

The Quest for Software Requirements



The Quest for Software Requirements

Roxanne E. Miller

Probing questions to bring
nonfunctional requirements into focus;
proven techniques to get the right
stakeholder involvement.

Copyright © 2009 by Roxanne E. Miller

All rights reserved. No part of the contents of this book may be reproduced or transmitted in any form or by any means without the written permission of the publisher.

Published by MavenMark Books, LLC

For information about special discounts for bulk purchases,
please visit www.MavenMarkBooks.com

Software Requirements Questions, an electronic version of the elicitation questions
contained in this book, is available at www.RequirementsQuest.com

ISBN: 978-1-59598-067-0

Publisher's Cataloging-In-Publication Data
(Prepared by The Donohue Group, Inc.)

Miller, Roxanne E.

The quest for software requirements : probing questions to bring nonfunctional requirements
into focus; proven techniques to get the right stakeholder involvement / Roxanne E. Miller.

p. ; cm.

Includes bibliographical references.

ISBN: 978-1-59598-067-0

1. Computer software--Development. 2. Requirements engineering. I. Title.

QA76.76.D47 M55 2009

005.1

Project Editor: Kira Henschel

Book Cover Design: Timpano Group

Bound and printed in the United States of America.

TABLE OF CONTENTS

Acknowledgments	i
Preface	iii
Purpose and Scope	iii
Who Should Reference This Book	v
Benefits of Using This Book	vi
Skills and Experience Needed by the Reader	vii
How This Book is Organized	vii
What This Book is NOT	ix
Collaboration Beyond This Book.....	ix
 Part One: Right Process, Right People, Right Tools.....	1
1. Requirements in Context	3
1.1 What is a Software Requirement?	4
1.2 Levels of Requirements	6
1.3 A Requirements Management Process	9
1.4 Requirements and the Software Development Lifecycle	12
1.5 A Requirements Framework	19
1.6 What Requirements are NOT	24
1.7 Chapter Summary	29
1.8 Suggested Reading.....	30
2. Involve the Right Stakeholders	31
2.1 Utilize the Right Sources	32
2.2 Engage the Right People	33
2.3 Build a Stakeholder Profile	41
2.4 Chapter Summary.....	52
2.5 Suggested Reading.....	53
3. Interviewing Tips and Tools	55
3.1 Preparing for the Interview	55
3.2 Asking the Right Questions	66
3.3 Conducting a STAR Interview	74
3.4 Chapter Summary	86
3.5 Suggested Reading	87

Part Two: Right Approach, Right Questions	89
The Anatomy of a Nonfunctional Requirement Category.....	91
How to Use the Suggested Questions.....	94
4. Understanding Nonfunctional Requirements	97
4.1 Are the Definitions Dysfunctional?	99
4.2 Nonfunctional Challenges	105
4.3 Vital, Yet Why so Difficult?	106
4.4 Classification Efforts	110
4.5 A User-Focused Approach	118
4.6 Chapter Summary.....	123
4.7 Suggested Reading	124
5. Operation Requirements	125
5.1 Access Security (ACS).....	127
5.2 Availability (AVL)	140
5.3 Efficiency (EFC)	152
5.4 Integrity (INT)	165
5.5 Reliability (REL)	181
5.6 Survivability (SRV)	194
5.7 Usability (USE)	206
5.8 Suggested Reading	223
6. Revision Requirements	225
6.1 Flexibility (FLX)	227
6.2 Maintainability (MNT)	242
6.3 Scalability (SCL)	255
6.4 Verifiability (VER)	265
6.5 Suggested Reading	284
7. Transition Requirements	285
7.1 Interoperability (IOP).....	287
7.2 Portability (POR).....	298
7.3 Reusability (REU)	307
7.4 Suggested Reading	316
References.....	317
About the Author.....	323
About Requirements Quest	325

Access Security (ACS).....	127	5.1 ACS
Availability (AVL)	140	5.2 AVL
Efficiency (EFC)	152	5.3 EFC
Integrity (INT)	165	5.4 INT
Reliability (REL)	181	5.5 REL
Survivability (SRV)	194	5.6 SRV
Usability (USE)	206	5.7 USE
Flexibility (FLX).....	227	6.1 FLX
Maintainability (MNT)	242	6.2 MNT
Scalability (SCL)	255	6.3 SCL
Verifiability (VER)	265	6.4 VER
Interoperability (IOP)	287	7.1 IOP
Portability (POR)	298	7.2 POR
Reusability (REU).....	307	7.3 REU

To my sunshines, Leanne and Tara

To my mother, Barbara

In loving memory:

Lawrence L. Brostowitz, Sr.



ACKNOWLEDGMENTS

This book started as a dream and became a reality thanks to family and friends. A loving thanks to my daughters, Leanne and Tara, whose unconditional love supported me through all the trials and tribulations. Their patience, understanding, and sacrifice of quality time with me have made this and so many other dreams come true. Thanks to my mother, Barbara, for her guidance, faith, and inspiration. She is the person I admire most as a role model for her “survivability,” spiritual strength, relentless persistence, and professional dedication to the education of others. And, thanks to my five sisters and five brothers for the life experiences and “stories” we share.

I am so grateful for the support from Suzanne Ebert and Polly Ann Thomas. They are always there, believing in and encouraging—sometimes even pushing—me to do my best. A sincere thanks to Bruce Opsal, who provided invaluable assistance with research and multiple iterations of, “What do you think of this?” A special thanks to Jim Dawkins and Diane Brussow, who contributed greatly during the early growth stages of Requirements Quest®.

My thanks for the encouragement received from several mentors with whom I have worked or who were willing to coach me over the years. James Minehan remains a true and devoted friend, and recognized the diamond in the rough when presented an “opportunity.” Leilani Allen at times seemed to be merciless, but her vision and tutelage have been the foundation of my career. Steven Davis gave me my first book on requirements and advised me to read it cover to cover, which I actually did. *Software Requirements*, by Karl Wiegers, now in its second edition, remains to be one of my most referenced and recommended books. Karl Wiegers reviewed the

ACKNOWLEDGMENTS

first rough draft of my book—four years ago. Karl is a fabulous role model and continues to assist me in opening new doors of opportunity. My friend Stephen Withall, a wicked reviewer with a wonderful sense of humor, gave me an opportunity to experience a slice of the publishing process, and unknowingly gave me the nudge and confidence I needed to complete this effort.

I have a deep respect for the professional talents and devotion of my reviewers and colleagues in the industry: John Argentiero, Carla Baumgartner, Terri Brouillard, Ginny Cashbaugh, Michael Connor, David DeBruine, Conrad Dennis, Steve Doty, Michael Enstrom, Bahri Gungor, Mai Fei Kinney, Karl Kleifgen, Dennis Kohlmeier, Frank Kowalkowski, Marj Krause, Deanna Mount, Kara Price, Susan Rose-Adametz, Gary Rush, and Wendy Vysoky. Moreover, Deb McCormick was an exceptional, fearless, unrelenting editor. And, I am most appreciative and indebted to Joseph Chapman for his contributions in countless ways.

A special thanks to those with the expertise to pull the book together. Thank you, Kira Henschel, for your wisdom, guidance, and patience throughout the publishing process. Thank you, Melanie Schmidt, for your positive moral support, and your creativity and marketing insight. And, thank you, Jared Gehling, for your expedient, quality service and assistance in converting figures, as well as numerous miscellaneous requests.

It amazes me that, in reality, often the person who teaches learns more than those who are taught. While training on software requirements practices, I have benefited from the experiences, knowledge, and questions of thousands of students, and I thank you all. Moreover, I appreciate my clients for whom I constantly strive to find better ways of doing things to make each individual requirements producer, supplier or receiver, and his or her organization more successful. My techniques have been sharpened through our shared work experiences. It has been my privilege to collaborate with each and every one of you.





PREFACE

IN THIS PREFACE:

Purpose and Scope	iii
Who Should Reference This Book	v
Benefits of Using This Book	vi
Skills and Experience Needed by the Reader.....	vii
How This Book is Organized	vii
What This Book is NOT	ix
Collaboration Beyond This Book.....	ix

PURPOSE AND SCOPE

This book answers a question that is critical to the success of any system development project, “How can the system’s nonfunctional requirements best be defined?”

Simply defined, while functional requirements describe *what the system must do*, the nonfunctional requirements define *how well it must do it*.

Nonfunctional requirements embody important elements of quality for software systems. A thorough specification of nonfunctional requirements creates systems (whether developed or purchased) that:

- ◆ Have fewer errors.
- ◆ Have more realistic time and cost estimates.
- ◆ Result in greater user satisfaction.
- ◆ Are more comprehensive.
- ◆ Better fit business needs.

The Quest for Software Requirements applies a user-focused classification for various categories of nonfunctional requirements. The intent of this book is to bring greater consistency, clarity, and understanding to the term “nonfunctional.” The nonfunctional classification comprises three software user needs:

- ◆ **OPERATION REQUIREMENTS.**
How well does the system perform for daily use?
- ◆ **REVISION REQUIREMENTS.**
How easy is it to correct errors and add functions?
- ◆ **TRANSITION REQUIREMENTS.**
How easy is it to adapt to changes in the technical environment?

These three requirement groups are further subdivided into 14 common nonfunctional categories. Cumulatively, this book provides over **2,000 suggested questions** to help you elicit nonfunctional requirements. In addition, this book offers step-by-step guidance on how to:

- ◆ Build a stakeholder profile and get the right people engaged.
- ◆ Prepare for and conduct a successful requirements-gathering interview.



WHO SHOULD REFERENCE THIS BOOK

This book can help you perform one or more requirements roles, which are introduced here and defined in Chapter 2, “Involve the Right Stakeholders.”

A **requirements producer** is an individual who elicits, analyzes, represents, and validates requirements for desired changes made to business policies, business processes, and supporting systems. The requirements producer is a liaison between the business area with a business need (“What needs to be done?”) and the solution delivery team (“How will the solution be implemented?”).

Those fulfilling the requirements producer role are known by a variety of job titles such as business analyst, business systems analyst, systems analyst, and requirements engineer. Moreover, some people perform more than one role in an organization. Thus, a project manager, lead developer, or software engineer might perform the requirements producer role in defining requirements.

This book can help the requirements producer avoid errors caused by missed requirements. It can help the requirements producer improve the quality and the quantity of gathered requirements by asking more questions to better understand what the business needs.

A **requirements supplier** is any individual with responsibility for defining the business needs. Some individuals (such as business sponsor, product manager, and executive leader) provide the strategy and direction of the system, while others (such as business lead, customer representative, and subject matter expert) provide details that define the necessary system functions. Requirements suppliers may reside in a particular business area or within the technology department. *The Quest for Software Requirements* can help the requirements supplier validate the accuracy, completeness, and quality of the requirements.

A **requirements receiver** is any individual who uses the specified requirements to implement a solution. A requirements receiver might be an individual who is responsible for designing and developing the solution, testing the requirements, training users, writing user procedure manuals, or managing the project. This book can help the requirements receiver to review the requirements, contribute to requirements-related issue resolution, and identify inaccurate or omitted requirements.



BENEFITS OF USING THIS BOOK

This book provides practical solutions that will help you:

- ◆ Understand the vital nature and complexity of nonfunctional requirements.
- ◆ Elicit nonfunctional requirements of the system environment of any enterprise more accurately.
- ◆ Get the right stakeholders involved during a requirements management process.
- ◆ Better prepare for, and conduct, requirements-gathering interviews.

Along with your improved skills, you and your organization will be positioned to experience the following benefits:

- ◆ Greater predictability for project success through reduced requirements errors.
- ◆ Increased opportunity of finding and resolving requirements-related anomalies during the requirements definition phase, and thereby reducing rework in subsequent, more costly phases of the project.
- ◆ Increased customer/client satisfaction because the resulting system will better reflect the user needs and concerns.
- ◆ Better communication between the requirements producer and the requirements suppliers and receivers because the requirements are easier to understand and interpret consistently.



SKILLS AND EXPERIENCE NEEDED BY THE READER

This book can be used by requirements producers, suppliers and receivers regardless of your skills and experience with software requirements activities (such as elicitation, analysis, representation, and validation, which are explained in Chapter 1, “Requirements in Context”). *The Quest for Software Requirements* intentionally tries to stay clear of acronyms, jargon, technology, tools, and methodologies. This book is written in plain language that can be read and absorbed easily by anyone seeking to increase his or her understanding of software requirements.

Reference this book whenever you encounter a software system where one or more of the nonfunctional requirement categories presented might be relevant.

A list of suggested reading is provided in each chapter of this book. You are encouraged to invest time to read one or more of the books. Readers who have read at least one of these recommended books or who are already experienced with software requirements are likely to find this book very beneficial.

HOW THIS BOOK IS ORGANIZED

As the “**Book At A Glance**” diagram inside the front cover illustrates, there are two parts to this book:

PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

The three chapters in Part One are intended to get you started on the *right* path with software requirements.

Chapter 1, Requirements in Context, launches your requirements understanding with a brief overview of software requirements terminology and explains levels of requirements. The chapter also describes nonfunctional requirements in context with software development process models.



Chapter 2, Involve the Right Stakeholders, guides you through the steps to build a stakeholder profile, a technique that helps to identify the knowledge expertise of individual stakeholders. This chapter also helps to uncover business areas and topics that lack expert representation. Essentially, stakeholder profiling will help you get the right stakeholders engaged.

Chapter 3, Interviewing Tips and Tools, offers you several tips and suggested techniques for preparing and conducting a requirements-gathering interview. Ordinary requirements producers, extraordinary interviews!

PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Part Two answers the most frequent question that requirements producers ask about eliciting requirements, “What should I ask?” The four chapters in Part Two are the heart of this book.

Chapter 4, Understanding Nonfunctional Requirements, helps you define nonfunctional requirements, and provides a user-focused classification scheme of 14 common nonfunctional categories.

Chapter 5, Operation Requirements, explains seven operation categories that describe the user need for a software system that performs well for daily use.

Chapter 6, Revision Requirements, defines four revision categories that relate to the user concerns for ease of correcting errors and adding new functions to the software system.

Chapter 7, Transition Requirements, describes three transition categories that define the user need for a software system that adapts easily to its changing technical environment.



WHAT THIS BOOK IS NOT

This is not a book about the process and activities for requirements development or requirements management. While it does include elicitation techniques, this book does not include techniques for analyzing, representing, and validating software requirements. These topics and techniques are sufficiently covered in other good books.

This book does not address any particular business industry. Nor does this book advocate any particular development model, methodology, or requirements management tool. Instead, this book explains a vital part of defining the software systems of any enterprise.

COLLABORATION BEYOND THIS BOOK

“Practice makes perfect,” it is said. Of course, no one has lived long enough to discover whether this is true of software requirements definitions. Through trial and error, you will develop a set of questions that become natural or comfortable for your personal style. You’ll discover for yourself which questions trigger the desired responses that ultimately bring about the highest quality requirements.

Success comes to all of us through collaboration. If you have questions that you find work really well in eliciting nonfunctional requirements, or if you have examples of nonfunctional requirements, you are encouraged to share them with others. Your suggested questions and requirement examples may be sent to Questions@RequirementsQuest.com. Additional questions might be posted on the Internet. Individuals will be recognized for their contributions and might be included with acknowledgment in future editions of *The Quest for Software Requirements*. The suggested elicitation questions contained in this book are available electronically via Microsoft Excel® file entitled *Software Requirements Questions*. To learn more, please visit <http://www.RequirementsQuest.com>.

Cheers to your greater success!

rem





This Requirements Quest® logo (“the hand”), developed by author and expert Roxanne E. Miller, symbolizes the organization’s five-stage Requirements Management Process and emphasizes the importance of user involvement, both internal and external, as a critical factor for project success. The five stages of the process are symbolized by the five fingers on “the hand.” The first four fingers represent the iterative activities of requirements development: Elicitation, Analysis, Representation, and Validation. Upon successful refinement, detailing, and approval, the requirements are baselined in the Change Control stage (represented by the thumb), where requests for change are monitored to maintain scope.



PART ONE

PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

While Part Two is the heart of this book and loaded with suggested questions, the three chapters in Part One tell you all you need to know to use the suggested questions with confidence. That confidence comes from knowing the requirements process, the appropriate people you should interview, and how to be effective in your interviews.

Chapter 1, “Requirements in Context,” describes the relationship of nonfunctional requirements with the requirements hierarchy (three levels of requirements), the requirements management process, software development models, and a requirements framework. It also identifies what should *not* be included in requirements specifications.

Chapter 2, “Involve the Right Stakeholders,” delivers step-by-step instructions for building a stakeholder profile, which will help you identify your stakeholders and why each stakeholder should be engaged on your project. The stakeholder profile is a tangible tool that will help you secure and manage the resources you need, not just the ones you’re told you can use. This chapter answers another question that requirements producers frequently ask, “Who should I ask?”

Chapter 3, “Interviewing Tips and Tools,” provides tips for mastering the most frequently used elicitation techniques. Based on real-life experience, I promise they work. Successful interviews aren’t just for journalists anymore!



The goal of
requirements management
is to obtain one, and only one,
consistent interpretation
of the requirements from all
stakeholders.



1

REQUIREMENTS IN CONTEXT

IN THIS CHAPTER:

1.1 What is a Software Requirement?	4
1.2 Levels of Requirements	6
1.3 A Requirements Management Process	9
1.4 Requirements and the Software Development Lifecycle	12
1.5 A Requirements Framework	19
1.6 What Requirements are NOT	24
1.7 Chapter Summary	29
1.8 Suggested Reading	30

Whether you are building a new in-house system, using commercial off-the-shelf software packages, or making enhancements to existing software, you will apply techniques to discover, illustrate and communicate the requirements. I based this book on a premise that a requirements process—regardless of structure or formality—is used to define the requirements. This process for gathering and specifying requirements can be applied to large and small systems. Additionally, you can apply it whether your system development environment is waterfall, iterative, or agile.

This chapter provides you with an overview of software requirements terminology. It provides a high-level view of requirements and where they fit in software development models



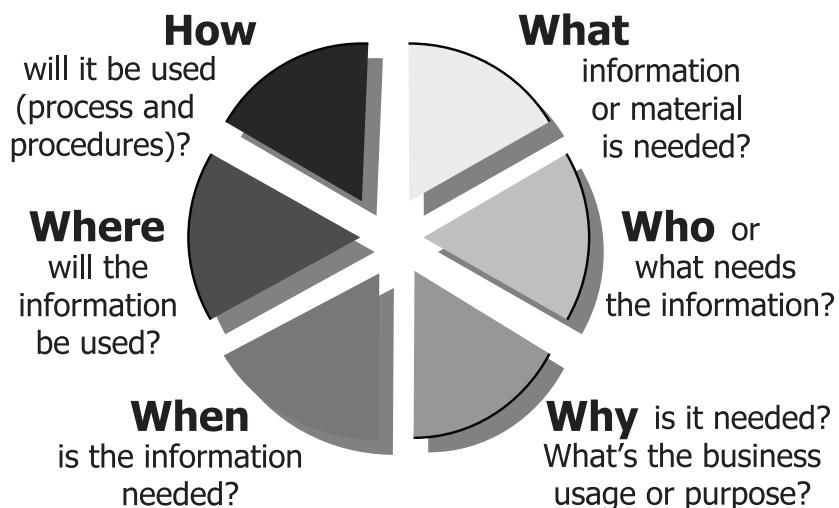
and a requirements framework. I'll start by providing a commonly used definition for the term *software requirement*.

