know when I’m done?,” “How detailed should my requirements be?,” “Have I missed any requirements?,” and “Have I overlooked any critical information in the requirements I’ve written?” Unfortunately, there’s no formulaic approach to the communication-intensive challenge of understanding and specifying requirements.

Stephen Withall’s *Software Requirement Patterns* can help any analyst write better requirements. These patterns provide a way to embody comprehensive and structured knowledge about different types of requirements. Requirements development is a journey of exploration, not just a simple collection or transcription process. The patterns Steve presents can help analysts ask the right questions to properly understand and specify requirements of many types in an appropriate level of detail. From the perspective of “know your audience,” the patterns include guidance to assist the developers and testers who must take the requirements to the next development stages. People learn from examples, and they work more efficiently with the help of templates rather than blank pages. To this end, Steve’s requirement patterns provide both templates and examples.

These requirements patterns are applicable to a wide variety of projects and products. You can apply the concepts in the book to develop new requirement patterns specific to your own industry, application domain, or product line. Too many projects begin specifying requirements from scratch, but the requirement patterns let organizations effectively reuse requirements knowledge captured on previous projects.

This book communicates a wealth of wisdom and insight for writing stellar requirements. Through the patterns, Steve points out the value of using a consistent style when writing requirements, which can enhance every analyst’s capabilities. Even if you don’t apply the patterns rigorously, the book contains hundreds of practical tips for specifying better requirements. Use the book as a reference: read the relevant patterns, try them, and absorb the ideas and advice Steve presents. Internalizing those patterns that fit your situation will make them a routine aspect of how you explore, analyze, document, and use software requirements.

Requirement patterns just might represent the next generation of software requirements thinking. Stephen Withall’s *Software Requirement Patterns* will likely remain the definitive treatise on requirement patterns for years to come.

Karl Wieggers
April 2007

Preface

The Purpose of This Book

There is nothing new under the sun. It has all been done before.

—Sherlock Holmes: A Study in Scarlet,
Arthur Conan Doyle

The purpose of this book is to help you decide and define what a new software system needs to do and to suggest what extra features to add to make it a *good* system—or even an *excellent* one. It saves you effort and enables you to be more precise, by providing detailed guidance on how to specify individual requirements. Requirement patterns are encapsulated expertise, conveniently prepackaged for reuse. The book contains 37 requirement patterns, each of which describes an approach to tackling a particular type of situation that crops up repeatedly in all kinds of systems, but focusing on commercial business software. Only a fraction of any system is specific to its business area; the bulk occurs over and over again no matter what your system is for. These patterns cover more than half of all requirements in some systems—a lot more if we add the extra requirements the patterns suggest.

If you're wary of the word “requirement” here, don't be; it doesn't mean you have to be embroiled in paperwork. This book is suitable for use by business analysts using a traditional analysis approach and by software architects and engineers who use agile methods. You can use requirement patterns to help you identify and define what a system needs to do even if you don't write formal requirements as a result.

The requirements for a software system specify the problem it needs to solve—its purpose and goals. If they're omitted or done badly—which is, unfortunately, all too frequently the case—a system is unlikely to be a perfect fit, no matter how well it's implemented. A disturbing proportion of computer systems are judged to be inadequate; many are not even delivered; more are late or over budget. Studies consistently show that the single biggest cause is poorly defined requirements: not properly nailing down a system's purpose and what it must do. Even a modest contribution to improving requirements offers the prospect of saving businesses part of a huge sum of wasted investment.

To build good systems more often, improvements are needed all along the chain. Serious efforts have been (and continue to be) made in nearly all of them. But most fundamental of all is what the requirements themselves actually *say* (and, just as importantly, *fail to say*). That's been neglected, but it's what this book concentrates on. If I want to define a requirement of a specific type, what do I need to write? How do I go about it? What extra requirements should I consider writing? What else should I worry about? This book identifies many areas (big and small) for which requirements are frequently inadequate, imprecise, or missed altogether—and suggests what you can do about it. The patterns themselves aim to be down-to-earth and practical—primarily distilled from personal experience—things I wish I'd known all along.

This is primarily a *reference work*, to be pulled out whenever you want help in a particular situation—to explain what a requirement needs to convey, raise questions to ask, point out potential pitfalls, suggest extra requirements, provide example requirements, and generally provide practical advice. You can start using the requirement patterns without having read the book through.