1.1 WHAT IS A SOFTWARE REQUIREMENT?

The *Oxford American Dictionary* defines the noun “requirement” simply as “a need.” In the software development industry, we use the term *requirement* in a broader and in-depth context. In order to identify a complete set of requirements, we must ask questions in six requirements focus areas, as illustrated in Figure 1-1.

The *need* is just a starting place for identifying *what* information is used. From there, we must understand *who* (or what) uses the information and *why* they need it (what business purpose or function they are trying to perform). We also have to understand *when* the information will be used, *where* it will be used, and *how* (process and procedures) it will be used.

Figure 1-1 Requirements Focus Areas



By asking questions in all six focus areas, you minimize the risk of omitting requirements. Due to their importance, the suggested elicitation questions contained in this book are organized by focus area. This requirements framework, as I call it, is explained later in this chapter.

As you'll read in Chapter 4, "Understanding Nonfunctional Requirements," the software industry lacks common definitions for the terms we use to describe our requirements activities. Among the numerous definitions of software requirements you might find, I prefer to use the version given by the Institute of Electronic and Electrical Engineers (IEEE):

"A software requirement is:

- 1) *A condition or capability needed by a user to solve a problem or achieve an objective.*
 - 2) *A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.*
 - 3) *A documented representation of a condition or capability as in (1) or (2).*"
- [IEEE, 1990]

This definition is written from the user's and the developer's view of the requirements. That is, the user is concerned with the external behavior of the system. On the other hand, the developer is more concerned with the internal workings of the system. The term *user* in element (1) of this definition should perhaps be generalized to the term *stakeholder*, as not all stakeholders are users. Simply defined, a *stakeholder* is anyone who is interested in or impacted by the outcome of the system (or product) under development. For example, the manufacturer of dog food refers to the dog as the user because the dog will be consuming the product. However, the dog's owner is considered a stakeholder (and is not a user) in the manufacturing process in order to understand the needs (requirements) and concerns of the customer buying the product.

I think of requirements as a set of graphical and textual representations that describe the functions and attributes of a system environment that provides value to a stakeholder. The perspective of all stakeholder viewpoints must be considered during the development of software requirements.



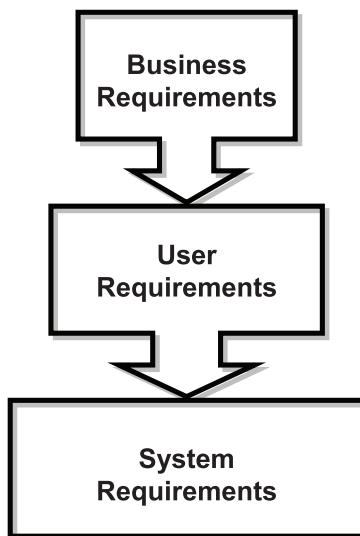
The goal of requirements management is to obtain one, and only one, consistent interpretation of the requirements from all stakeholders.

A stakeholder profiling technique is presented in Chapter 2, “Involve the Right Stakeholders.” The technique helps you identify stakeholder roles (various viewpoints) and get the right stakeholders involved in requirements development.

1.2 LEVELS OF REQUIREMENTS

This section provides definitions for fundamental terms that are used in the software requirements discipline. Software requirements are often grouped into three distinct levels of requirements, or a requirements hierarchy, as shown in Figure 1-2.

Figure 1-2 A Requirements Hierarchy (Levels)



Business-level requirements represent the high-level objectives of the organization, customer or client who requests the system. These requirements typically come from several stakeholders, including but certainly not limited to, the funding sponsor for a project, the acquiring customer, the manager of the users, the marketing department, and a product visionary. The business requirements describe why the organization is implementing the system (why the project is being done), and the objectives the organization hopes to achieve (measurable business benefits that align with the organization's vision).

User-level requirements bridge the needs of the business (business-level requirements) and the requirements of the solution to be developed (system-level requirements). User requirements describe user goals or tasks that the users must be able to perform with the product. User requirements identify what the user will be able to do with the system.

Gause and Weinberg [Gause, 1989] broadly define the term **user** as anyone who affects or is affected by the product. This definition includes people and things (machines, computers, and external systems) that interact with the business process under development, as well as other people and things that receive by-products (information and materials) from the business process or system. To elicit user requirements thoroughly, you may need to include stakeholders other than direct end-users.

System-level requirements describe the top-level requirements for a system that contains multiple subsystems. According to the Institute of Electronic and Electrical Engineers (IEEE), **system** is defined as:

"An interdependent group of people, objects, and procedures constituted to achieve defined objectives or some operational role by performing specified functions. A complete system includes all of the associated equipment, facilities, material, computer programs, firmware, technical documentation, services, and personnel required for operations and support to the degree necessary for self-sufficient use in its intended environment." [IEEE, 1998]



Therefore, a “system” can be a computer, a software application, a subsystem, a whole business enterprise, or a manual process performed by a human.

System-level requirements describe what functions the business process (whether manual or automated) must enable the users to perform (functional requirements), and the attributes and constraints of the environment that affect how well the users perform those functions (nonfunctional requirements).

Provided the product meets its required amount of functionality, the non-functional properties—how usable, convenient, inviting and secure it is—may be the difference between an accepted, well-liked product, and an unused one.

—Suzanne and James Robertson, authors of
Mastering the Requirements Process [Robertson, 2006]

Within the system level requirements, there are two viewpoints: system function/feature and system environment. The ***functional requirements*** (system function/feature view) describe functionality that must be built into the business process to enable the users to perform their business goals or tasks. A “feature” is a set of logically related functional requirements that provides a capability to the user and enables the accomplishment of the user’s business goal.

On the other hand, ***nonfunctional requirements*** describe the system environment view or characteristics of the system. They describe the constraints imposed on the design and construction of the system, as well as quality attributes related to the user need for operation, revision, and ability to handle change or growth.

Some definitions of functional and nonfunctional requirements are explored in Chapter 4, “Understanding Nonfunctional Requirements.” Let’s move on to understand the relationship of the levels of requirements and a requirements management process.



1.3 A REQUIREMENTS MANAGEMENT PROCESS

In simple words, ***software requirements management*** is the process of studying user needs to arrive at a definition of the software requirements. A ***software requirements specification*** generally is used to refer to the document that contains the clear and precise requirements.

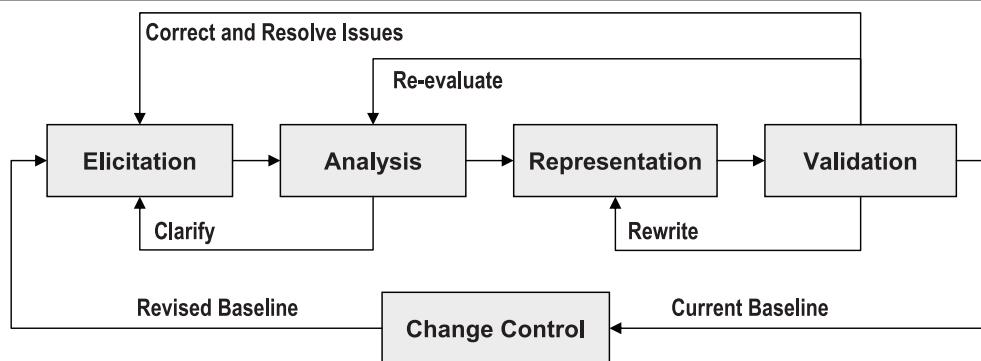
Yet again, inconsistencies in industry terminology exist with regard to what to call the requirements discipline. Some authors refer to the discipline as requirements engineering, while other authors refer to it as requirements management.

Requirements management, as defined by Leffingwell and Widrig, is:

“a systematic approach to eliciting, organizing, and documenting the requirements of the system, and a process that establishes and maintains agreement between the customer and the project team on the changing requirements of the system.” [Leffingwell, 2003]

I prefer the term ***requirements management*** and subdivide the following activities into five components, as shown in Figure 1-3. The first four components are collectively referred to as ***requirements development***.

Figure 1-3 A Five-Stage Requirements Management Process Model (based on [Wiegers, 2003])



The five components of the requirements management process are described as follows:

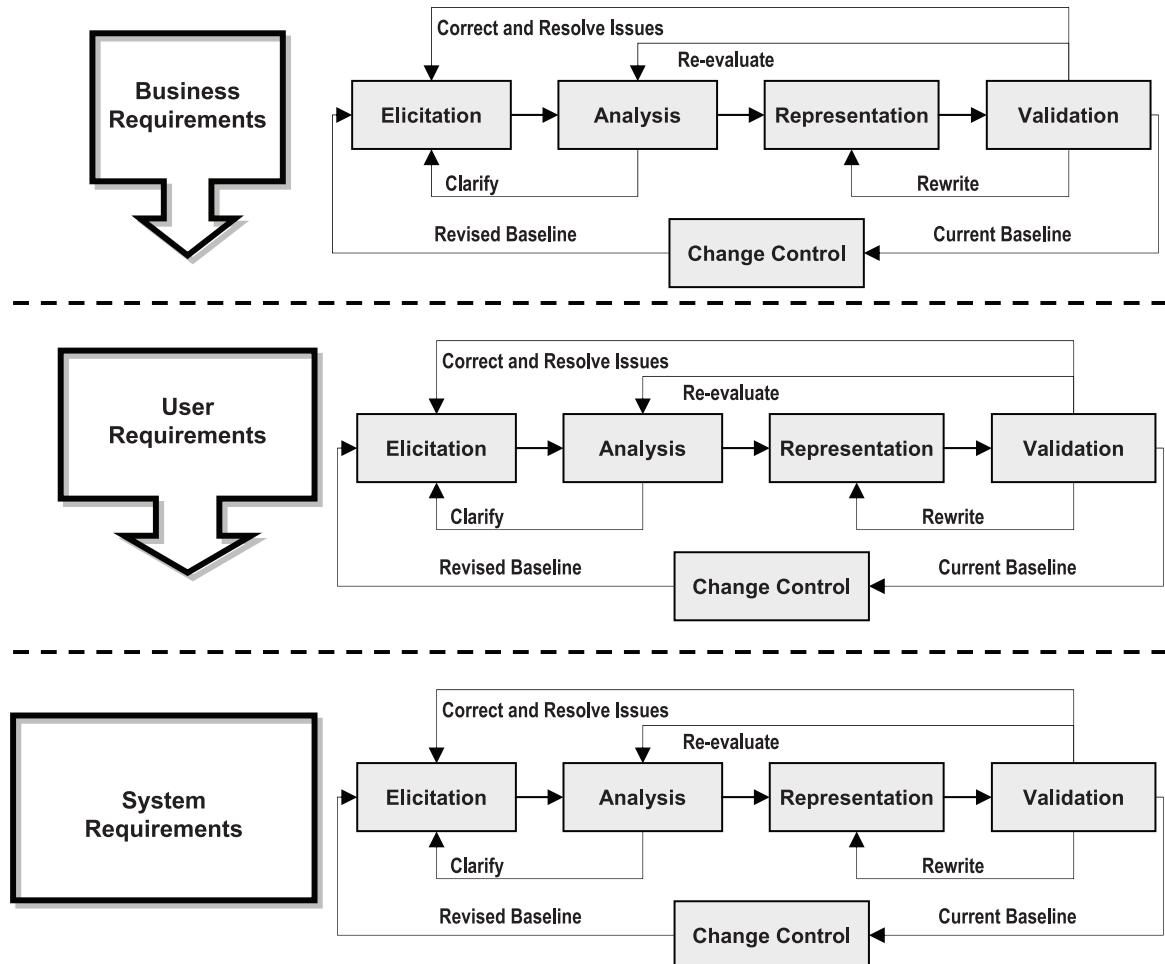
- ◆ **REQUIREMENTS ELICITATION** is the process of identifying stakeholder classes, defining the project scope, identifying requirement sources, and applying techniques to gather the information.
- ◆ **REQUIREMENTS ANALYSIS** is the process of analyzing the information elicited, resolving conflicts, documenting assumptions, constraints, and dependencies, and working with the stakeholders to establish priorities.
- ◆ **REQUIREMENTS REPRESENTATION** (also known as specification) is the process of specifying the business needs using a combination of graphical models and textual documents.
- ◆ **REQUIREMENTS VALIDATION** is the process of reviewing the represented information with the stakeholders for quality characteristics such as completeness, correctness, and feasibility.
- ◆ **CHANGE CONTROL** is the process of maintaining a set of approved or baselined requirements throughout the lifecycle of the development project.

Note in Figure 1-3 that requirements management is an iterative process. The lines connecting the four development stages are not simply linear. At any time in the process, a previous stage can be revisited. For example, during the analysis of information gathered in elicitation, it is discovered that two different stakeholders provided seemingly different responses to the same elicitation question. Revisiting each stakeholder and asking further questions is done to “clarify” the information gathered.

Furthermore, the iterative process is performed at each level in the requirements hierarchy (business, user, and system) as shown in Figure 1-4. As you move from the high-level business requirements to the detailed system requirements, you involve different stakeholders. Thus, within each level, you will elicit, analyze, represent, and validate information provided by the various stakeholders.



Figure 1-4 Requirements Management and the Levels of Requirements



Many organizations and individuals have discovered the benefits of applying a requirements management process such as those described in Table 1-1.

1.4 REQUIREMENTS AND THE SOFTWARE DEVELOPMENT LIFECYCLE

A *software development lifecycle* defines the steps required to develop software systems. That is, the purpose of a software development lifecycle is to establish the order of stages or phases of work activities involved in software development, as well as to determine the entry and exit criteria for transitioning or progressing from one stage to the next. Thus, a software process model addresses two software implementation project questions:

- (1) What should we do next?
- (2) How long should we continue doing it, or when are we done?

Many software project teams experience frustration as a result of performing development tasks in the wrong order. Software process models are important because they provide guidance on the order (stages or phases) in which the project team should perform its major tasks. The names of these stages or phases vary, but the concepts of each are as follows:

- ◆ **PLAN:** The view of the business enterprise as a whole. What is the overall strategy for the system development effort? What is the scope? What are the priorities?
- ◆ **REQUIREMENTS:** The detailed analysis of the business areas impacted. What business functions are performed? What data are required by who, when, where, and why? Who performs each function and how?
- ◆ **DESIGN:** The investigation and application of technology components to meet business needs. In this stage, for example, functions become program specifications or data structures become database designs. At this point, human interfaces are also factored into the proposed solution.



Table 1-1 Benefits of Requirements Management

Business Objective	Benefits of a Requirements Management Process
Satisfy customer needs	<ul style="list-style-type: none">+ Promotes complete, clear, and correct definition of the business requirements.+ Enables the project team to fully understand and meet the needs of customers the FIRST time.+ Enables early identification of missing requirements and requirements deficiencies, ambiguities, and errors.
Meet project deadlines	<ul style="list-style-type: none">+ Provides the method for controlling and prioritizing requirements, which form the basis for creating an accurate project schedule.+ Allows identification of requirements changes that may affect the schedule.+ Reduces scope creep by communicating any changes, and obtaining agreement on project schedule.
Lower project cost	<ul style="list-style-type: none">+ Manages cost by reducing extraneous features.+ Reduces rework by involving business owners and development team members in a formal validation of the requirements early in the project lifecycle.+ Provides the means to more accurately estimate timeframes and work estimates.
Promote communication	<ul style="list-style-type: none">+ Provides a common understanding of where and how requirements documents are to be managed, which promotes and supports reuse.+ Promotes full definition, including the use of supporting information.+ Offers a formal process for proposing and managing changes to requirements.+ Promotes discussing and investigating the impact of changes prior to making them.+ Improves communication between team members and business owners or customers through a formal requirements review process, which helps to ensure customer needs can be met the first time.
Maintain a practical project scope	<ul style="list-style-type: none">+ Promotes tracking relationships between requirements to aid in impact analysis on proposed changes. Keeps project team focused on the goals of the project.
Comply with certification standards (e.g., Capability Maturity Model Integration, or CMMI)	<ul style="list-style-type: none">+ Provides guidelines for documenting requirements in a consistent manner.+ Allows analysis of requirements and requirement relationships.+ Drives the use of requirements as a basis for solid application development and testing.



PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

- ◆ **CODING:** The actual building of the solution system in the coding or construction stage.
- ◆ **TESTING:** The validation and verification activities to demonstrate that the solution system meets the business requirements.
- ◆ **IMPLEMENTATION:** The transition or integration of the system into the new infrastructure of the enterprise. This stage includes activities such as training, definition of new roles and processes, and conversion of existing data.
- ◆ **OPERATION:** The ongoing monitoring and maintenance of the production environment.

With regard to software development lifecycles, the scope of this book concentrates on the requirements phase.

The following are some of the well-known software development lifecycles:

- ◆ Waterfall Model
- ◆ Spiral Model
- ◆ Iterative Approach
- ◆ Extreme Programming (Agile)

The waterfall model and iterative approach are briefly described in this book to give you an understanding of requirements within the broader development lifecycle. There are books and many papers that are solely devoted to describing these models and others in more detail. I encourage you to read some of these to understand software development models more comprehensively.



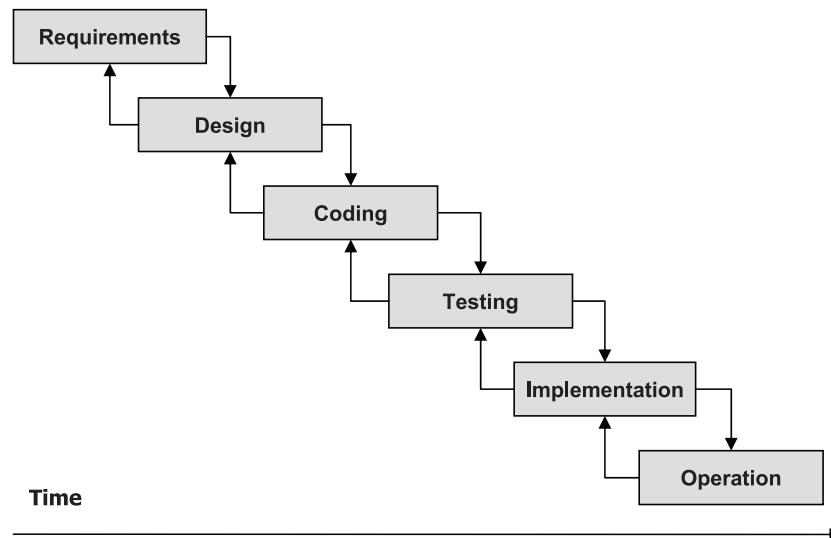
1.4.1 Waterfall Model

As early as the 1950s, the software industry recognized value in developing software in successive stages. Also recognizing that there are increased costs when software defects are revealed late in the development cycle, the industry adopted a logical, step-by-step process model that transitions from a requirements phase, to a design phase, to a coding phase, to a testing phase, and so on.

Under the influential work of Winston Royce in the 1970s, the waterfall model, depicted in Figure 1-5, provided two major enhancements:

- (1) Feedback loops between stages, thereby recognizing that work done in the design phase impacts the requirements, that work done in the coding phase will cause design to be revisited, and so on.
- (2) Prototypes developed in parallel with requirements analysis and design activities.

Figure 1-5 Waterfall Model of Software Development (based on [Royce, 1970])



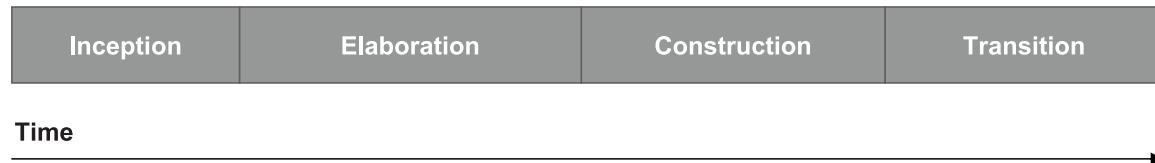
1.4.2 Iterative Approach

Many development teams have migrated to the *Iterative Approach* introduced by Philippe Kruchten [Kruchten, 1999]. It is an approach that is viewed by some to be a combination of the best aspects of the *Waterfall Model* and *Spiral Model* (developed by Barry Boehm [Boehm, 1988]). This view will not be debated or challenged in this book.

In the Waterfall Model, time moves forward through a series of sequential stages or phases. For example, the requirements activities precede design, and design activities precede coding, and so on. Conversely, the lifecycle phases in the Iterative Approach are decoupled from the logical order of activities that occur in each phase, which enables the team to revisit various activities such as requirements, coding, and testing during various iterations of the project. In addition, each successive iteration is intended to mitigate whatever risks are known at that particular iteration.

The Iterative Approach is comprised of four phases (as shown in Figure 1-6): inception, elaboration, construction, and transition. In the inception phase, the scope and feasibility of the project are identified. In the elaboration phase, the requirements are refined and an early feasibility prototype is demonstrated. The focus is on implementation in the construction phase, whereby the coding is typically done, and the system architecture and design are fully developed. In the transition phase, the testing and user training is performed, and the system is deployed to the user community.

Figure 1-6 Phases in an Iterative Approach

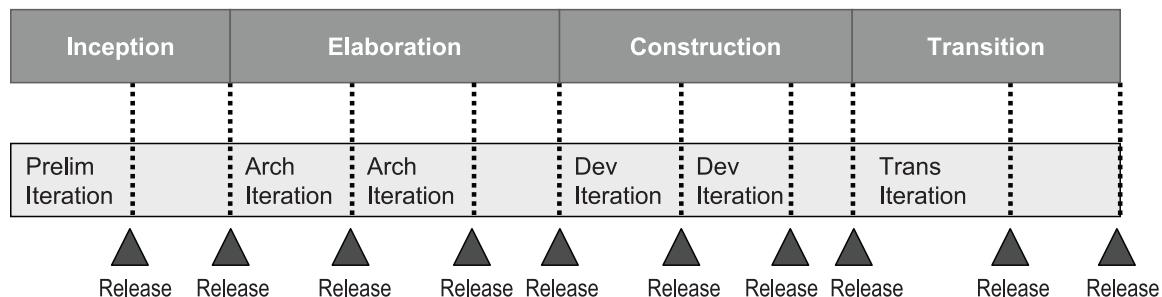


Within each phase, the project typically executes multiple iterations. For example, you can see there are more than two iterations in the construction phase pictured in Figure 1-7. An **iteration** is defined as *a sequence of tasks with an established plan and evaluation criteria that results in some type of executable*. Each iteration builds on the work products produced in the prior iteration. Therefore, the project is released in an “iterative and incremental” manner.

The tasks within an iteration are organized into a set of disciplines (shown in Figure 1-8), such as Requirements, Analysis and Design, Implementation, and so on. Each discipline is made up of a logically related set of tasks, and each defines how the tasks must be sequenced in order to produce a work product or artifact.

During a specific iteration, the team devotes as much time as appropriate in each discipline. The isolated iteration would look very similar to a mini-waterfall. The size of the “hump” shows the relative amount of effort invested in a discipline. For example, Figure 1-8 indicates that a significant amount of time is spent refining the requirements during the three iterations in the elaboration phase.

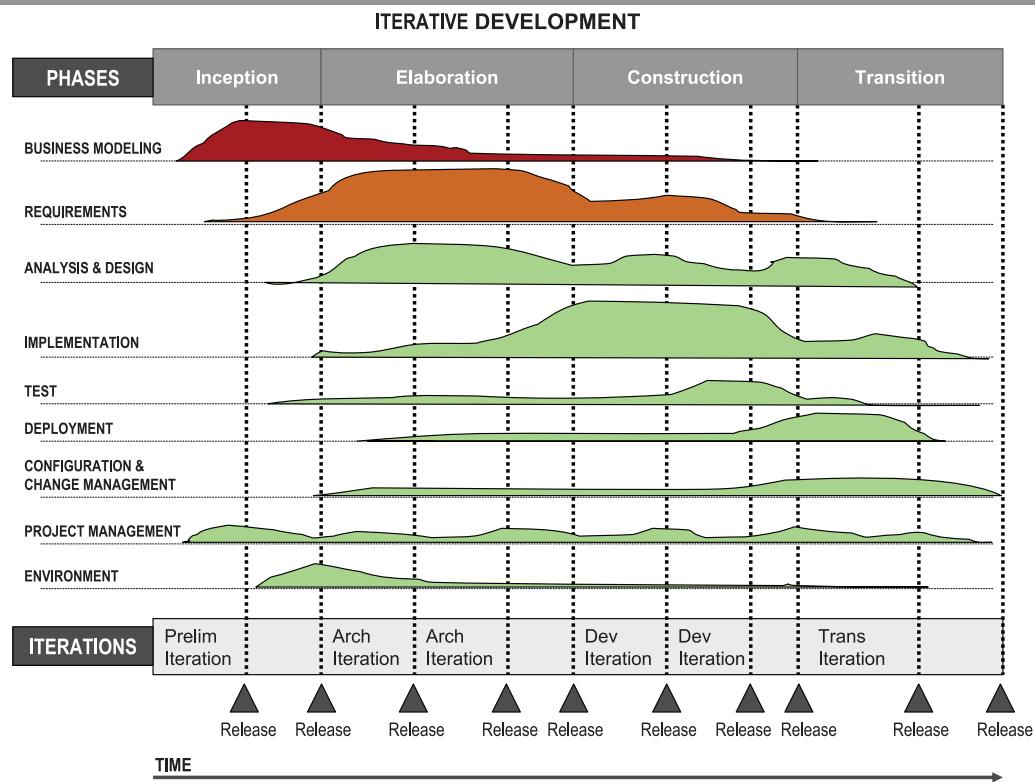
Figure 1-7 Phase Iterations (resulting in variable releases)



PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

The diagram presented in Figure 1-8 was adopted with permission from the Rational Unified Process [IBM, 2009]. The Rational Unified Process, often referred to as RUP, is an iterative software development process framework created by Rational Software Corporation, a division of IBM since February 2003.

Figure 1-8 Disciplines of the Iterative Approach [IBM, 2009]



1.5 A REQUIREMENTS FRAMEWORK

While a *software development lifecycle* defines the stages or phases required to develop a system, a *requirements framework* defines the perspectives of the roles in the development lifecycle, and the view that each role has in the development and implementation of requirements throughout the lifecycle stages.

In 1987, John Zachman published his *Framework for an Information Systems Architecture* [Zachman, 1987]. He makes two significant observations about the software development lifecycle:

- (1) It is important to recognize that systems are developed by distinct groups (consisting of multiple roles) with different viewpoints in parallel with the movement from one development stage or phase to another.
- (2) At each stage in the development lifecycle, one must consider each different perspective or point of view related to data, function, network, people, time, and motivation.

Zachman's framework is typically depicted as a tabular matrix of six rows and six columns. The rows of the matrix represent the different perspectives and the columns represent the areas of interest that are viewed from each perspective. The most recent version of Zachman's diagram and definitions can be viewed online [Zachman, 2009].

While agreeing in general with the meaning behind Zachman's framework, for the sake of simplicity, I respectfully modified the matrix to coincide with the application of its meaning as used in the context of this book. Figure 1-9 shows the translation of Zachman's framework (rows and columns) to my modified version. Succeeding this translation, Table 1-2 on page 22 illustrates the resulting modified version, and will henceforth be referred to here simply as the **REQUIREMENTS FRAMEWORK**. None of my modifications to terminology or organization in any way changes the fundamental structure of Zachman's Framework. The definitions of the modified matrix are presented in the following sub-sections.



PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

Figure 1-9 Zachman Framework Modification Translation

Modified Rows (Perspectives)

Zachman	Modified Version
Scope Strategists	Scope Requirements Supplier
Business Executive Leaders	Business Model Requirements Supplier
System Architects	System Model Requirements Receiver
Technology Engineers	Design Requirements Receiver
Components Technicians	Construction Requirements Receiver
Operations Workers	Operations Requirements Supplier

Modified Columns (Areas of Interest)

Zachman	Inventory (what)	Process (how)	Network (where)	Organization (who)	Timing (when)	Motivation (why)
Modified Version	Data (what)	Process (how)	Logistics (where)	Roles (who)	Timing (when)	Purpose (why)



1.5.1 Requirements Framework Perspectives (Rows)

In the context of this book, the rows of the requirements framework represent the perspectives or viewpoints of the roles in a requirements management process (explained in Chapter 2, “Involve the Right Stakeholders”). As the following explains, the six perspectives in Zachman’s Framework are condensed into two views:

- (1) **REQUIREMENTS SUPPLIER’S VIEW:** *Requirements suppliers* are individuals who are responsible for defining the business needs to be satisfied. This includes supplying all the detail from the initial idea or inception through each incremental iteration of the system development. The *requirements supplier’s view* includes scope, business models, and operations. Scope defines the vision and mission of the enterprise. Business models convey the conceptual nature of the business (structure, processes, organization, and so on). Operations define what the new system is going to be.
- (2) **REQUIREMENTS RECEIVER’S VIEW:** *Requirements receivers* are individuals who receive and use the requirements specifications as the input to develop their work products relating to implementing the requirements. They assist in the requirements process by validating individual requirements, primarily for feasibility, completeness, and conflicts. They can assist in validating additional quality characteristics such as necessity, priority, and traceability. The *requirements receiver’s view* includes system, technology, and components aspects of development. System models are used to explore optional solutions. Technology and construction components are used to demonstrate feasibility of the system under development.



PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

1.5.2 Requirements Framework Areas of Interest (Columns)

All six columns in Zachman's Framework are used in the context of this book. The columns are re-ordered (from left to right), but retain essentially the same meaning and are described as follows:

- ◆ **DATA (what):** This column relates to the information used by the enterprise. Functional requirements describe the data while the nonfunctional requirements are about the data.
- ◆ **ROLES (who):** This column comprises the people and organizations or other systems that are involved or interact with the business enterprise. The functional requirements address the interaction between the role and the business process while the nonfunctional requirements are about taking care of the roles.

Table 1-2 Requirements Framework

	DATA (what)	ROLES (who)	PURPOSE (why)	TIMING (when)	LOGISTICS (where)	PROCESS (how)
Scope, Business Model, Operations <i>Requirements Supplier's View</i>	Identify and define the data used by the business enterprise	Identify and define who performs a role in the business processes	Identify and define the business goals and strategies	Identify and define the events and triggers of business processing	Identify and define the enterprise locations and network	Identify and define processes and procedures of the business
System Model, Design, Construction <i>Requirements Receiver's View</i>	Design, construct, and test data entities	Design, construct, and test interfaces and system work	Design, construct, and test system to enforce business rules	Design, construct, and test system cycles	Design, construct, and test system locations and network	Design, construct, and test system processing



- ◆ **PURPOSE (why):** The motivation behind what the business does is described in this column. The functional requirements describe the ends and means while the nonfunctional requirements describe the environment to support accomplishment of the ends and means.
- ◆ **TIMING (when):** This column identifies the effects of time on the business enterprise. Functional requirements describe the processing of events while nonfunctional requirements describe the timing of the events.
- ◆ **LOGISTICS (where):** Logistics is concerned with the geographical distribution of the data and functions of the enterprise. Functional requirements describe locations while nonfunctional requirements support the locations.
- ◆ **PROCESS (how):** This column incorporates the processes and procedures the enterprise executes. Functional requirements describe the processes while nonfunctional requirements support the processes.

1.5.3 Requirements Framework Applied

The modified Requirements Framework depicted in Table 1-2 is used as the foundation for organizing the lists of suggested elicitation questions contained in Part Two, Chapters 5 through 7. Each nonfunctional category presented in this book includes numerous possible elicitation questions. The questions are first grouped under the six areas of interest (columns) and further separated by perspective (rows).

Applying this Requirements Framework, the “Requirements” phase of a project might be defined as the process of translating the executive leader or business owner’s views of the business enterprise into the development team’s view. This presumes, of course, that this is for a particular project whereby the scope was defined during the “Planning” phase.



1.6 WHAT REQUIREMENTS ARE NOT

Stephen Withall presents one of the most detailed explanations of the contents of a requirements specification that I have yet to encounter—and I do mean detailed. Comparing Withall's recommended structure, as shown in Figure 1-10, with others of equal reputation, you could conclude that there are definitely elements that are NOT identified as content for a requirements specification. Let's take a look at some of the elements that don't belong—despite the fact that they often appear in them.

1.6.1 “Why” and “What” (Requirements)—Not “How” (Design)

Notice that each of the three requirements levels explored previously in Section 1.2 focuses on “why” or “what.” The “why’s” and “what’s” keep focus on the business process and the flow of information and materials in to and out of the business process.

- ◆ Business Level: “Why” is the project being done?
- ◆ User Level: “What” does the user need?
- ◆ System Level: “What” does the system have to do to support the user?
(Functional Requirements) and “What” attributes, constraints, or standards must be applied to the design and construction of the system in order to support the user? (Nonfunctional Requirements).

“How’s” typically describe the design elements or the way in which the system will be constructed. How will the information or material be presented to the user? How will the system receive, store, and retrieve the information that is desired? The “how’s” are the solutions to providing “what” is needed.

For example, during requirements-gathering interviews, I often hear sponsors say that they want a new screen or a new report. The screen or report is the “how,” or the means by which the desired information is presented to the user. The information displayed on the screen or printed on the report is “what” the users really need to perform their role or achieve their goal.



Figure 1-10 Suggested Structure of a Requirements Specification [Withall, 2007]

1. Introduction
 - 1.1 The Purpose of the <System Name> System
 - 1.2 The Purpose of this Document
 - 1.3 The Format of Requirement Definitions
 - 1.4 Glossary
 - 1.5 References
 - 1.6 Document History
2. Content
 - 2.1 Scope
 - 2.2 Major Assumptions
 - 2.3 Major Exclusions
 - 2.4 Key Business Entities
 - 2.5 Infrastructure
3. <Functional Area A Name>
4. <Functional Area B Name>
5. <Functional Area C Name>
6. Major Nonfunctional Capabilities



Certainly, legitimate solution constraints may dictate that the solution be delivered via a screen or a report. Still, the screen and report are “how,” not “what.” It is absolutely essential that the solution team first correctly identify what is needed before a solution is designed and constructed.

1.6.2 Exclude Project-related Information

Project-related information (such as budgets, schedules, implementation plans, project plans, configuration management plans, verification plans, project task lists, and risk management plans) is frequently packaged with the requirements documentation. This is often done for the convenience of any number of project team members in an attempt to centralize information. In general, this must be strictly avoided since changes in the project-related information increase the volatility and foster the tendency for “the requirements” to be out of date. This compromises the integrity of the requirements as they become less trustworthy and more likely to be ignored.

Moreover, the inevitable debates that occur regarding the project-related information should be separated from discussions on what the system is supposed to do. There are different stakeholders who must contribute and participate in the discussions, and the purpose of each discussion is significantly different.

Essentially, if the information doesn’t focus on describing the functionality or behavior of the system or the quality attributes or constraints of the system environment, then the information doesn’t belong in the requirements specification.



1.6.3 Business Rules are Not Requirements

Defined loosely, business rules are the decisions made regarding policies and procedures for how a business will operate day to day. The Business Rules Group's definition is as follows:

A business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the business. [BRG, 2000]

Almost every business in the world has business rules, but the degree to which the rules are documented and accessible vary greatly. The following are commonly applied business rules:

- ◆ An employee is entitled to four weeks of paid vacation following their fifth consecutive year anniversary of full-time employment.
- ◆ An employee with at least one month of employment is entitled to company paid holidays.
- ◆ Employees are paid bi-monthly with 24 pay periods in a calendar year.

While business rules represent the management decisions and policies relating to the operations of the business, requirements relate to a specific application or system that is being considered or developed, which usually automates the business processes.

Requirements are documented at a specific point in time to define what changes are necessary in the specific application or system to enforce any changes in the business rules. Therefore, you will generally derive functional and nonfunctional requirements from business rules in order to enforce the rules or support the regulations and standards that govern the operation of the business.



Let's say, for example, that an organization currently has an automated system to process payroll. The current system, whether a purchased software package or an in-house developed application, is processing according to business rules that dictate 24 payroll periods. That is, employees are paid bi-monthly on the 1st and 16th day of the month (with additional rules to handle non-business processing days throughout the year such as holidays and weekends).

Continuing the example, executive leadership of the organization makes a decision to purchase a new payroll system. A two-year "Payroll Project" is planned and launched to convert from the current system to the new system. Executive leadership has also decided to change the number of payroll periods from 24 to 26. The new business rule is to process payroll on a bi-weekly basis every other Friday (again, taking non-business days into account). The new business rule will take effect upon installation (the go-live date) of the new system.

The requirements specifications created for the Payroll Project will detail all the changes necessary to convert payroll processing from 24 to 26 periods. These detailed requirements will describe the necessary changes to processes, sub-processes, procedures, calculations, event triggers, error and exception handling, and so on. The requirements describe what the new payroll system must do (functional) and how well it will do it (nonfunctional).

1.6.4 Standards are Business Rules (which are NOT Requirements)

Standards are the written or unwritten business rules, regulations, and/or contracts that dictate the business operations. Standards are typically set by upper management to guide the day-to-day business decisions.

Standards are the required level of quality that the product or system must measure against or comply with. Standards are often imposed on a business by external government or regulatory agencies. In these instances, the imposed standards must be adopted by the business as business rules (whether they like it or not). If the business does not operate within compliance of these forced business rules, the business risks going out of business.

Sources for standards, however, are not to be overlooked when identifying requirements stakeholders. Examples of sources for standards are included in Chapter 2, "Involve the Right Stakeholders."



1.7 CHAPTER SUMMARY

- ◆ Software requirements are grouped into three distinct levels: business (why is this project being done), user (what information and materials are needed to conduct business), and system (what should the system do—functional, and how well should it do it—nonfunctional).
- ◆ A requirements management process involves the iterative activities for eliciting, analyzing, specifying, validating, and managing changes to requirements. This iterative process is applied beginning at the high-level business requirements, through the user requirements level, and on to the detailed system level requirements.
- ◆ A number of software development models exist and help a project team order the phases of work. The requirements management discipline is one of the various work phases. Thus, the requirements management process is performed regardless of which specific development model is used (unless, of course, the requirements phase is ignored). The timing and formalities in specifying the requirements may differ from model to model, but the requirements process is still applied.
- ◆ While a development lifecycle focuses on the stages or phases of a project effort, the architecture framework asserts that it is necessary to recognize the varied perspectives or views of the roles that perform the development process. Applying the architecture framework not only helps to identify project stakeholders, but also provides guidance for the line of questions that relate to the various views held by the stakeholders.
- ◆ In order to better understand requirements, it is necessary to explore what are not requirements such as project-related information, design, business rules and standards.