This book contains lots of example requirements—over 400—many of which are suitable for applying unchanged to any system and others that are a useful starting point for a requirement to suit the reader's needs. These examples are the heart of the book. It was from the study of real-life requirements that the requirement patterns in this book were identified. Omissions, ambiguities, and other weaknesses in these real requirements fed much of what the requirement patterns have to say.

This book also provides guidance on how to write other kinds of information that belong in a requirements specification—such as system scope, assumptions, a glossary, document history, and references—and how to structure a requirements specification.

What This Book is Not

This is not a book about the process of specifying requirements or about analysis techniques or requirements management. There are other good books that explain all those things, and this book can be used as a reference alongside them. This book can, however, be used perfectly well by itself; it includes a “crash course in specifying requirements” for readers with no previous experience.

This book doesn't advocate any particular methodology, approach, or specification software tool. It provides relevant advice no matter which way you choose to work. It isn't prescriptive: it doesn't say, “You must do it this way.” It steers clear of jargon and avoids introducing its own terminology as far as possible.

You won't agree with everything in this book, and you won't need to act on all the suggestions made by any requirement pattern. But if the time it saves you, when writing requirements or later, is worth more than the purchase price, it has earned its keep. I hope that these patterns prove useful one way or another, by containing enough useful and thought-provoking material to lead you to produce better systems.

Who Will Benefit from Using This Book

The primary audience of this book is **anyone involved in deciding what a new software system needs to do**. This is the business of specifying the requirements for a software system, even if you don't like the word “requirement” or you don't end up writing a full requirements specification. For convenience, we refer to any person who specifies requirements as an **analyst**; they could be a business analyst, a systems analyst, a systems architect, or a software engineer; they could be a business-oriented or technical person. They might have previous experience with specifying requirements, or they might not. They can be divided into those who use traditional analysis processes and those who use more agile methods:

- a. **Business analysts**, or anyone fulfilling that role. This book makes no assumptions about how much the reader knows: it's suitable for both junior and experienced business analysts as well as for business executives and software engineers who have never specified requirements before. Requirement patterns can be put into practice quickly.
- b. **Software architects and engineers** on any system for which requirements have *not* been written—because the gap must be filled, and it will be one way or another. This book's advice

is equally relevant no matter who decides what a system needs to do. Its advice is of just as much value to any organization that does not have dedicated analysts, and particularly those that take an *agile* approach to development. Agile methods place little (if any) emphasis on writing requirements specifications, but still the functionality of the system must be identified—and the requirement patterns in the book can help just as well here as when using a traditional approach. In *extreme programming*, in particular, requirement patterns can help you write user stories, interpret user stories, and formulate “rules” for good practices for developers to follow. Software architects and engineers who are familiar with *design patterns* should be particularly comfortable using *requirement patterns*.

Secondary audiences are:

- c. Anyone asked to **review** a requirements specification, which covers a wide range of technical, managerial, and sales people as well as a new system’s user community. This book can help reviewers judge a specification’s quality and completeness, and discover omissions.
- d. **Software developers** who must implement requirements. Each requirement pattern contains a “Considerations for Development” section to assist developers.
- e. **Software testers** who must test how well the delivered system satisfies its requirements. Each requirement pattern contains a “Considerations for Testing” section for testers with suggestions on how to test requirements of that type.
- f. **Project managers** who manage a system’s requirements, changes to them, and a project to implement them.

Job titles of people who will find this book valuable include business analyst, systems analyst, business systems analyst, software architect, systems architect, software engineer, testing engineer, product manager, project manager, project office manager, and chief technical officer.

Benefits the Reader Will Gain

You, dear reader, will be able to improve your skills and productivity in the following ways from reading this book (and from using it as a reference):

- 1. You will be able to define better requirements—with more detail, precision, and clarity, and with less ambiguity.
- 2. You will be able to write requirements more quickly and with less effort, by taking advantage of the effort already put into the book (reuse!).
- 3. You will recognize extra topics that requirements should specify, further improving their results and making them more complete.
- 4. You will be better able to organize a requirements specification and to write general sections (such as the glossary).

As a result you, your colleagues, and the organization you work for will see further benefits:

- 5. It is easier to estimate the effort needed to build a specified system.
- 6. Development and testing teams will find it easier to understand the requirements.
- 7. The resulting system will better reflect the organization’s needs, potentially yielding considerable extra return on the investment in it. What price can you put on avoiding a big mistake?

8. Fundamental mistakes, misunderstandings, and omissions will be spotted earlier—with potentially huge savings, given that fixing a defect during the design phase costs roughly ten times more than during requirements, and during development ten times more again.