1.8 SUGGESTED READING

A Guide to the Business Analysis Body of Knowledge, Version 2.0, International Institute of Business Analysis, [IIBA, 2009]. The BABOK® is an excellent source of **requirements activities and defined terms**.

CMMI®: Guidelines for Process Integration and Product Improvement, 2nd Edition, by Mary Beth Chrissis, Mike Konrad, and Sandy Shrum, [Chrissis, 2007]. **Requirements Management** is an engineering process area at maturity level 2, and **Requirements Development** is an engineering process area at maturity level 3.

Managing Software Requirements: A Use Case Approach, 2nd Edition, by Dean Leffingwell and Don Widrig, [Leffingwell, 2003]. This book provides a nice **A-to-Z overview of software requirements**.

Mastering the Requirements Process, 2nd Edition, by Suzanne Robertson and James Robertson, [Robertson, 2006]. This book presents an effective **overview of the requirements management process**, although focusing mostly on specifying requirements using their Volere approach.

Software Requirement Patterns, by Stephen Withall, [Withall, 2007]. This book is an **excellent source for real-life requirement examples**. Withall reports that as much as 65 percent of requirements crop up over and over across systems. This book presents the templates for 37 requirement patterns grouped into eight domains to help you **write better requirements**.

Software Requirements, 2nd Edition, by Karl Wiegers, [Wiegers, 2003]. Chapter 4 provides a generous explanation of the industry **role of the requirements analyst**. Overall, the book provides a **good end-to-end understanding of requirements engineering**, requirements development and requirements management.





2

INVOLVE THE RIGHT STAKEHOLDERS

IN THIS CHAPTER:

2.1 Utilize the Right Sources	32
2.2 Engage the Right People	33
2.3 Build a Stakeholder Profile	41
2.4 Chapter Summary	52
2.5 Suggested Reading	53

Lack of user input, incomplete requirements, and changing requirements are the major reasons why information technology projects do not deliver all of their planned functionality on schedule and within budget.

—The Standish Group International [Standish, 1995]

Far too often, development teams make the mistake of leaving an essential person out of the requirements process. Before describing ways to identify and involve the right people in the requirements definition process, I want to emphasize that people are not the only source for requirements.



2.1 UTILIZE THE RIGHT SOURCES

Requirements can be identified through several sources. However, keep in mind that it is unlikely you would use all of them on a particular project effort. You might consider using the following sources:

- ◆ **ENTERPRISE ANALYSIS** to gain an understanding of the current state of the business process. This is usually done by creating a variety of models that help people “see” their business.
- ◆ **REQUIREMENTS DOCUMENTS** for current systems. Searching requirements specifications from previous projects can uncover potentially reusable material, thus saving time and duplication of effort.
- ◆ **PROBLEM REPORTS AND ENHANCEMENT REQUESTS** for a current application.
- ◆ **FORM ANALYSIS** to understand the communication channels in the organization. A form is simply a way to structure data entry and retrieval.
- ◆ **BUSINESS ARTIFACTS** such as training manuals, end-user documentation, and procedure guides.
- ◆ **MARKETING SURVEYS AND USER QUESTIONNAIRES**. These are great techniques to use when there is a very broad audience of stakeholders with little or no opportunity (usually due to time and cost) to elicit from all of them.
- ◆ **OBSERVING USERS AT WORK**. Observation may be performed in an active-visible manner (the observer is actively interjecting questions while the user is working) or a passive-invisible manner (the observer is silent and non-intrusive).
- ◆ **EXPERIENCING LIFE AS A USER**. Figuratively, walk in the user’s shoes for a day. This is only possible when the work is not too complex or dangerous to be taken on by a novice. Working with users can help you understand the problems they encounter.



- ◆ **SCENARIO ANALYSIS** of user tasks. In the most general sense, a scenario is a step-by-step account of what an end-user does to achieve a desired goal. A set of logically-related scenarios (associated to the same user goal) can help the development team see the big picture of the work performed.
- ◆ **INTERVIEWS** and discussions with end-users and stakeholders. This is the most direct way to find out what the users need—ask them. While there are many elicitation techniques to choose from, I've included a thorough explanation of interviewing in Chapter 3, "Interviewing Tips and Tools."
- ◆ **WORKSHOPS** with a group of stakeholders bring the right collection of people together, and enable them to correct and improve on their own requirements (increasing buy-in). The workshop approach is a cycle, in which all participants contribute in building the requirements deliverable.

Whether requirements are gathered from one or all of the above sources depends on what defines the critical success of the project, the resources available (monetary, personnel, software and hardware, old documents), and any schedule constraints under which the product or service must be delivered. Based on the risks of developing the software system, the team must select the appropriate combination of resources to tap into.

2.2 ENGAGE THE RIGHT PEOPLE

The key to a successful project is involving all the right people. Certainly this includes individuals who are part of the project team itself. Additionally, there will be individuals who are invited to participate that are external to a particular business department or to the organization.

2.2.1 Stakeholders: Client vs. Customer vs. End-User

The terms client, customer, and end-user (informally called user) are often used differently from one organization to another or from one industry to another. Therefore, I feel it is necessary to establish their meanings as used in the context of this book. It might be important for you to



PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

establish consensus on these terms within your organization and certainly within your project team.

I use the term ***client*** to define the person or organization (indeed, some clients are external) which is invoiced or paying the bill for the actual development and/or implementation costs of the system. The client is the department whose budget will carry the cost of the project effort. In many organizations, the client is referred to as the project sponsor.

The ***customer***, on the other hand, is the person who buys a system or service that is sold for public use. The customer may be synonymous with user if the customer is ultimately the person who uses the system.

In software development models, an ***end-user*** is any person who is affected by or has an affect on the system being built, or is any system that needs to interface with the system under development.

It is critical to involve a variety of stakeholders, consisting of individuals who might be a client (sponsor), a customer, and an end-user, to define a set of requirements that are complete.

A ***stakeholder*** is any person who has:

- ◆ A vested interest and/or decision-making authority in the development of the system.
- ◆ An expert understanding of the current process.
- ◆ The vision of what the future process should be.
- ◆ A role in the development process.
- ◆ A vested interest as a direct or indirect end-user of the system.

Industry research and reports tell us year after year of the importance of user involvement as a project success factor. This chapter is devoted to helping you achieve project success by getting the right people involved.



2.2.2 Potential Stakeholders

There is a saying, “two heads are better than one.” It is the collaborative efforts of all the stakeholders that dramatically increase the accuracy of the requirements gathered, and significantly reduce the likelihood that requirements are missed. Involving the end-users and other key stakeholders early in the requirements-gathering process, as well as throughout the development and implementation lifecycle, increases their commitment or buy-in into the requirements. This typically results in fewer changes to the requirements during the later development phases when the cost to change requirements is significantly higher.

I use a simple classification of requirements management roles as shown in Figure 2-1. I recommend using a classification such as the one here to organize and identify potential stakeholders at the start of a project and throughout the development and implementation phases.

Figure 2-1 Requirements Management Roles

Requirements Producer	Requirements Supplier
Requirements Supporter	Requirements Receiver

This grouping of stakeholder roles serves as a checklist or tickler of those who should be involved in the requirements development process. I suggest that the stakeholders who are involved in the early planning phase of the project (which typically includes the project sponsor, project manager, and business analyst) conduct a brainstorming session to identify a starter list of stakeholders. The following list identifies potential stakeholders by the role(s) they may play in the development of the system:



(1) **REQUIREMENTS PRODUCER.** This is any individual who is responsible for capturing the requirements (eliciting, analyzing, representing, and coordinating the validating and approval of the requirements). “Producers” are also responsible for managing the requirements to ensure compliance and integration throughout the development or installation of the system. ***Business Analysis*** is a term commonly used to encompass the activities performed by the requirements producer. Examples of requirements producers in the area of requirements management include the following:

- ◆ Requirements Architect
- ◆ Requirements Engineer
- ◆ Business Analyst
- ◆ Data Analyst
- ◆ Network Engineer
- ◆ System Analyst/Engineer
- ◆ Software Analyst/Engineer

(2) **REQUIREMENTS SUPPLIER.** The supplier is any individual who is responsible for defining the business needs to be satisfied. This includes supplying all the detail from the initial idea or inception through each incremental iteration of the system development and implementation. Examples of requirements suppliers are as follows:

- ◆ Business Owner or Business Initiator
- ◆ Business Model Analyst
- ◆ Project Sponsor or Champion
- ◆ Product Manager



- ◆ Functional Manager
 - ◆ Key Executive Participants
 - ◆ Business Coordinator
 - ◆ Business Team Member
 - ◆ Business Subject Matter Expert (SME) or Business Area Knowledge Expert
 - ◆ Technical Subject Matter Expert (SME) or Application Expert
 - ◆ Customer Representative
- (3) **REQUIREMENTS RECEIVER.** A receiver is any individual who receives and uses the requirements specifications as the input to develop their work products relating to implementing the requirements. “Receivers” assist in the requirements process by validating individual requirements, primarily for feasibility, completeness, and conflicts. Receivers can assist in validating additional quality characteristics such as necessity, priority, and traceability. Following are examples of requirements receivers:
- ◆ Project Manager
 - ◆ Data Warehouse Specialist
 - ◆ Database Administrator
 - ◆ Developer (Programmer)
 - ◆ Usability Engineer
 - ◆ Network Planner
 - ◆ Operations Analyst
 - ◆ System Designer



- ◆ Technical Architect
- ◆ Test Analyst/ Quality Control/ Tester
- ◆ End-user Trainer
- ◆ Writer or Editor

(4) **REQUIREMENTS SUPPORTER.** The requirements supporter is any individual responsible for supporting the requirements management process and project effort. “Supporters” may be involved in looking at both the completed requirements specifications and the requirements processes that are followed to judge their effectiveness for future projects. “Supporters” provide support in the capacity of managing, controlling, or sponsoring the effort. They assist the individuals who are responsible for eliciting, analyzing, representing, validating, or managing the requirements set for the project. Examples of requirements supporters are as follows:

- ◆ Project Manager
- ◆ Program Manager
- ◆ Project Director
- ◆ Facilitator
- ◆ Process Owner
- ◆ Quality Assurance
- ◆ Practice Leader



2.2.3 Institutionalized Stakeholder List

I recommend that each organization create a defined list of stakeholders that are appropriate for their specific business. For example, Table 2-1 provides simple examples of stakeholder roles in three different industries. Undoubtedly, these roles would vary considerably when applied to systems in different departments across the organization.

Each organization should name the roles that are meaningful to the business, and then provide a definition for each role so that the terms are used consistently. Figure 2-2 on the following page lists a starter set of examples to consider in creating a stakeholder roles checklist.

Table 2-1 Stakeholder Role Examples by Industry

Banking Roles	Insurance Roles	Higher Education Roles
Teller	Agent	Student
Personal Banker	Policyholder	Professor
Examiner	Claim Adjuster	Chancellor
Borrower	Insured	Dean
Lender	Beneficiary	Alumni
Loan Officer	Claimant	Undergraduate
Account Holder	Attorney	Advisor
Co-signer	Physician	Scholarship Donor
Trustee	Witness	Resident Assistant



PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

Figure 2-2 Stakeholder Roles Checklist

Requirements Suppliers:	Requirements Receivers:	Requirements Supporters:
<ul style="list-style-type: none"><input type="checkbox"/> Sponsor/Champion/Client<input type="checkbox"/> End-User/Customer<input type="checkbox"/> Business Subject Matter Expert (SME)<input type="checkbox"/> Business Process Area Experts<input type="checkbox"/> Technical Subject Matter Expert (SME)<input type="checkbox"/> Government Authority<input type="checkbox"/> Regulatory or Compliance Authority<input type="checkbox"/> Industry Standards Authority<input type="checkbox"/> Special Interest Groups<input type="checkbox"/> Cultural Interest Groups<input type="checkbox"/> Public Opinion Representatives<input type="checkbox"/> Professional Organizations<input type="checkbox"/> Market Analysts<input type="checkbox"/> System End-users<input type="checkbox"/> System Buyers<input type="checkbox"/> Recycling and Waste Managers<input type="checkbox"/> Usability and Efficiency Experts<input type="checkbox"/> Business Support Departments<ul style="list-style-type: none"><input type="checkbox"/> Audit<input type="checkbox"/> Sales<input type="checkbox"/> Marketing<input type="checkbox"/> Accounting<input type="checkbox"/> Legal<input type="checkbox"/> Human Resources<input type="checkbox"/> Mail Room<input type="checkbox"/> Test Market Representatives<input type="checkbox"/> Inspectors<input type="checkbox"/> Business Architect<input type="checkbox"/> Business Strategist	<ul style="list-style-type: none"><input type="checkbox"/> User Acceptance Test Group<input type="checkbox"/> Development Team Members<ul style="list-style-type: none"><input type="checkbox"/> System Architect<input type="checkbox"/> Quality Assurance<input type="checkbox"/> System Analyst<input type="checkbox"/> Designer<input type="checkbox"/> Developer (Programmer)<input type="checkbox"/> Database Administrator (DBA)<input type="checkbox"/> Data Warehouse Specialist<input type="checkbox"/> Tester<input type="checkbox"/> Release Coordinator<input type="checkbox"/> Technical Writer<input type="checkbox"/> Production Support Personnel<input type="checkbox"/> End-user Trainer or Training Personnel<input type="checkbox"/> Network Planner<input type="checkbox"/> Usability Engineer<input type="checkbox"/> Business Operations Support Personnel<input type="checkbox"/> Technical Operations Support Personnel<input type="checkbox"/> Implementation Architect<input type="checkbox"/> Configuration Management<input type="checkbox"/> Product Disposers	<ul style="list-style-type: none"><input type="checkbox"/> Project Sponsor<input type="checkbox"/> Business Process Owner<input type="checkbox"/> Project Manager<input type="checkbox"/> Requirements Management Process Owner<input type="checkbox"/> Project Team Members<input type="checkbox"/> Implementation Support Team<input type="checkbox"/> Project Investors<input type="checkbox"/> Maintenance and Service Staff



The stakeholder roles identified for your specific organization must also be defined. This will help people to use the terms consistently and avoid potential confusion or requirement errors on projects.

Furthermore, each organization should maintain a list of government, regulatory, and industry standards sources (including individuals/departments to contact and contact information) that are specifically applicable to the organization's products and services. Representatives from these industry standard agencies are usually involved as requirements suppliers. You will probably recognize some of the following examples of industry standard sources:

- ◆ ANSI—American National Standards Institute
- ◆ IEEE—Institute of Electrical and Electronic Engineers
- ◆ ISO—International Standards Organization

2.3 BUILD A STAKEHOLDER PROFILE

Stakeholder profiling is a very effective, yet simple to use, technique for identifying stakeholders. A **stakeholder profile** helps to get the appropriate representation of all interested and affected areas. The stakeholder profile might be used to identify the appropriate individuals to involve on the project team itself, and to identify the appropriate individuals as resources for requirements. The key to a successful stakeholder profile is not only to identify an individual by name and the area that they represent, but also to define the role and responsibilities that the individual will fulfill on the project. The following are step-by-step procedures to help you engage the right stakeholders (perhaps what the National Aeronautics and Space Administration, NASA, terms the *right stuff*).



PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

2.3.1 Step 1: Identify Stakeholder Roles

The stakeholder profile begins by identifying a list of potential stakeholders using a three-column tabular format such as Table 2-2, which provides a simple example of a Potential Stakeholder List. Reading from left to right, the columns are as follows:

- ◆ **STAKEHOLDER NAME.** This column is used to identify each potential stakeholder.
- ◆ **TOPIC OF EXPERTISE.** The middle column is used to identify the area or organizational department the stakeholder represents, as well as the specific business expertise or knowledge needed for the scope of the project.

Table 2-2 Potential Stakeholder List

Stakeholder Name	Topic of Expertise	Stakeholder Role
Michael	Workers compensation line of business management	Sponsor
Joseph	Claim assignment business rules	Subject Matter Expert
Anthony	Reserving procedures	Subject Matter Expert
Sayid	Subrogation procedures	Subject Matter Expert
Elizabeth	Reinsurance procedures	Subject Matter Expert
Jinlee	Monthly state reporting	Accounting
Lucas, Maria	Policy pricing based on loss experience	Actuarial
Anthony	Claim investigation and resolution	Claim Adjuster



- ◆ **STAKEHOLDER ROLE.** This column is used to describe the stakeholder roles in the requirements management process and their specific interest in the development of the system or product.

The route taken to complete the deliverable is of no real concern to the audience who benefits from the finished product. As is the case with so many templates used to create a deliverable, the sequence taken to build it is not the same sequence in which the deliverable is read. The potential stakeholder list is completed by working backwards—filling in the columns from right to left. Thus, completing the “Stakeholder Role” column is the first step in the procedures to build the stakeholder profile.

Start with your institutionalized Stakeholder Roles Checklist (Figure 2-2 on page 40) and select the roles appropriate for the scope of the project as typically not all roles will apply. As shown in Table 2-2, note that some stakeholder roles might apply more than once depending on the scope of topics for the particular project (e.g., Subject Matter Expert).

2.3.2 Step 2: Brainstorm Topics of Expertise

This is the most critical of the six steps to building the stakeholder profile. Identifying *why* a stakeholder might participate significantly decreases the likelihood of leaving out a key contributor. Forgetting a key contributor altogether or bringing them in late in the development lifecycle almost inevitably results in costly rework.

I recommend that the requirements producer meet with an initial subset of stakeholders such as the sponsor, project manager, functional manager, a lead business subject matter expert, and a technical lead to discuss the project scope. Current-state business models and workflow diagrams are reviewed for impacted business areas. The outcome of the meeting is a list of business areas impacted and the specific topics of business and technical knowledge that must be represented by the stakeholders selected to participate. Use the resulting list of topics to complete the column headed “Topic of Expertise” as shown in Table 2-2.



Tip 2-1: Use visual aids to communicate

Most people communicate more effectively when presented with a visual aid or picture. Using simple requirements models such as relationship maps, process maps and event-response tables will significantly increase the communication effectiveness of the stakeholders. Similarly, scope models such as context diagrams and use case diagrams will also help the stakeholders see what business processes and sub-processes are affected.

2.3.3 Step 3: List the Names of Potential Stakeholders

The “Stakeholder Name” column is completed by identifying one or more individuals believed to be knowledgeable in each topic of expertise. In Table 2-2, for example, both Lucas and Maria are named as potential stakeholders with knowledge of “policy pricing based on loss experience.” Furthermore, one individual stakeholder (e.g., Anthony) might be named as knowledgeable in more than one business topic.

In this step, it is important to avoid pre-screening out potential stakeholders due to assumed unavailability. Try to approach this step as if it were a perfect world, and anyone you choose will be freed up to work on the project. Keep focused on the purpose of this step, which is simply identifying *potential* stakeholders. The last three steps of the six-step procedure will confirm which stakeholders participate.

2.3.4 Step 4: Survey Each Potential Stakeholder

While the first three steps of the procedures to build the stakeholder profile result in a collective list of several individuals representing various areas and roles, the stakeholder survey relates to a single stakeholder or stakeholder group representative. The stakeholder survey is used to identify the depth of knowledge that a potential stakeholder has in each of the topics of expertise needed for a particular project.



Tip 2-2: Create a “pecking order” of stakeholders

Try using the following list as a “pecking order” to identify potential stakeholders:

1. Start with the individuals assigned to the project team.
2. Next, look for individuals from within the project sponsor's business area or process.
3. Next, look within the sponsor's “partner” support business or technical areas.
4. Then, turn to others inside the company.
5. Finally, reach out to others outside the company.

This order implies that resources are decreasing in availability while increasing in possible costs associated with availability.

The template for the stakeholder survey, shown in Table 2-3 on page 47, is created by replicating the “Topic of Expertise” column from Step 2.

Each potential stakeholder is asked to complete the stakeholder survey and indicate his or her level of expertise in each topic of expertise listed for the project. Four degrees of experience might be used as follows:

- (1) Little = very limited or no experience.
- (2) Some = apprentice level or enough to be dangerous.
- (3) Proficient = familiar with topic and good understanding of the day-to-day tasks.
- (4) Expert = solid knowledge; person everyone goes to for help on the topic. (It doesn't necessarily mean the person knows it all.)



Tip 2-3: Build rapport

I prefer to spend about 20 minutes with the potential stakeholder in an interview setting and ask him or her to complete the stakeholder survey. It is an opportunity to build rapport and explain what the technique is and how it will be used, as well as what it is not (for example, it is not going to be used to evaluate the person's performance). If the stakeholder is not available for a face-to-face interview, call the person on the phone. Don't just send the survey in an email—make personal contact first.

After the survey has been completed, ask for other potential stakeholders. For example, if my interviewee indicates having little, no, or some experience, in any of the topics, I will ask, “Is there anyone else you would go to with questions on these topics?” Any named individuals will be added to the Potential Stakeholder List and will also be asked to complete the stakeholder survey.

Finally, ask each potential stakeholder to review the list of topics and ask, “Based on your understanding of the scope of this project, what other topics of expertise should be added? Removed?”

The key to this step is having each potential stakeholder fill out the survey for his or her expertise. The survey is not filled out by the requirements producer, project manager, project sponsor, or any other role involved in the project. If stakeholders inflate their experience level in any given topic area, you'll become aware of it during the interviews and meetings on the specific topic. On the other hand, if stakeholders deflate their expertise (say, in an attempt to avoid participating in meetings), other stakeholders will name that individual as an expert so you'll probably end up inviting them anyway.



CHAPTER 2: INVOLVE THE RIGHT STAKEHOLDERS

Table 2-3 Stakeholder Survey Example

Stakeholder Name: Anthony (claim adjuster)

Topic of Expertise	Little	Some	Proficient	Expert
Workers compensation line of business management	✓			
Claim assignment business rules	✓			
Reserving procedures				✓
Subrogation procedures			✓	
Reinsurance procedures		✓		
Monthly state reporting		✓		
Policy pricing based on loss experience	✓			
Claim investigation and resolution			✓	



PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

2.3.5 Step 5: Analyze the Stakeholder Surveys

The stakeholder surveys are compiled in a tabular format such as Table 2-4. This table uses the first letter of each degree of experience. For example, where the stakeholder indicated little or no experience, an “L” was inserted accordingly.

Table 2-4 Stakeholder Profile Assessment

Topic of Expertise	Anthony	Elizabeth	Jinlee	Lucas	Michael	Sayid	Joseph	Maria
Workers compensation line of business management	L	L	L	L	E	L	L	L
Claim assignment business rules	L	L	L	L	P	L	E	S
Reserving procedures	E	S	L	P	E	S	S	P
Subrogation procedures	P	L	L	L	L	E	S	L
Reinsurance procedures	S	E	L	S	S	L	S	L
Monthly state reporting	S	L	E	L	S	L	L	S
Policy pricing based on loss experience	L	L	L	E	L	L	L	P
Claim investigation and resolution	E	L	L	L	E	S	S	S

Key: L = Little or no experience S = Some experience P = Proficient E = Expert



CHAPTER 2: INVOLVE THE RIGHT STAKEHOLDERS

Alternatively, you might use a number to correspond to each degree of experience as shown in Table 2-5. For example, the number 1 signifies little or no experience, while the number 4 equates to the expert level.

Table 2-5 Stakeholder Profile Assessment (Alternative)

Topic of Expertise	Anthony	Elizabeth	Jinlee	Lucas	Michael	Sayid	Joseph	Maria
Workers compensation line of business management	1	1	1	1	4	1	1	1
Claim assignment business rules	1	1	1	1	3	1	4	2
Reserving procedures	4	2	1	3	4	2	2	3
Subrogation procedures	3	1	1	1	1	4	2	1
Reinsurance procedures	2	4	1	2	2	1	2	1
Monthly state reporting	2	1	4	1	2	1	1	2
Policy pricing based on loss experience	1	1	1	4	1	1	1	3
Claim investigation and resolution	4	1	1	1	4	2	2	2

Key: 1 = Little or no experience 2 = Some experience 3 = Proficient 4 = Expert



The stakeholder profile assessment is analyzed to make sure that all business topic areas for the scope of the project are adequately represented. Each row (relating to a specific topic area) is reviewed for adequate coverage. Ideally, you want to have at least two stakeholders in each topic that are either proficient or expert. This assumes that the saying, “two heads are better than one,” holds true. Finding at least two stakeholders in each topic also helps alleviate some of the inflated egos identified in Step 4. Scanning Table 2-4, you should see that I might want to pick up a few more stakeholders in at least three of the topic areas: workers compensation line of business management, reinsurance procedures, and monthly date reporting.

The stakeholder profile assessment can be utilized as a tool for requirements producers to organize their elicitation activities, as well as to schedule meetings that work in conjunction with the availability of the stakeholders. It also optimizes the utilization of project resources by decreasing the need to have all stakeholders present for all meetings. For example, in Table 2-4, only Anthony, Lucas, Maria, and Michael might be invited to a discussion of the project requirements related to “reserving procedures.” Furthermore, if I want to decrease involvement of an over-utilized stakeholder, I could leave Michael off the invite list for discussions on “reserving procedures.”

Tip 2-4: Create a reusable repository of profiles

I recommend compiling stakeholder profiles for every business area. A centralized repository of profiles is reusable and will save time at the start of each project. Management might also use the profiles to cross train staff. There is some maintenance, however, to keep the profiles accurate and up to date (at least every six months). If they become outdated, then people won’t trust them and will discontinue applying the stakeholder profile technique to get the right stakeholders involved. Make sure the date on which the profiles were created (and updated) is recorded.



2.3.6 Step 6: Secure Stakeholder Resources

Secure the needed stakeholders by meeting with the project manager, project sponsor, and/or a stakeholder's supervisor. Confirm the availability of stakeholders for the project timeline.

It is conceivable for no one to have a great deal of expertise on some topics. Unavailable or nonexistent resources may pose a significant project risk. This was certainly the case on one project I worked on—for a system intended to include brand-new technology, and no internal stakeholders had experience with the new technology.

Tip 2-5: Secure management support

You may need to educate the sponsor of the project on the benefits of applying the stakeholder profile technique. At the start of the project, sponsors are sometimes guilty of telling you what resources to which you can have access. Quite often when the resource to which you're directed does not have sufficient subject matter expertise to supply requirements, the resource becomes a middle man and must go to the real expert with questions. It is this true expert with whom you want to be talking directly.

Rather than telling you what resources to use, sponsors should encourage you to complete the stakeholder profile. The sponsor should work with you to identify the best approach to filling the resource needs based on the specific topics within the project scope.

One of the benefits of the stakeholder profile is the ability to utilize resources more effectively. Not all stakeholders need to be invited to all discussions. The sponsor will find it more challenging to ignore requests for specific resources when presented with a tangible model that demonstrates the lack of expertise of "assigned" resources.



2.4 CHAPTER SUMMARY

- ◆ Stakeholders (such as requirements suppliers and receivers) are the most common source for eliciting requirements. Creating an institutionalized list of stakeholder roles provides consistency across projects and provides requirements producers with a reusable, time-saving checklist.
- ◆ User involvement is a critical project success factor. The stakeholder profile technique is proven to help get the right people engaged so that fewer requirements are missed.



2.5 SUGGESTED READING

A Requirements Pattern: Succeeding in the Internet Economy, by Patricia Ferdinandi, [Ferdinandi, 2002].

This book does a good job of describing **roles and responsibilities** of the development team in Chapter 8.

Exploring Requirements: Quality Before Design, by Donald Gause and Gerald Weinberg, [Gause, 1989].

Chapter 7 explores “**getting the right people involved**.”

Requirements Engineering, 2nd Edition, by Elizabeth Hull, Ken Jackson, and Jeremy Dick, [Hull, 2005].

Chapter 5 of this book includes a discussion on Requirements Engineering in the Problem Domain including the **identification of stakeholders**.

Requirements Engineering: A Good Practice Guide, by Ian Sommerville and Pete Sawyer, [Sommerville, 1977]. This book devotes Chapter 13 to **viewpoints**, which is a term that is broadly synonymous with a perspective on a system. Similar to stakeholder profiling, viewpoints are designed to reduce the risk of overlooking stakeholders and their requirements. They may also help to manage and analyze the requirements once they have been elicited.

Software Engineering, Edited by Merlin Dorfman and Richard Thayer, Institute of Electrical and Electronic Engineers (IEEE), [Dorfman, 2000]. This is an impressive compilation of software engineering papers that includes a list of software engineering **standards sources**.

Software Requirements, 2nd Edition, by Karl Wiegers, [Wiegers, 2003]. Chapter 6 describes challenges in finding the **voice of the customer**, such as identifying different classes of users and finding the sources of user requirements.

Writing Better Requirements, by Ian F. Alexander and Richard Stevens, [Alexander, 2002]. Chapter 2 provides a practical approach to **identifying stakeholders**.



PREPARATION
is the link
between
ORDINARY people
and
EXTRAORDINARY
interviews.



3 | INTERVIEWING TIPS AND TOOLS

IN THIS CHAPTER:

3.1 Preparing for the Interview	55
3.2 Asking the Right Questions	66
3.3 Conducting a STAR Interview	74
3.4 Chapter Summary	86
3.5 Suggested Reading	87

Interviewing is much more effective if it uses a structured technique or process; it is not merely asking questions. Interviewing requires the development of basic social skills, the ability to listen, and knowledge of a variety of interviewing tactics. This chapter presents suggestions in three areas of the interview techniques for gathering requirements.

3.1 PREPARING FOR THE INTERVIEW

Aside from conducting the interview itself, the real success lies in the preparations done prior to the interview. If I were asked to identify the single most important tip for a successful interview, I'd say, "never walk into an interview empty-handed." I mean that figuratively and literally! Read the following six guidelines on preparing for the interview to find out why.



3.1.1 Guideline 1: Schedule the Interview

Assuming you've used the stakeholder profile technique presented in Chapter 2, "Involve the Right Stakeholders," the next step is to schedule the interview with each identified stakeholder. For the context of this chapter, I'll use the term interviewee instead of stakeholder. Additionally, the content of this chapter is presented with the intent that you are interviewing one interviewee at a time.

The goal of the interview is *quality* requirements as well as quantity. For example, responses to your prepared list of questions might give you the level of detail you are seeking, while other responses prompt you to ask additional probing or follow-up questions. You want to make the most of the allotted time with the interviewee, and gather as much useful information from them as you can. Schedule the date, time, and place of the interview that is the most convenient for the interviewee.

Where you conduct the interview—in a neutral setting or in the interviewee's normal work environment—depends on the interviewee and the purpose or level of desired detail. A neutral setting helps to reduce outside distractions such as the interviewee's phone, computer, other audio disruptions, and perhaps other co-workers that might interrupt. Neutral settings help when the nature of your questions involve getting the interviewee's opinions or vision.

Conducting the interview in the interviewee's normal work environment can offer the benefit of having colleagues nearby to provide additional information. The interviewee may want to share examples from an existing system, or access reference manuals, documents, forms, and other tools used as work aids. It usually is very helpful for you to experience an interviewee's work setting.

Tip 3-1: Deal diplomatically with cell phones

Here's a way to politely ask interviewees to turn off their cell phones. You might say, "We can really optimize the limited time we have by shutting out distractions. Would you mind turning off your cell phone while we talk?"



Select a convenient date and time for the interview. Take into consideration the interviewee's normal working hours. Does the person arrive early or late? Does the person leave early or late? Does the person work flexible hours in which he or she is unavailable on certain days of the week or leaves earlier on a particular day of the week? What is the person's usual lunch period? Are there any times of the day when the person is particularly busy? Remember, the goal is to select a time and place that maximizes the focus or attention of interviewees. As a general rule, do not schedule an interview:

- ◆ **Between two other appointments on an interviewee's calendar.** You want to be certain you have adequate time to complete the interview.
- ◆ **During the first hour or the last hour of the work day.** Some people are either too distracted or too tired. Of course, for other people, these are the most effective times to catch them. Be alert and don't over-generalize.
- ◆ **The day prior to or the day following a vacation or holiday.**

It's also true that you might extract different useful information from the same person at different times, or in different environments. If you talk to interviewees when they're relaxing at the end of the day, they might be a lot more forthcoming about ideas, or much more prepared to speculate. They just might tell you stories or embarrassing information that would otherwise remain hidden! Just like an experienced private investigator, there might be things you can only find out after interviewees think the interview is over, and you've put your pen and paper away.

3.1.2 Guideline 2: Prepare an Agenda

The interview is really just a form of a meeting. You want to follow good meeting etiquette. Having an agenda for the interview is essential to keeping the interview on track. The agenda sets the tone for specifically what will and will not be discussed as appropriate. Depending on the interviewee and the anticipated level of detail you are seeking, allow time to ask probing and follow-up questions to your prepared questions. At times, deviating from the agenda can also be



Tip 3-2: Use your agenda as a visual aid

The agenda is your visual aid (tool) for keeping the interview on track. Most adults communicate more effectively in a visual mode. You can take advantage of the visual nature of the interview using a printed copy of the agenda. Although you send the agenda prior to the interview, I recommend that you print a hardcopy and take it to the interview. Make a comment such as, "I took the liberty of printing a copy of the agenda for you," as you hand the interviewee the agenda. Now that the agenda is in front of the interviewee, you can use it as needed to visually capture the interviewee's attention.

If the interviewee is getting a bit off track, you can literally point to the next discussion topic or question on the agenda as you say, "Thank you, [interviewee name], let's go on to the next question." The closer you move in, and the longer you point to the item, the more you draw in the interviewee's attention. As soon as you have the interviewee focused, move the interview in the direction you want to go. You're back in control.

Of course, the effectiveness of this tip is dependent on the time you spend preparing your questions and your agenda. And, with some interviewees, the interview might go better if they are unaware of the agenda.

effective, depending on what you learn as you go along. Essentially, be flexible. Here is a summary of what to include in the agenda:

- ◆ **PURPOSE.** Clearly state the purpose of the interview and the specific topics of discussion. You may even consider providing an example question.
- ◆ **SCHEDULE CONFIRMATION.** Communicate the interview logistics—date, time, and place.



- ◆ **PARTICIPANTS.** Identify all participants including a note taker if one is used (highly recommended as it typically increases the quantity of information gathered; however, I understand that resources are often unavailable). See Section 3.1.4 for more information on this.
- ◆ **FORMAT.** Explain how the interview will be conducted. This is helpful when it is necessary to educate the interviewee on the interview process. What topics will be discussed? Are you allotting a specific length of time to cover each topic? Are you asking the interviewee a list of questions he or she was given at the start of the interview? Have you sent the interviewee a list of questions prior to the interview that you'll be reviewing? Are you going to ask prepared questions that only you have a copy of?
Additionally, you might provide an opportunity for the interviewee to add topics you did not include but that directly relate to the subject for which you are gathering information. There may be hidden knowledge or processes of which you and others are not aware.
- ◆ **PREPARATION ACTION ITEMS.** Communicate to the interviewee any expectations about materials that he or she is to review prior to the interview, or work products that should be brought along to the interview. Also let the interviewee know if there are work products or documents that you will be bringing to the interview. Action items should specify who is responsible, what is to be done, and by when.
- ◆ **FOLLOW-UP ACTIONS.** Explain what the interviewee can expect after the interview. Note, this is a great opportunity to educate the interviewee on the requirements-gathering process you're using (refer to the requirements process overview presented in Chapter 1, "Requirements in Context"). The actions of who, what, and when should be agreed upon at the close of the interview.

A suggested format of an interview agenda is presented in Tool 3-1 on the next page.



PART ONE: RIGHT PROCESS, RIGHT PEOPLE, RIGHT TOOLS

Tool 3-1 Suggested Interview Agenda Format

This simple agenda format might be used to communicate the logistics of the interview.

Purpose:

Date: _____

Time: _____ to _____

Duration: _____

Place: Building _____

Room _____

Participants:

Topics (or questions) for discussion:

Preparation action items:

Post-interview action items:

Attachments:

3.1.3 Guideline 3: Distribute the Interview Materials

As a general rule of thumb, it is better to start with a draft of something to review than to start with a blank canvas, provided you do not unnecessarily constrain the interviewee or interject your own bias. Depending on where you are in the requirements-gathering process, or what iteration or phase in the project, there usually is something to review. If you're at the very beginning, there generally is something such as an e-mail (electronic mail), a problem report, or an enhancement request that triggered the project. If you use a project development life cycle methodology, there may be project deliverables such as a project overview, project definition, or



Tip 3-3: Provide questions prior to the interview if appropriate

"To send or not to send?" That is the question you must answer regarding whether or not you provide the interviewee with a list of questions prior to the interview.

Sending a list of prepared questions to interviewees in advance allows the interviewee adequate time to research and prepare. This is very beneficial when you are seeking a lot of factual information and actual statistical or historical data. You should validate (through the stakeholder profiling technique presented in Chapter 2, "Involve the Right Stakeholder") that you have the right subject matter expert(s) so you feel confident that you have accurate facts. Be aware of interviewees who supply you with the responses to your questions and then declare that the interview is no longer necessary.

Conversely, providing the list of questions to the interviewee at the start of the meeting, or not at all, also has advantages. The person will not have had time to ponder the questions in advance so responses are candid and not rehearsed. This is a good approach when eliciting opinions and estimates, and works well when asking for the interviewee's vision.

business case. At a minimum, the interview agenda should be distributed prior to the interview. Again, use good meeting etiquette, and send all materials out in a timeframe that is adequate relative to the amount of material the interviewee is expected to review.

3.1.4 Guideline 4: Recruit a Note-taker

A lot of us like to think that we are able to multi-task. However, the interview is not the time to do so. It is a talented individual indeed who is able to "actively" listen and record notes at the same time. Recruiting someone who can take notes makes it easier for you to actively listen. It also increases the quantity of requirements you are able to elicit, as you don't have to slow down



the interview process by note-taking. I am not suggesting that you take no notes at all; in fact, I find it necessary to jot down key words or phrases in order to help me remember and paraphrase what I just heard. Jotting down a few key words can also help you to remember to go back and ask additional probing questions.

I know that it is difficult to find an available (or even willing) note-taker. I suggest pairing up with a buddy to take notes. You might take notes for the buddy's interviews and he or she will take notes for yours. Whenever possible, try to use the same note-taker. This person will get used to your interview style and approach, and will also learn your preferences about the level of detail for notes. Make sure it's someone with at least a modicum of understanding of what's going on. Otherwise the note-taker may miss important points, or write something down incorrectly.