Skills and Experience Needed by the Reader

This book can be used with no previous experience of specifying requirements. Chapter 1 is a “crash course” containing the bare minimum that a novice reader needs to get started. A good general book on requirements engineering (such as those cited at the beginning of Chapter 1) is a better introduction, and readers who have read them or who are already experienced business analysts are likely to get more from this book. Software engineers using agile methods can use the book in isolation. Anyone responsible for *reviewing* a requirements specification needs no previous knowledge or skills in order to use this book to help them.

This book is accessible to a nontechnical reader. It focuses on writing textual requirements in natural language that can be read by anyone. It is free of arcane diagram formats, deep theory, and jargon. You can read it without knowing UML (Unified Modeling Language) or any other formal technique.

The Structure of This Book

This book is divided into two parts:

- **Part I: Setting the Scene** These four explanatory chapters open with Chapter 1, “Crash Course In Specifying Requirements,” written for someone who is inexperienced at specifying requirements—but everyone should read it, because it states a few principles that are important to the rest of the book. Chapter 2, “The Contents of a Requirements Specification,” describes the types of material, in addition to requirements, that belong in a requirements specification. The versions of Chapters 1 and 2 printed in the book are merely synopses of much longer, “full” versions that can be downloaded from the associated Web page (as described in the “Supporting Resources” section that follows). Chapter 3, “Requirement Pattern Concepts,” explains what requirement patterns are all about: the basics, what each pattern contains, how they’re organized (into domains), and related concepts. Chapter 4, “Using and Producing Requirement Patterns,” explains how to use requirement patterns and to write your own.
- **Part II: Requirement Pattern Catalog** These are sets of patterns for types of requirements that occur repeatedly, to be used as a reference. It opens with a snapshot of the requirement patterns in this book and then has eight chapters (5 through 12) containing the requirement patterns themselves.

Bringing up the rear are a glossary of terms and acronyms used and encountered in the book, plus a list of references.

I advise that you read through Part I to understand what’s going on. If Chapters 1 and 2 in the book don’t tell you enough, refer to the Web page for the full versions. You don’t need to devour Part II systematically: familiarize yourself with the patterns that it contains (unless you’re an analyst keen for advancement!), and refer to it whenever you encounter a situation in which one of the patterns will help.

Supporting Resources

You can download the following documents from the book's companion Web page at <http://www.microsoft.com/mspress/companion/9780735623989>:

1. The full version of Chapter 1, "Crash Course in Specifying Requirements."
2. The full version of Chapter 2, "The Contents of a Requirements Specification."
3. "Example Requirements," a complete set of all the examples in the book, plus the requirement templates for all the requirement patterns, to make it easy to copy and paste an individual template or example to use as a starting point when writing a requirement of your own. This document also includes a requirement pattern template, to use if you want to write your own patterns.
4. A "Ready Reference" suitable for printing, containing a diagram of all the requirement patterns plus a list of all the requirement patterns and the "applicability" of each one (to make it easy to figure out which pattern to use when).

The first two are available both as Adobe PDF (Portable Document Format) documents and Microsoft XML Paper Specification (XPS) documents. The last two are available as Microsoft Word documents. To download these documents, you will need about 6 MB of disk space. For system requirements for viewing these files, see the companion Web page.

Acknowledgments

I greatly appreciate the diligent and generous contributions of a number of people, without whose assistance this book would have been much the poorer—or wouldn't even have been completed at all. First, special thanks to Trish Reader for encouragement all the way through, sound business analysis advice, and feedback on various drafts.

I am deeply indebted to all my reviewers, especially those who heroically read and commented cover to cover: to Roxanne Miller, for her deep understanding of what all business analysts will look for in this book, and for keeping me (relatively) honest on analysis techniques; and to Lydia Ash, for her expertise on testing but also countless invaluable suggestions on almost everything. I appreciate the feedback and suggestions of Robert Posener for scrutinizing the text with the all-seeing eye of the consummate project manager; Craig Malone on development methodologies (especially agile matters); Marc Munro for his database expertise on the information and data entity patterns; security guru Eric Fitzgerald on access control; and accessibility experts Annuska Perkins, Norm Hodne, Ramkumar Subramanian, and Laura Ruby. Finally, thanks to Shanno Sanders for perceptive insights on the overall direction of the book. Sometimes I have rashly persisted in disregarding their advice, for which I assume full responsibility—as I do for all errors that remain.