You might consider tape recording or videotaping the session when a note-taker is not an option or available. However, most interviewees don't like being recorded.

A note-taker is particularly recommended when you are conducting an interview with someone whose time is considered extremely valuable (or expensive), whose time is not easily granted, and with whom you are unlikely to get an opportunity for a follow-up interview.

Tip 3-4: Ask permission to bring a note-taker

When interviewing only one person, ask permission to bring a note-taker or to record the interview before the agenda is delivered. Do not ask at the start of the interview and risk offending the interviewee. Bringing someone into the room unexpectedly may make the interviewee feel uneasy. Avoid giving the "two-against-one" impression—the interrogation.



3.1.5 Guideline 5: Prepare a List of Questions

Yes, I said prepare. Say it with me, “Prepare!” In addition to the agenda, a list of prepared questions is essential to keeping the interview on track. Having a prepared list of questions does not mean you have to literally read off each question—although in some cases, reading the questions exactly as they are written is very helpful. Some questions are intended to be follow-up questions based on the interviewee’s response. For example, based on the interviewee’s response, you might skip a question or two, or skip an entire section of questions. Or you might add extra follow-up questions on the spur of the moment. Consider the following suggestions in preparing your list of questions:

- ◆ **Arrange the list of questions in a logical order**, and grouped by relevant issue. For example, group your questions by the type of requirement (user, functional, and nonfunctional) you are eliciting.
- ◆ **Determine how much time to spend** on each issue, topic, or logical group of questions.

Tip 3-5: Apply a business requirement statement format

For interviews at the business requirements level, write the following statement format on a white board and ask the interviewee to help fill in the blanks:

The **PURPOSE** of the _____ [*insert project name*]

IS TO _____
[*describe what the project team is expected to implement or deliver*]

SO THAT _____
[*describe measurable business benefits*]



- ◆ **Understand the types of questions used and the expected answers.** You should give some thought to what answers you might get in response to your questions. This will help you know whether or not the interviewee actually answered your question. It will also help you know the direction you'll take in asking additional probing questions.
- ◆ **Use the questions to maintain the direction of the interview.** Just as there is a logical hierarchy or relationship among the requirement types (refer to Chapter 1, “Requirements in Context”), there is a logical progression to the interview questions. You would not, for example, want to start your interview by asking questions to elicit functional requirements (system level) if you have not already identified the user requirements (user level).

3.1.6 Guideline 6: Identify the Success Criteria

How will you know whether or not the interview was a success? Was the purpose to identify top-level business requirements? If so, did you capture all of the specific measurable business benefits? From the interview notes, are you able to produce a deliverable that represents the business requirements? Was the purpose of the interview to validate all of the users and focus on gathering user requirements? Using your agenda, were you able to keep the interview on track? Did you successfully discuss all the topics identified in your agenda? Did you run out of time? Did you schedule too much time? If you achieve everything before the end of the allotted time, close the interview. Don’t pad it out unnecessarily.

The checklist presented in Tool 3-2 will help you to organize and plan your session. The success criteria for your interview will be based on the level of the requirements, which equates to the level of detail that you seek. Refer back to Chapter 1, “Requirements in Context,” for an explanation of requirement levels.



Tool 3-2 Interview Preparations Checklist

- 1. Interviewee is contacted and request for interview granted.
- 2. Convenient date/time/place is confirmed with interviewee.
 - Date: _____
 - Time: _____
 - Place: _____
- 3. Place/location reservations are confirmed.
 - Confirmation number: _____
 - Conference call information: _____
 - Building/room number: _____
 - Address: _____
 - Create a map and instructions to the site if the location is offsite or unfamiliar to the interviewee.
- 4. Necessary equipment is accessible or reservations confirmed.
 - Video camera, screen, projector
 - Audio recording device
 - Conference telephone
 - White board, markers, eraser
 - Flip chart, markers, easel
 - Note pads, writing instruments
 - Personal computer or laptop
 - Internet connectivity
 - Power source (electrical, battery, other)
- 5. Facility environment is verified.
 - Lighting
 - Temperature (climate) controls
 - Table, chairs, other
 - Food and beverages
- 6. Agenda is prepared.
- 7. Pre-interview review materials are identified.
- 8. Agenda and pre-interview review materials are distributed to interviewee.
- 9. If a note-taker is involved:
 - Prior permission granted by interviewee to have a note-taker or recording device present.
 - Note-taker is briefed on interview process or format, as well as any other special instructions.
- 10. Interview list of questions is prepared.
- 11. The interview success criteria are



3.2 ASKING THE RIGHT QUESTIONS

Asking the right question is as much how you ask it, as it is what you ask. Altering the way a question is phrased can make all the difference in the response it invokes. For example, if I ask you, “Are your lights on?”, your response will most likely be a simple “yes” or “no.” However, if I alter the question by adding just a single word and ask, “Why are your lights on?”, I should expect you to respond with one or more reasons for your lights being on. What questions you ask will depend on the following two key components: why you ask, and what you expect. Both should be given considerable attention when deciding what types of questions you ask.

3.2.1 Why Do You Ask?

You must know why you are asking each question. In other words, you must know the purpose or goal for asking each question. Are you seeking facts? Are you seeking an opinion from the interviewee? Are you intending to change the direction of the discussion? Do you need to get the interviewee to elaborate on a particular topic? Are you asking the interviewee to explain something? Your answers to these questions will change the types of questions you will use in the interview. I’ll give you examples of types of questions to ask depending on what your purpose or goal is in Section 3.2.3.

Tip 3-6: Ask “why?”

Asking “why” is a powerful way of discovering requirements. The question “why?” used by itself might seem threatening so be careful not to overuse it. “Why” questions can be stated in an unaggressive manner such as:

- ◆ “What is the underlying reason for that?”
- ◆ “What is the rationale behind that?”
- ◆ “Would you explain why you need it to do that?”
- ◆ “What is the purpose of doing that?”
- ◆ “Help me to understand why it is done that way.”
- ◆ “How does that happen? Can you explain why it matters?”



3.2.2 What Do You Expect?

Knowing what you expect relates to the expected response from the interviewee. Once you've asked your question, what is the expected response? By this, I mean what kind of response are you expecting to hear? Giving some thought to what you expect the interviewee to respond with will help you with the following:

- (1) **Knowing whether or not the interviewee answered your question.** If the question was not answered, it will be necessary for you to interpret why not. Perhaps the interviewee did not understand the question, in which case you will need to rephrase the question and ask it again. Or, perhaps the interviewee does not want to admit he or she doesn't know the answer, in which case you may want to move on to another topic.
- (2) **Deciding which direction to take regarding follow-up questions or your next line of questioning.** For example, say you have already elicited the business, user, and some functional requirements, and you are now interviewing for more detail on the functional and nonfunctional requirements. You may ask a question related to interoperability (an indication of how easily the system can exchange data or services with other systems) such as, "What other systems or processes does the current system need to exchange data or services with?" If the interviewee's response is "none," then you will want to skip your prepared questions related to interoperability. However, if the response indicates that there are other systems to

Tip 3-7: Deal tactfully with "information hoarders"

Some interviewees are very protective of the knowledge they hold and do not want you to interview someone else. Other interviewees might not be the subject matter expert and are too embarrassed or shy to acknowledge this. A question such as, "Is there anyone else I should talk to about this?" might not be well received. To avoid offending someone, the following question generally works quite well, "In the event you're out of the office for any reason, who is your backup?"



interface with, then you will want to proceed with your prepared questions related to interoperability.

- (3) **Assessing the appropriate level of detail for the particular purpose of the interview.** For example, you may ask the project sponsor, “Why is this project being done?” This question is typically asked early in the requirements-gathering process in order to identify the business requirements. The sponsor might respond by telling you that three new management reports are needed, and proceeds to give you some details about these reports. While this information or level of detail is important, it is probably too detailed for the intent of the current interview. Before describing the information desired on a report, you should identify which users are interacting with the business. You will want to better understand the work the users perform. Once you understand the purpose of the work, you can then fill in the details about the information needed.

3.2.3 The Right Types of Questions

CONTEXT-FREE QUESTIONS are high-level questions posed early in the project to obtain information about the project goals, objectives, and scope. These questions are most useful in identifying the business requirements in an interview with the project sponsor. Context-free questions are appropriate for any project regardless of business or process, and are independent of any specific design or technology. Table 3-1 contains a sampling of context-free questions based on Gause and Weinberg [Gause, 1989].

OPEN-ENDED QUESTIONS are questions that allow the interviewee to answer freely. The interviewee will provide facts, give an opinion, or explain something. “Is dinner ready?” is an example of a closed-ended question, typically characterized by a “yes” or “no” response. On second thought, perhaps this question isn’t such a good example! I’ve heard of situations when this question invites very colorful elaboration! In other words, don’t try this one at home.

Open-ended questions are good for a tester to consider when determining if a requirement or condition passed or failed. However, closed-ended questions are generally not as useful in



Table 3-1 Example Context-Free Questions

- | | |
|----|---|
| C1 | Who is the customer? (Who is paying?) |
| C2 | What problems would this product or system solve? For whom? |
| C3 | What related problems are not being addressed by this project? Why? |
| C4 | What problems could this product or system create? |
| C5 | Who else do you think should be involved in this project and why? |
| C6 | Why is this project visible to upper management? Who is upper management? |
| C7 | What is the business reason for wanting to solve this problem? |
| C8 | What obstacles or roadblocks are there to the success of this project? |
| C9 | How long has this concept or product been considered? What were its previous innovations? |

Tip 3-8: Convert to open-ended questions

As a rule of thumb, you can turn most close-ended questions into open-ended ones by adding the word “what” or “why” to the beginning of the question.

eliciting details. And, of course, a question that is too open could cause the interviewee to go off topic.

If you ask a closed-ended question, you usually end up asking a follow-up question in order to get at the information you were seeking initially. If the purpose of the interview is to maximize the quantity of information gathered in the allotted time, the interview will be more effective if the right questions are asked to begin with. This allows more time to ask additional worthy questions, rather than follow-up questions that are attempting to get at the very same information.

For example, let’s say you ask a business manager, “Do you have any employees?” Because it was a closed-ended question, the business manager responds simply by saying,



“Yes.” Then you have to ask a follow-up question such as, “What is the maximum number of employees who will need security access?” Now, this simple dialog exchange might not seem like a big deal, but expand this type of dialog over a one-hour interview, and then see how much time is wasted. You need to make your questions open-ended in order to get right to the point.

DIRECT QUESTIONS are used to obtain factual and objective responses. Generally speaking, direct questions are led by one of the following: *who, what, where, when, why* or *how*. Here are some examples of direct questions that result in factual responses—although be aware that the responses might not be complete or even accurate.

- ◆ Who uses this billing information?
- ◆ What information is needed for that task?
- ◆ Where is the report being distributed?
- ◆ When does the user need the information?
- ◆ Why is it done in that sequence?
- ◆ How many regions are there currently?

ELABORATION QUESTIONS or phrases are used to elicit subjective responses or opinions. Subjective responses and opinions are helpful in gaining insight to potential alternative solutions to the current problem—though it is not your responsibility to pursue solutions. Often the users of a system have ideas about how to improve the system. The following phrases and questions are used to prompt the interviewee to elaborate on the topic or to give his or her opinion:

- ◆ Tell me more about that.
- ◆ Explain what you mean by that.
- ◆ Describe what happens next.
- ◆ What do you think about this?



- ◆ How do you feel about it?
- ◆ Where could the workflow be improved?
- ◆ What concerns do you have?

Tip 3-9: Don't "lead the witness"

Avoid questions that provide or anticipate the response. For example, instead of asking, "Should it be done daily or weekly?" ask, "How often should it be done?"

You might want to first demonstrate some level of understanding on the subject. Don't hide everything you know, in case it makes you look inexperienced!

LADDERING QUESTIONS are used to raise or lower the level of information gathered. They usually begin with "how," "what," or "why," and are used to prompt the interviewee to respond with a different level of detail. For example, ask "why" questions to raise the level of the discussion (move up the ladder). If the interviewee is giving too much detail about how a process is done, ask, "Why is the user (role) doing this?"

On the other hand, use "how" questions or elaboration questions, as described in the previous section, to lower the level of detail (move down the ladder). If the interviewee is not giving enough detail about the work that is done, ask, "How does the user do that?"

Laddering questions are particularly useful when the interviewee has a tendency to talk about how to design the system, instead of giving the requirements for the system. Let's examine the following interview dialogue to see how laddering questions can help raise the level of the conversation.

Interviewer: "What does the clerk do next in the order process?"

Interviewee: "The clerk clicks on the check order button on the New Order Screen."

Interviewer: "Why does the clerk click the check order button?"



Tip 3-10: Differentiate fact from opinion

Be aware of the difference between facts and opinions. Whenever possible, facts should be validated by another source to confirm the accuracy. With opinions, you should investigate whether other people hold different opinions. Opinions are liable to vary more than perceptions or facts do.

Unfortunately, knowing the difference between facts and opinions isn't easy. You might want to ask a follow-up question such as, "Is that your opinion or a fact? If it is a fact, do you have something to substantiate it?"

Interviewee: "She does this to find out if all the items ordered are in stock."

Interviewer: "Why does the clerk need to know if all the items ordered are in stock?"

Interviewee: "Because if any items ordered are not in stock, the clerk has to submit a backorder request along with the order."

Interviewer: "So the next step in the order process is to fill out a backorder request for each item that is not in stock. Is that right?"

Interviewee: "Yes, that's right."

In the above example, the interviewer used the "why" questions to raise the level of detail. Clicking the button on the screen is a step in "how" (design) the clerk performs the next step in the order process. The actual next step in the process is to complete the backorder request.

Before getting into the details of the process or the information needed, it is important to understand the purpose for doing the process. You must first clearly identify the user goals that the system is supporting. That is, what business functions do the users need to perform?



Tip 3-11: Don't confuse "how" with design

Don't confuse "why" and "how" questions with the "whys and whats" of requirements and the "hows" of design.

Furthermore, the interview is not an "interrogation." Be careful not to offend interviewees or make them feel that you are questioning their skills or authority.

METAQUESTIONS are questions about questions. Metaquestions are helpful in assessing the progression of the interview. These questions are intended to be a litmus test of the interview process itself and are typically closed-ended type questions. Table 3-2 lists some examples of metaquestions.

Table 3-2 Example Metaquestions

-
- M1** Am I asking too many questions?
 - M2** Do my questions seem relevant?
 - M3** Are you the right person to answer these questions? Are your answers definitive?
 - M4** To be sure that we understand each other, I've found it helps me to have things in writing so I can analyze them later. May I write down your responses? Is it OK for a note-taker to capture our discussion?
 - M5** (*Use this if your communication thus far has only been in writing.*) The written material has been helpful, but I find that I understand some things better if I can discuss them face to face. Can we get together at some point to discuss the material further?
 - M6** Is there anything else I should be asking you?
 - M7** Are there important topics I haven't mentioned?
 - M8** Is there anything you would like to ask me?
 - M9** May I return or call you with more questions later, in case I don't cover everything in our allotted time?



Tool 3-3 Suggested Questions (Part Two: “Right Approach, Right Questions”)

Over 2,000 suggested questions for eliciting nonfunctional requirements are included in Part Two, “Right Approach, Right Questions.” These questions can help you prepare for your requirements-gathering interview and feel confident that you’re asking the right questions. After reading Chapter 2, “Involve the Right Stakeholders,” and a good part of Chapter 3, “Interviewing Tips and Tools,” you might realize that there’s a little bit of work involved in developing “good” requirements.

3.3 CONDUCTING A STAR INTERVIEW

Interviewing can be a very exhausting activity. Interviewing is mentally and physically draining due to the level of concentration and focus that is required by the interviewer in order to get the most out of the interview, both in terms of time allowed and the quality and quantity of information elicited.

This section provides guidance for conducting your interview, but not just any interview—a **STAR** interview:

- ◆ **S**tart with style.
- ◆ **T**QLR (apply the active listening strategy: Tune in, Question, Listen, and Review).
- ◆ **A**sk questions.
- ◆ **R**ecap and wrap-up.

3.3.1 **STAR—Start with Style**

Starting the interview with style means following the Golden Rule. That is, treat others as you would want to be treated. The interview should be conducted in a professional manner, and with the utmost courtesy toward the interviewee. It is important to show respect for the interviewee’s



expertise or area of authority, and to express genuine appreciation for the time the person invested in the interview. In the following paragraphs, I give you a structure to follow when opening the interview, as well as some general guidelines for conducting the interview with style.

3.3.1.1 OPEN THE INTERVIEW

- (1) **Start with introductions.** The introduction step is intended to establish good rapport with the interviewee. Begin the interview by thanking the interviewee for his or her time. If you have not met the person face to face before, continue the interview with proper introductions. Also, don't forget to introduce the note-taker and briefly explain that role.
- (2) **Review the purpose and goals.** This step is intended to set the ground rules or review the expectations for the interview. Use the interview agenda to discuss the interview purpose and goals with the interviewee.
 - ◆ **Explain the topics that will be discussed** and, as appropriate, which topics will not be covered. Explain how the information elicited during the interview will be used.
 - ◆ **Confirm that the interviewee is still available** for the scheduled duration of the interview. This is particularly important when interviewing executive management or individuals who have heavy demands on their time. You might ask, "This interview is scheduled for 45 minutes. Is this workable for you?" The note-taker might be the timekeeper and makes sure the interview keeps within the allotted time.
 - ◆ **Review the format of the interview** as needed. Briefly describe how you are going to conduct the interview. This may also include a time allocation per issue. To explain the format, you might say



something like, “I’m going to use a list of questions that I’ve prepared beforehand to guide our discussion. Then we’ll recap the discussion, and identify the next steps or any follow-up action items. Is this an agreeable format to you?”

- ◆ **Follow up on any materials that were distributed** prior to the interview. You want to confirm that the interviewee has adequately reviewed all the materials. If the materials are important to the interview and the interviewee has not read the materials, it may be necessary to stop and reschedule the interview.

3.3.1.2 INTERVIEW GUIDELINES

- (1) **Take notes or record the interview.** Even if you have a note-taker present, most interviewers find they need to take a few notes of their own. If the note-taker focuses on capturing, as closely as possible, what the interviewee stated, then your notes may just be phrases to help you keep the interview moving, or one or two words to jog your memory about a follow-up question to the interviewee’s current response. Try to avoid writing word for word what the interviewee is saying. A good listener should be intent on listening, and not writing feverishly while the person is talking.

Tip 3-12: Explain your interview style

Assuming you do not have a note-taker and are not recording the interview, you might explain your style, “I’m not very good at multi-tasking; that is, I’m not listening effectively if I’m writing. What you have to share with me is important so I’m letting you know that I’ll ask a question or two, listen to your response, and then I’ll need to pause to take notes.”

This is a positive way to say that you prefer to devote your whole attention to what the interviewee is saying. It makes the interviewee feel more important!



(2) **Stay on track.** Your primary goal in the interview is to maximize the quality and quantity of information in the allotted time. Use your agenda as a visual aid, as well as your prepared list of questions to maintain control of the discussion. This can be quite a balancing act as you ask “elaboration” type questions, and allow the interviewee to answer fully. If you have an interviewee who tends to stray from the questions or “builds a clock” when asked for the time, you’ll want to remain courteous and cajole the interviewee back on track.