I am grateful to Karl Wiegiers for contributing such a generous Foreword, and for the early encouragement that was the nudge I needed to write this book.

I'd like to thank everyone at Microsoft Press, especially acquisitions editor Ben Ryan for his faith in the concept, and editors Devon Musgrave and Maria Gargiulo for their never-failing patience, their good-natured reactions to even the quirkiest of my ideas, and their painstaking copy editing.

Finally, this book could never have been written at all if not for the innumerable people who have contributed to my professional experience over the years. The most valuable have been those at the two extremes of the spectrum: the excellent, from whom I've learned so much about how to specify and develop good systems; and the inept, whose creativity in finding ways to do things wrong is an education in itself. Thanks to you all.

Microsoft Press Support

Every effort has been made to ensure the accuracy of this book. Microsoft Press provides corrections for books through the World Wide Web at the following address:

<http://www.microsoft.com/mspress/support/>

To connect directly to the Microsoft Press Knowledge Base and enter a query regarding a question or issue that you may have, go to:

<http://www.microsoft.com/mspress/support/search.asp>

If you have comments, questions, or ideas regarding this book, please send them to Microsoft Press using either of the following methods:

Postal Mail:

Microsoft Press

Attn: Software Requirement Patterns Editor

One Microsoft Way

Redmond, WA 98052-6399

E-Mail:

mspinput@microsoft.com

Contents at a Glance

Part I **Setting the Scene**

- 1 Synopsis of "Crash Course in Specifying Requirements"3
- 2 Synopsis of "The Contents of a Requirements Specification" 11
- 3 Requirement Pattern Concepts..... 19
- 4 Using and Producing Requirement Patterns 39

Part II **Requirement Pattern Catalog**

- 5 Fundamental Requirement Patterns 51
- 6 Information Requirement Patterns 85
- 7 Data Entity Requirement Patterns 119
- 8 User Function Requirement Patterns..... 155
- 9 Performance Requirement Patterns..... 191
- 10 Flexibility Requirement Patterns..... 239
- 11 Access Control Requirement Patterns 281
- 12 Commercial Requirement Patterns 325
- Glossary 341
- References 349

Table of Contents

Foreword.....	ix
Preface.....	xi

Part I **Setting the Scene**

1 Synopsis of "Crash Course in Specifying Requirements"	3
1.1 What Are Requirements?	4
1.2 Where Do Requirements Fit in the Grand Scheme?	5
1.3 A Few General Principles	6
1.4 A Traditional Requirements Process	7
1.5 Agile Requirements Processes	8
An Extreme Requirements Process	9
An Incremental Requirements Process	10
2 Synopsis of "The Contents of a Requirements Specification"	11
2.1 Introduction Section	12
System Purpose	12
Document Purpose	12
Requirement Format	13
Glossary	14
References	14
Document History	15
2.2 Context Section	15
Scope	15
Major Assumptions	16
Major Exclusions	16
Key Business Entities	16
Infrastructures	17

 **What do you think of this book? We want to hear from you!**

Microsoft is interested in hearing your feedback so we can continually improve our books and learning resources for you. To participate in a brief online survey, please visit:

www.microsoft.com/learning/booksurvey/

2.3 Functional Area Sections	17
2.4 Major Nonfunctional Capabilities Section	18
3 Requirement Pattern Concepts	19
3.1 Introduction to Requirement Patterns	19
3.2 The Anatomy of a Requirement Pattern	21
Basic Details	22
Applicability	23
Discussion	24
Content	24
Template(s)	24
Example(s)	26
Extra Requirements	26
Considerations for Development	28
Considerations for Testing	29
3.3 Domains	29
Domains and Infrastructures	30
3.4 Requirement Pattern Groups	31
3.5 Relationships Between Requirement Patterns	32
Requirement Pattern Classifications	33
Refinement Requirements	35
Divergent Requirement Patterns	36
Requirement Patterns and Diversity of Approaches	36
Use Cases for Requirement Patterns	37
Business Rules and Requirement Patterns	38
4 Using and Producing Requirement Patterns	39
4.1 When and How to Use Requirement Patterns	39
4.2 Tailoring Requirement Patterns	41
4.3 Writing New Requirement Patterns	42
How to Find Candidate Requirement Patterns	43
How to Write a Requirement Pattern	45
Part II Requirement Pattern Catalog	
5 Fundamental Requirement Patterns	51
5.1 Inter-System Interface Requirement Pattern	51
5.2 Inter-System Interaction Requirement Pattern	62