Here are some example phrases or questions you might use to keep the interview moving. Notice that these are “close-ended” questions, as they are intended to be a checkpoint in the discussion, and won’t encourage the interviewee to continue to elaborate.

- ◆ “In order for us to keep within our allotted time, may I move on to the next question?”
- ◆ “Is this a topic that is more important than the others, or should we continue to the next topic?”
- ◆ “Did we capture the key points on this topic? May we go forward?”
- ◆ “We’ve spent about 10 minutes on this topic. Should we continue discussing this topic or continue to the next topic?”

Tip 3-13: Make phone interviews “visual”

The face-to-face interview is always preferred. You cannot maintain eye contact or observe body language while conducting a telephone interview.

However, when face-to-face interviews are not feasible, sharp listening skills and your list of prepared questions will still enable you to conduct a highly effective interview. Keep the interviewee visually engaged by referencing the agenda, “Looking at our agenda, let’s move on to Question 3.”



- ◆ “May we go to the next question, and come back to this topic if we have time at the end?”
- (3) **Pay attention.** I recommend using the following guidelines for maintaining good rapport throughout the interview, and to monitor the interview process.
- ◆ **Use the person’s name.** People want to be recognized, and there’s no better way to do it than to use their names. And, occasionally, in between questions, you can thank the interviewee for his or her answers. For example, “Thank you, James, my next question is...”
 - ◆ **Maintain good eye contact** with the interviewee. This increases active listening, which we’ll discuss in more detail in the next section.
 - ◆ **Use one-word responses or short phrases** to indicate that you are listening. You might say something like, “Yes,” “I see,” “Go on,” “Okay,” or “Great.” Take care not to overdo it, especially if you think the interviewee might begin to feel this is a deliberate tactic on your part. You don’t want to become repetitive or annoying.
 - ◆ **Observe the interviewee’s body language.** Look for signs of fatigue, distraction, indifference, irritation, agitation, or discomfort. Look for any kind of unhappiness, really. Also use the metaquestion approach to help gauge how the interview is proceeding. End the interview and reschedule, if necessary.
 - ◆ **Respect the interviewee’s personal space.** Ideally you want to sit approximately four feet or less from the interviewee. Sitting across a table is good for note taking; just be sure that the table doesn’t put too much space between you. Personally, I prefer sitting “a corner” away: if the interviewee sits on one side, I’ll tend to sit at one end. Then the table doesn’t appear to be a barrier. We’re not on opposing sides!



- ◆ **Maintain good posture.** Be aware of your own body language. You should sit facing your interviewee. Your arms and legs should be uncrossed to give an “open” and receptive expression. A simple gesture, such as a nod, can also indicate to the interviewee that you are listening. Sitting in your seat with a straight posture or a slight lean forward also indicates you are interested in what the interviewee is saying.

Tip 3-14: Maintain an active listening posture

To open your body language, try sitting forward on the chair seat so that your back is not against the seat back. Adjust the seat height so that your hips are slightly lower than your knees. Sit with both feet flat on the floor. This will help you maintain your balance and your posture. Your upper torso will naturally lean forward to also maintain your balance. This open body language gives interviewees the impression that you are truly interested in what they have to say, which, of course, you are. Right? I repeat, right?

3.3.2 STAR—TQLR, An Active Listening Strategy

One of the biggest mistakes you can make as an interviewer is to “reload,” that is, formulate the next question or response in your mind, instead of truly listening to what the interviewee is saying. I recommend using the **TQLR**—Tune in, Question, Listen, and Review—strategy for active listening. The **TQLR** strategy has been around for many years, but its origin is uncertain. (Perhaps no one was taking adequate notes in the meeting at which it was devised!)

- (1) **TUNE IN.** This means getting ready to listen, and giving the interviewee your full attention. During your preparations for the interview, you selected a time and place that minimized distractions for both the interviewee and yourself. You must also get mentally psyched up to listen to your interviewee—but don’t take deep breaths as if warming up for a boxing match.



(2) **QUESTION.** Ask yourself questions while you are listening. This is not you thinking about what your next question will be because you have already prepared your list of questions. Rather, your job as a good listener is to identify the main ideas or points that the interviewee is making. In other words, question your comprehension of what is being presented to you. Here are some good questions to ask yourself while you are listening:

- ◆ What is the purpose of what I am listening to? (To give me directions? To give me information? To give me examples?)
- ◆ What is new in what I am hearing? Is it important? Do I need to remember it? Do I need to ask follow-up questions to get more information?

(3) **LISTEN.** Listen to get answers to your questions. Listen for clues or phrases that help you predict what is coming. Here are some example phrases:

- ◆ “There are three reasons why ...” (Expect to hear three items).
- ◆ “First ... Second ... Third ...” (There they are).
- ◆ “And most important ...” (Here comes a main point).
- ◆ “Also ...” (Here comes something similar).
- ◆ “Remember that ...” (This is probably important).
- ◆ “Et cetera.” Or “And so on.” There is more that the interviewee is not telling you. Be alert and perk up whenever you hear “et cetera” or equivalents, because it might mean the interviewee should have given you an exhaustive list, but didn’t.

(4) **REVIEW.** This is your ability to recall what was said. When you hear something important, repeat it to yourself immediately. Say it in your own words. Write it down in short form so you can relay what you understood. Rephrase your understanding back to the interviewee for confirmation of accuracy. For example:



You ask, “What are the four steps to process a payment?”

“Well,” the interviewee says as he or she begins the response, “the first thing I do is ...”

And, then continues with, “Next I do this ... Then this ... Then this ... Next this ... And finally this ...”

Applying the **Q** (Question) and the **L** (Listen) from the **TQLR** listening strategy, what the interviewee is telling you sounds as if there are more than four steps. You need to confirm what you understood; this is not the same as simply repeating what you heard.

You ask for confirmation by saying, “Let me summarize what I understood to be the four steps to processing a payment. First, you do ...”

“Yes, that is correct,” the interviewee affirms.

Restate each step and wait for the interviewee to affirm the accuracy.

3.3.3 STAR—Ask Questions

Use your list of prepared questions to navigate your interviewee through the interview. Remember that your prepared list is a guide and typically will not be used verbatim during the interview. Based on some of the answers the interviewee gives, you may want to skip some questions on your prepared list, or you may want to ask an additional set of questions. In addition, you will need to ask follow-up questions to probe further, or ask clarifying questions to reduce communication errors.

3.3.3.1 FOLLOW-UP QUESTIONS

Earlier in this chapter, I described various types of questions to ask such as open-ended, elaboration, and laddering. Follow-up questions are the questions you ask as a result of a response to a previously asked question. The intent of the follow-up question is to probe for additional information, or to raise or lower the level of detailed information in the response.



Tip 3-15: Schedule the interview at the right time

Due to the level of concentration that is required to actively listen, don't schedule more than three or four interviews in a normal 8-hour work day. Try not to exceed 60 minutes per interview.

If you have several interviews to conduct in a short period, I recommend staggering the days you interview. For example, schedule three interviews per day on Tuesday and Thursday. Schedule no interviews on Wednesday and Friday so that you have time to review and analyze your notes within 24 hours of the interview. The sooner you review the notes and write down your thoughts regarding any follow-up questions you may need to ask, the better. (Mondays are reserved for last-minute preparations for Tuesday's and Thursday's interviews.)

Apply the **TQLR** strategy described in the previous section, and “listen” for clues or phrases that indicate that you should ask follow-up questions. Here are some good clues to listen for and corresponding follow-up questions to ask:

- ◆ **Response:** “I don’t know.” Or “I’m not sure.”
Follow-up: “Who else should I ask about this? Is this outside your area of responsibility?”
- ◆ **Response:** “Yes” or “No,” or vague answers.
Follow-up: Ask open-ended questions and/or elaboration questions.
- ◆ **Response:** Technology-oriented terminology is used. It may be an indication that the interviewee is telling you “how” to design the product.
Follow-up: “Why does the user need that screen?” “Is this a solution constraint? What is it based on?”
- ◆ **Response:** Ambiguous answers or unfamiliar terminology.
Follow-up: “What do you mean by that? Could you please define that term?” Or



you could ask for a source where specialist terminology is defined—especially if you’re encountering more than a few new terms. You don’t want to exhaust all of your interview time asking for definitions. Furthermore, a written source is likely to give you better formal definitions than an interviewee.

Tip 3-16: Ask follow-up questions

Don’t be afraid to ask follow-up questions or clarifying questions in order to better understand the interviewee. Your responsibility is to elicit accurate and complete information. Believe me, when the wrong product is delivered, the client or project sponsor won’t mind asking questions!

3.3.3.2 CLARIFYING QUESTIONS

You must be sensitive to communication errors, and be listening for errors throughout the interview. Here are some common communication errors to listen for:

- ◆ **Observational.** People perceive things differently.
- ◆ **Recall.** Memory is fallible—and usually self-serving.
- ◆ **Interpretation or ambiguity.** Listen for words such as “small,” “special,” and “normal.” Do you have the same understanding of what the words mean? How will you know unless you ask?
- ◆ **Contradictions.** The information given contradicts with previously provided information, either from the current interviewee or by a previous interviewee.
- ◆ **Facts.** You’ll want to validate any facts that the interviewee gives. Generally, this is accomplished by talking to another subject matter expert about the same topic area or asking the same series of questions. Use the relative importance or significance of the information as a guideline for validating facts. The higher the importance, the more you need to validate the facts as true.



Tip 3-17: Save time—don't type up interview notes

Save yourself the time and hassle of typing up the notes taken during the interview, and sending the notes to the interviewee for confirmation of the accuracy. That is, more often than not, I find that typing up the notes and trying to get the interviewee to read them and respond, to be a waste of time.

My experience has shown that if you do a good job reviewing (refer to the TQLR explanation in Section 3.3.2), then the accuracy of the information gathered is confirmed before you leave the interview. As much as 50 percent of the information gathered in an interview never translates to requirements.

As described in Chapter 1, "Requirements in Context," requirements development is the process of eliciting, analyzing, representing, and validating the information. The interview is an elicitation activity. During requirements analysis the information is combined with other gathered information, and conflicts are identified and resolved. Therefore, the accuracy of the "requirements" derived from an interview should be confirmed during the requirements validation activity.

Moreover, I have seen design and coding begin based on information presented in interview notes (prior to validation and approval). And, you can guess what happened. That's right—rework!

3.3.4 **STAR—Recap and Wrap-up**

The interview generally comes to a close when all the questions have been asked and answered, or when the allotted time has passed. However, that isn't the end just yet. As they say, "It's not over, well, until it's over!" As you bring the interview to a close, it is important to leave the interviewee informed of what will happen next. These wrap-up activities can take as much as ten minutes (or more). Plan for adequate time at the end of the interview, and set a realistic length of time to conduct the interview including the activities to open and close the interview.



I use the following general steps to bring closure to the interview:

Step 1: Summarize the information that was elicited. This helps to confirm to the interviewee that you understood what he or she told you.

Step 2: Discuss and assign any unanswered questions that arose during the interview. The purpose here is not to discuss how to resolve the issues, but to make sure that you've accurately identified the issues. Once you have identified the issues, assign each one for resolution. Agree on the individual who will take responsibility for researching or resolving the issue, and set a target resolution date (be specific and realistic). If necessary, discuss any potential obstacles to meeting the target resolution date.

Step 3: Solicit and answer the interviewee's questions about the interview.

Step 4: Discuss the next steps (allot about three to five minutes for this). This is your chance to educate your business expert, sponsor, or user on the requirements process. The interview is a technique used in the elicitation stage, and is just the beginning of the iterative requirements management process. You'll want to analyze the information gathered in the interview, and integrate the new information with information that was previously received. The information then has to be documented and validated by the stakeholders.

Step 5: Evaluate the interview process. Identify ways in which you may improve it.

Step 6: Thank the interviewee!

Tip 3-18: Say "thank you!"

"Thank you" goes a long way in building rapport with stakeholders. Besides, it just makes people feel better.

On a side note, I usually receive better service in a restaurant by thanking the server. For example, I might say, "Please, when you have a moment, I'd like another napkin, thank you." If you use the server's name, the service usually is even better. Try it!



3.4 CHAPTER SUMMARY

- ◆ Interviewing stakeholders is certainly not the only source for requirements; it just happens to be a commonly used elicitation technique.
- ◆ I know there are people, and you might just know some of them, who do not believe in taking the time to prepare for any sort of interview (even a job interview). I challenge them to compare the time it takes to prepare well for a requirements-gathering interview, as opposed to the time spent fixing problems with the system due to missed or poorly defined requirements. Get my point? As Nike® says, “Just do it.”
- ◆ Interviewing is very much a social skill that requires practice to master. Good listening skills are an essential element to being really effective.



3.5 SUGGESTED READING

Exploring Requirements: Quality Before Design, by Donald Gause and Gerald Weinberg, [Gause, 1989].

Context-free questions are explored in Chapter 6, while “the tried but untrue use of **direct questions**” is described in Chapter 4.

Requirements by Collaboration: Workshops for Defining Needs, by Ellen Gottesdiener, [Gottesdiener, 2002]. This book dives into the **facilitated workshop** elicitation technique. It is a wonderful alternative to the one-on-one interview, as it is known to significantly reduce the requirements-gathering time.

Writing Better Requirements, by Ian F. Alexander and Richard Stevens, [Alexander, 2002]. Chapter 3 describes techniques for gathering requirements from stakeholders such as **interviews**, workshops, experiencing life as a user, observing users at work, acting out what needs to happen and prototypes. Additionally, Chapter 4 identifies **other sources of requirements**.



*It is better to ask a few
well-thought-out questions,
than a *lot* of questions
without thinking.*



PART TWO

PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Part One establishes a foundation of software requirements terminology and context. It also provides step-by-step guidelines for building a stakeholder profile (Chapter 2, “Involve the Right Stakeholders”) and conducting an outstanding requirements-gathering interview (Chapter 3, “Interviewing Tips and Tools”). This foundation sets you up for success in applying what you’re about to learn here in Part Two.

Part Two is comprised of four chapters that define nonfunctional requirements and offer over **2,000 suggested elicitation questions**.

Chapter 4, “Understanding Nonfunctional Requirements,” provides an in-depth definition of nonfunctional requirements, and explains why these critical requirements are so difficult. It provides a user-focused classification for 14 common nonfunctional requirement categories, which are comprised in three groups.

Chapters 5 through 7 provide definitions, example requirements, suggested elicitation questions, and suggested reading for each nonfunctional category shown in Table II-A.



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Table II-A A User-focused Classification of Nonfunctional Requirements

User Needs	User Concerns	Nonfunctional Categories
Chapter 5 Operation Requirements	How well is it guarded against unauthorized access?	Access Security (ACS)
	How dependable is it during normal operating times?	Availability (AVL)
	How fast, how many, and how well does it respond?	Efficiency (EFC)
	How accurate and authentic are the data?	Integrity (INT)
	How immune is the system to failure?	Reliability (REL)
	How resilient is the system from failure?	Survivability (SRV)
	How easy is it to learn and operate the system?	Usability (USE)
Chapter 6 Revision Requirements	How easy is it to modify to work in different environments?	Flexibility (FLX)
	How easy is it to upkeep and repair?	Maintainability (MNT)
	How easy is it to expand or upgrade its capabilities?	Scalability (SCL)
	How easy is it to show it performs its functions?	Verifiability (VER)
Chapter 7 Transition Requirements	How easy is it to interface with another system?	Interoperability (IOP)
	How easy is it to transport?	Portability (POR)
	How easy is it to convert for use in another system?	Reusability (REU)



THE ANATOMY OF A NONFUNCTIONAL REQUIREMENT CATEGORY

This section describes the presentation layout of each nonfunctional category included in chapters 5 through 7. As shown below and in Figure II-B, each nonfunctional category consists of the following components:

- (a) **Nonfunctional Category Name.** Each category is named.
- (b) **User Concern Short Description.** A brief question is used to identify a general concern that users have regarding the software system.
- (c) **Related Categories.** What is the relationship of this particular nonfunctional category to other categories? What other categories might influence this category? If changes are made to this requirement category, what other categories might be affected?
- (d) **Category Definition.** Each nonfunctional category is defined.
- (e) **Category Discussion.** The nature of the nonfunctional category is described along with further information to clarify the category.
- (f) **Category Examples.** Real-world requirement statements are provided to illustrate requirements derived from the requirements-gathering activities.
- (g) **Category Suggested Questions.** A list of suggested questions is divided by a requirement framework (refer to Chapter 1, “Requirements in Context”) or interest areas as follows:
 - ◆ **DATA**—**What** information is used within the system?
 - ◆ **ROLES**—**Who** (user roles) provides and/or receives information in performing these business functions?
 - ◆ **PURPOSE**—**Why** is the information necessary? What is the business rationale or strategy?
 - ◆ **TIMING**—**When** is the information used?



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

- ◆ **LOGISTICS**—**Where** is the information used?
- ◆ **PROCESS**—**How** is the information used?
Is there a process or procedure that drives usage?

Within each requirement interest area, the questions are further subdivided by perspective or view. That is, who are you asking—requirements supplier or requirements receiver?

Additionally, each question is numbered with a unique identifier for ease of reference. Each category is assigned a three-character abbreviation, for example, **EFC** is assigned to Efficiency. Questions intended for requirements suppliers have an **S** following the question ID, for example **EFC28s**. Questions intended for requirements receivers have an **R** following the question ID, for example **EFC35R**. If a question might be used for requirements suppliers and receivers, the question ID is followed with a **B** (both), for example **EFC37B**.