5.3 Technology Requirement Pattern	65
5.4 Comply-with-Standard Requirement Pattern	71
5.5 Refer-to-Requirements Requirement Pattern	79
5.6 Documentation Requirement Pattern	81
6 Information Requirement Patterns	85
6.1 Data Type Requirement Pattern	86
6.2 Data Structure Requirement Pattern	94
6.3 ID Requirement Pattern	97
6.4 Calculation Formula Requirement Pattern	102
6.5 Data Longevity Requirement Pattern	107
6.6 Data Archiving Requirement Pattern	110
7 Data Entity Requirement Patterns	119
7.1 Living Entity Requirement Pattern	129
7.2 Transaction Requirement Pattern	133
7.3 Configuration Requirement Pattern	138
7.4 Chronicle Requirement Pattern	144
7.5 Information Storage Infrastructure	154
Implementation Requirements	154
8 User Function Requirement Patterns	155
8.1 Inquiry Requirement Pattern	156
8.2 Report Requirement Pattern	161
8.3 Accessibility Requirement Pattern	168
8.4 User Interface Infrastructure	187
8.5 Reporting Infrastructure	189
9 Performance Requirement Patterns	191
9.1 Response Time Requirement Pattern	195
9.2 Throughput Requirement Pattern	204
9.3 Dynamic Capacity Requirement Pattern	212
9.4 Static Capacity Requirement Pattern	215
9.5 Availability Requirement Pattern	217
10 Flexibility Requirement Patterns	239
10.1 Scalability Requirement Pattern	241
10.2 Extendability Requirement Pattern	246
10.3 Unparochialness Requirement Pattern	254

10.4 Multiness Requirement Pattern	261
10.5 Multi-Lingual Requirement Pattern.....	272
10.6 Installability Requirement Pattern	274
11 Access Control Requirement Patterns	281
11.1 User Registration Requirement Pattern	284
11.2 User Authentication Requirement Pattern.....	295
11.3 User Authorization Requirement Patterns	305
11.4 Specific Authorization Requirement Pattern	308
11.5 Configurable Authorization Requirement Pattern	313
11.6 Approval Requirement Pattern	318
12 Commercial Requirement Patterns	325
12.1 Multi-Organization Unit Requirement Pattern	325
12.2 Fee/Tax Requirement Pattern.....	330
Glossary.....	341
References.....	349
Index	351

 **What do you think of this book? We want to hear from you!**

Microsoft is interested in hearing your feedback so we can continually improve our books and learning resources for you. To participate in a brief online survey, please visit:

www.microsoft.com/learning/booksurvey/

Part I

Setting the Scene

The four chapters in Part I tell you all you need to know to use the requirement patterns in Part II, "Requirement Pattern Catalog." Part I contains two chapters on requirements in general (on the *how* and the *what*, respectively) and two chapters on what requirement patterns themselves are all about.

This is a book about requirement *patterns*, so we don't want it to be bogged down by long explanations of how to specify requirements and what to include in a requirements specification. Yet you need at least a passing understanding of those subjects to make the most of the requirement patterns. How can we reconcile those goals? The answer is to provide longer, full versions of Chapters 1 and 2—just not in the printed book itself. They are available for download from the book's companion Web site, <http://www.microsoft.com/mspress/companion/9780735623989>. The Chapters 1 and 2 that follow are synopses of the full versions and are organized in the same way. They give you the quickest possible overview of these two subjects. If you want to know more about anything in one of the synopsis chapters, please refer to its full version.

Chapter 1, "Crash Course in Specifying Requirements," is a flying introduction to what requirements are all about and how to figure them out, whether you choose to do so in the traditional manner or take an agile approach. In this context, *traditional* means specifying all the requirements before designing and building the system; *agile* means worrying less about specification documents up front, and beginning development as early as possible.

Chapter 2, "The Contents of a Requirements Specification," describes what a requirements specification needs to contain. The full version of Chapter 2 provides a level of guidance about the sections in a requirements specification that is similar to what the patterns provide for individual requirements. This enables you to write a complete, well-balanced, full requirements specification.

Chapter 3, "Requirement Pattern Concepts," describes the role that requirement patterns play, explains what each pattern contains (its anatomy), and introduces a few related concepts.

Chapter 4, "Using and Producing Requirement Patterns," discusses when and how to use requirement patterns, and describes how to produce new requirement patterns by tailoring existing patterns or by writing new ones from scratch.