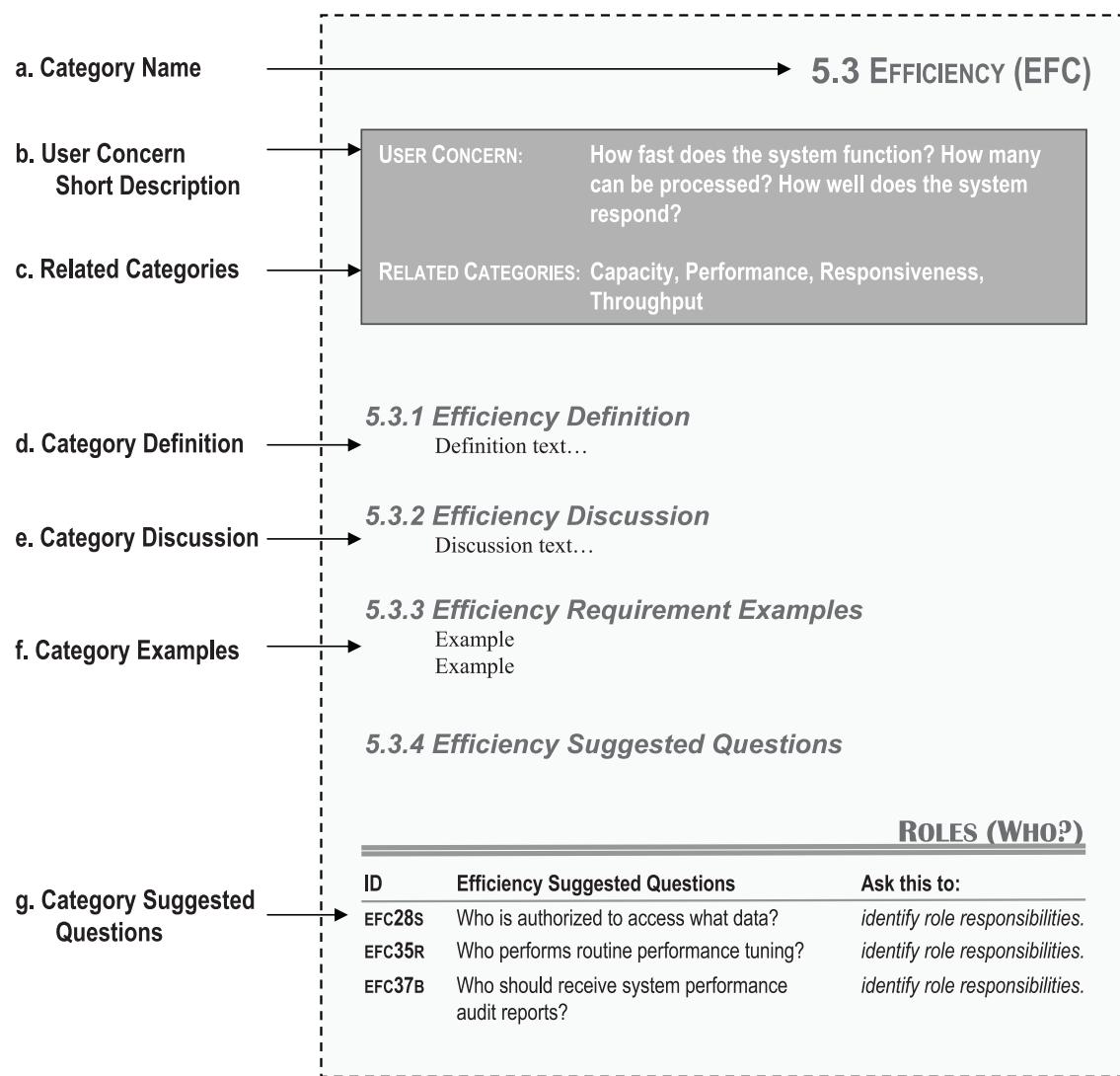
Finally, each individual question is followed by an explanation of the target or intent of the question for further understanding of the question and why it might be asked. These are placed along the right-hand margin under the label **Ask this to:**

This book describes common categories of nonfunctional requirements that apply to software systems. Your organization might determine that additional categories are necessary based on the particular products and services offered. The components of the above “anatomy of a nonfunctional category” might serve as a pattern for developing and defining additional nonfunctional categories (a comprehensive list of nonfunctional types is included in Chapter 4, “Understanding Nonfunctional Requirements”).



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Figure II-B Example: Anatomy of a Nonfunctional Category



HOW TO USE THE SUGGESTED QUESTIONS

This book offers questions that might be asked. The collective set of questions is certainly not an exhaustive or all-inclusive list. The following steps are intended to provide guidance for helping you organize the questions you choose to ask.

Step 1: Where are you in the process?

Questions should be chosen based on the iterative process of eliciting requirements and the level of detailed information already known (business, user, or system), as well as the level of detail being sought. Nonfunctional requirements are generally explored in the development of system level requirements as described in Chapter 1, “Requirements in Context.” It is important to note that in many of the possible questions, the term system is used interchangeably to reference a business process or sub-process that is automated by a software system or software application, or to a product that relies on a software component to operate.

Step 2: Who are your stakeholders?

You must consider the requirements role (supplier or receiver) when deciding which questions to ask. You’ll find that some questions are repeated from one stakeholder to another. Although it is possible to find one person who can effectively answer all the questions, don’t count on it—besides, two heads are better than one. Get to know the stakeholders. Chapter 2, “Involve the Right Stakeholders,” guides you through a technique that helps to identify the knowledge expertise of individual stakeholders, as well as uncover business areas and topics that need stakeholder representation. The stakeholder profiling technique explained in Chapter 2 will help you get the right stakeholders engaged.

Step 3: What nonfunctional categories are relevant to your project?

Chapter 4, “Understanding Nonfunctional Requirements,” introduces the 14 common categories of nonfunctional requirements, while chapters 5 through 7 provide definitions, examples, and suggested questions.



Step 4: What are your success criteria?

It is highly unlikely that you will ask all of the questions in this book for a given project effort. Each individual question is followed by a brief explanation of the target or intent of the question. Review these explanations to help you select questions based on what you’re trying to accomplish. Furthermore, some questions are very similar, and depending on the perspective of the stakeholder and the requirements focus, you might choose one question over another. However, you might intentionally ask similar questions because you anticipate a varied response. Chapter 3, “Interviewing Tips and Tools,” explains that slight changes in the wording of a question can significantly alter the interpretation of the question.

Step 5: Prepare your list of questions.

You are encouraged to rewrite the questions and put them in your own words. Chapter 3, “Interviewing Tips and Tools,” guides you through effective preparation activities and describes how to conduct a successful interview.

Step 6: Follow your game plan.

Ask your list of prepared questions (explained in Chapter 3).

Step 7: Anticipate follow-up questions.

Apply an active listening strategy (explained in Chapter 3), and ask questions to probe further into the response to your original question. Rephrase the response and ask questions to both clarify and confirm that you understand the response you get.



A nonfunctional requirement
is a specification of how well
a software system
must function.



4 UNDERSTANDING NONFUNCTIONAL REQUIREMENTS

IN THIS CHAPTER:

4.1 Are the Definitions Dysfunctional?	99
4.2 Nonfunctional Challenges	105
4.3 Vital, Yet Why so Difficult?	106
4.4 Classification Efforts	110
4.5 A User-Focused Approach	118
4.6 Chapter Summary	123
4.7 Suggested Reading.....	124

This chapter is intended to help you understand the challenges of hard-to-identify nonfunctional requirements. We'll take a look at problems with definitions of the term *nonfunctional*, as well as factors that contribute to difficulties in understanding these requirements. Additionally, we'll examine various industry efforts to classify nonfunctional requirements. Finally, we'll define the user-focused classification scheme chosen for this book.

Over years of mentoring and coaching on business analysis and requirements management practices, I have repeatedly found that people struggled in formulating nonfunctional



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

requirements. When asked to review a requirements specification (please refer to Figure 1-10 in Chapter 1), I found the functional requirements section to be quite lengthy (not necessarily a metric of quality, mind you), while the nonfunctional requirements section was, far too often, empty.

This book is my personal quest to help people understand these overlooked nonfunctional requirements. My experience has shown that an increased knowledge leads to people asking more questions that elicit nonfunctional requirements. This generally results in a significant reduction in missed requirements. The goal is to define **better requirements** (helping project teams identify requirements that they otherwise wouldn't have), which contribute to **better systems**.

Get the Users “IN” the Doghouse and Keep Yourself “OUT.”



4.1 ARE THE DEFINITIONS DYSFUNCTIONAL?

The complexity of software systems is due partly to its functionality, and partly to its nonfunctionality. For the time being, let's simply define functionality as what the system must do and nonfunctionality as how well the system must do it.

We know that functional requirements are important. Try leaving out necessary functions and see how long it takes for users and others to complain. There are many good books that adequately explain functional requirements including definitions, examples, and guidance on how to elicit, analyze, represent, and validate them. In other words, a fair amount of coverage in literature has been given to system functionality. Therefore, I only briefly incorporated functional requirements in this book because of their integrated relationship to nonfunctional requirements.

Let's begin with reference to a variety of definitions for functional and nonfunctional requirements from well-known and respected sources for comparison purposes. Although functional definitions are included, the focus of this book is on defining nonfunctional requirements. Additionally, keep in mind that all of the referenced definitions are within the terminology and scope of software requirements (refer to Chapter 1, "Requirements in Context," for an overview of software requirements terminology).

4.1.1 A Variety of Definitions

Only five definition sources were selected, shown in Table 4-1 on the next page, and used in my comparative analysis to derive the definition I use henceforth in the context of this book. Some sources were left out, frankly because I didn't agree with them. I struggle with so-called definitions that don't actually define the term but just provide examples.

I won't even provide the source for my least favorite definition: "A nonfunctional requirement is anything that is not functional." Are you kidding me? How do you take a source seriously with a definition like that?



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Table 4-1 Definitions for Functional and Nonfunctional Requirements

<p><i>Functional requirement</i>—A system/software requirement that specifies a function that a system/software system or system/software component must be capable of performing. These are software requirements that define behavior of the system, that is, the fundamental process or transformation that software and hardware components of the system perform on inputs to produce outputs.” [Thayer, 2000]</p>	<p><i>Nonfunctional requirement</i>—in software system engineering, a software requirement that describes not what the software will do, but how the software will do it, for example, software performance requirements, software external interface requirements, software design constraints, and software quality attributes. Nonfunctional requirements are difficult to test; therefore, they are usually evaluated subjectively.” [Thayer, 2000]</p>
<p><i>Functional requirements</i>—These are statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what the system should not do.” [Sommerville, 2007]</p>	<p><i>Non-functional requirements</i>—These are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process and standards. Non-functional requirements often apply to the system as a whole. They do not usually just apply to individual system features or services.” [Sommerville, 2007]</p>
<p><i>Functional requirement</i>—Something that the product must do. Functional requirements are part of the fundamental processes of the product.” [Robertson, 1999]</p>	<p><i>Non-functional requirement</i>—A property, or quality, that the product must have, such as an appearance, or a speed or accuracy property.” [Robertson, 1999]</p>
<p><i>Functional requirement</i>—A statement of a piece of required functionality or a behavior that a system will exhibit under specific conditions.” [Wiegers, 2003]</p>	<p><i>Nonfunctional requirement</i>—A description of a property or characteristic that a software system must exhibit or a constraint that it must respect, other than an observable system behavior.” [Wiegers, 2003]</p>
<p><i>Behavioral requirements</i>—Those requirements that specify the inputs (stimuli) to the system, the outputs (responses) from the system, and behavioral relationships between them; also called functional or operational requirements.” [Davis, 1993]</p>	<p><i>Nonbehavioral requirements</i>—Requirements that describe the required overall attributes of the system, including portability, reliability, efficiency, human engineering, testability, understandability, and modifiability.” [Davis, 1993]</p>



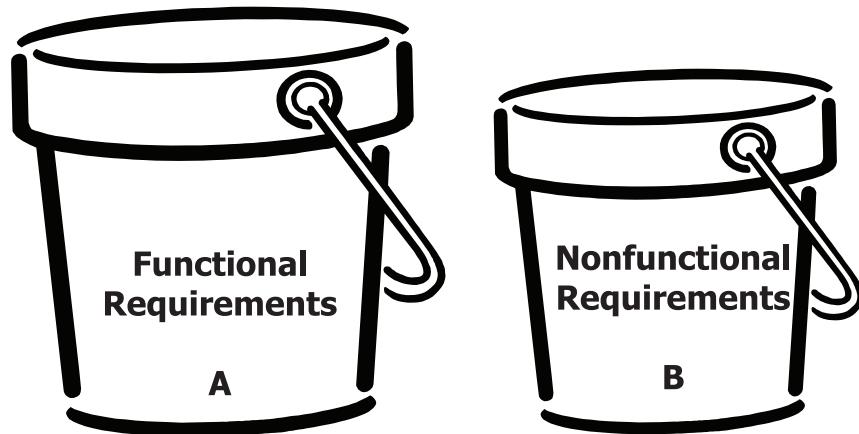
At any rate, my comparison analysis of the definitions is explained in Section 4.1.3, “A Simplified Definition.” Before jumping ahead, I challenge you to spend a few minutes analyzing these definitions yourself.

For the sake of avoiding an argument, let’s assume that there are some good definitions. However, as the example definition illustrates, there are some not-so-good ones. Let’s consider the latter in the next section. (Besides, the scenic route heightens the suspense of revealing my “simplified definition.”)

4.1.2 A Wrong Definition?

In a conversation with Stephen Withall, author of *Software Requirement Patterns* [Withall, 2007], Withall pointed out that it seems we insist on separating the system-level requirements into two buckets as pictured in Figure 4-1 (system-level requirements are explained in Chapter 1, “Requirements in Context”). Based on quantity, there are typically many more functional requirements than nonfunctional, which is why the functional bucket is larger.

Figure 4-1 System-Level Requirement Buckets



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Withall further hypothesized that if we define functional requirements (Bucket A) as *what the system must do*, then one logical definition for nonfunctional requirements (Bucket B) could be *what the system should not do*. Thus, the term “nonfunctional” makes perfect sense, right? Wrong!

It is important to identify that nonfunctional requirements are NOT:

- ◆ **The opposite of functional requirements** (un-functional). This line of thinking didn't work so well in the soft drink industry for 7UP® as the “Un-cola.”
- ◆ **The definition of a system that fails to function**, which ironically is the meaning of the word *dysfunctional*. I don't think this is the direction intended.
- ◆ **The end-all, catch-all of whatever doesn't fit the definition of functional requirements**. That is, if it doesn't belong in Bucket A, then it automatically defaults into Bucket B. Thus before long, Bucket B becomes the scary, elusive, black hole! With an image like this one, it is no wonder that project teams fail to adequately include these important aspects of the software system.

Theoretically (in reality?), let's say there are *only* two buckets. Everything defined as “functional” naturally falls into the first bucket (Bucket A). Some would argue that the only clean, pure, and infallible definition of “nonfunctional” is *everything that's not in the functional bucket*. The very fact that the name of the second bucket is “nonfunctional” is a strong indication that *all things that aren't functional requirements* are, indeed, what the bucket contains. This might explain why we see many definitions for nonfunctional requirements resorting to mentioning the name of the first bucket—as if that's their way of making sure there's nothing left over.

Now suppose the second bucket was wrongly labeled or horribly misnamed. (Gasp!) Did you know that for a long, long time, people believed that our planet was flat? What if there are really more than two buckets at the system requirements level? Is it possible that we could prove the theory of “nonfunctional” requirements to be false? Maybe, just maybe mind you, we discover there are at least four requirement buckets: functional, operation, revision, and



transition. We'll explore these possibilities in sections 4.4, "Classification Efforts," and 4.5, "A User-Focused Approach."

Let's face the reality of current nonfunctional definitions. This next section unveils my simplified definition.

4.1.3 A Simplified Definition

Let's start by analyzing the nonfunctional definitions included in Table 4-1 on page 100. First, we'll remove the following elements (with the remaining text shown in Table 4-2 on the following page):

- ◆ "Nonfunctional requirement" (or other similar reference) from the beginning of each definition simply to cut down on the words.
- ◆ "Examples" of nonfunctional requirements.
- ◆ "Descriptive" information about nonfunctional requirements that don't add to the definition.
- ◆ Inclusion or reference to "functional requirements."

From the remaining text, we can extract words that are common across the definitions. This reveals three words that are used to identify the subject of the definition: *software*, *system*, and *product*. In reviewing the definitions behind these words, I interpret the terms as interchangeable and I prefer the term *system*.

Upon further analysis of the scaled-down versions, we can identify three additional words that are used to *describe* the subject (system): constraint, quality, and attribute. Naturally, I had to dig up definitions of these terms. Unfortunately, the definitions added little or no value. The more I dug for answers and depth of understanding, the more I felt like I was just digging a deeper and wider hole. Needless to say, I was frustrated. And, in this discouraged state, I began to ponder: if experts and industry organizations come up with definitions as confusing as these, is it any wonder that the less-expert of us write poor requirement specifications? Don't dismay; help is on the way.



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Table 4-2 Scaled-down Nonfunctional Definitions

Original Definition	Scaled-down Version
<p><i>“Nonfunctional requirement</i>—in software system engineering, a software requirement that describes not what the software will do, but how the software will do it, for example, software performance requirements, software external interface requirements, software design constraints, and software quality attributes. Nonfunctional requirements are difficult to test; therefore, they are usually evaluated subjectively.” [Thayer, 2000]</p>	<p>... how the <u>software</u> will do it.</p>
<p><i>“Non-functional requirements</i>—These are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process and standards. Non-functional requirements often apply to the system as a whole. They do not usually just apply to individual system features or services.” [Sommerville, 2007]</p>	<p>... constraints on the services or functions offered by the <u>system</u>.</p>
<p><i>“Non-functional requirement</i>—A property, or quality, that the product must have, such as an appearance, or a speed or accuracy property.” [Robertson, 1999]</p>	<p>... property or quality that the <u>product</u> must have</p>
<p><i>“Nonfunctional requirement</i>—A description of a property or characteristic that a software system must exhibit or a constraint that it must respect, other than an observable system behavior.” [Wiegers, 2003]</p>	<p>... property or characteristic that a <u>software system</u> must exhibit or constraint that it must respect</p>
<p><i>“Nonbehavioral requirements</i>—Requirements that describe the required overall attributes of the system, including portability, reliability, efficiency, human engineering, testability, understandability, and modifiability.” [Davis, 1993]</p>	<p>... required overall attributes of the <u>system</u>.</p>



At the conclusion of my extensive analysis, I derived the following simplified definition for use in the context of this book:

**A nonfunctional requirement
is a specification of how well
a software system
must function.**

Alas! Now that we have a clear definition, all of our problems have disappeared—unfortunately, not. Moving right along ... Let's look at the difficulties of eliciting nonfunctional requirements.

4.2 NONFUNCTIONAL CHALLENGES

Perhaps surprisingly, nonfunctional requirements have received far too little attention in software engineering literature. They are certainly less well understood compared to other factors in the development and selection of high quality systems. Inconsistent terminology, confusing definitions (as seen in the previous section), and the absence of a universally accepted classification scheme make understanding nonfunctional requirements a challenge.

The term “nonfunctional” has been disliked by many people in the software requirements industry—most especially by me—for a long time. This may be due in part by a misperception of the term itself. People often correlate negation—for example, words in the English language such as “no,” “not,” “non-,” and “none”—with importance, or the lack thereof, as is often the perception with the term nonfunctional.

In addition to alternative names such as *quality attributes*, *quality requirements*, and *nonbehavioral requirements*, nonfunctional requirements also have been referred to by nicknames such as “*ilities*” and “*ties*.“ These nicknames are derived from descriptors that end in the suffix “-ility,” such as portability, reliability, usability, and survivability. Nonfunctional requirements are commonly characterized by adjectives, while functional requirements are characterized by verbs, according to Suzanne and James Robertson [Robertson, 2006].



No doubt also stemming from inconsistent terminology and confusing definitions, we cannot agree on how to spell these important requirements. Is it “non-functional” or “nonfunctional” (with or without a hyphen)? This may be quite trivial to many, but it is still an indication of the lack of uniformity. I prefer spelling the term without the hyphen. And, for what it’s worth, so does *Webster’s Dictionary!* However, there are specific references throughout the book where I use the hyphenated spelling—this is only done out of respect for the other author. Within your organization, I recommend choosing one and being consistent in using it.

Without a uniform classification, a possible starting point for eliciting nonfunctional requirements might be an “overwhelmability” list of nonfunctional types, such as Table 4-3 on the following page. It is highly unlikely, however, that all of these types will apply to your particular project and your specific organization. This is due to the subjective and relative nature of nonfunctional requirements (explained in Section 4.3). Quite honestly, working from an overwhelming list such as this would not exactly make me eager to focus on nonfunctional requirements!

In conclusion, the significance of the term name and how it is spelled pale in comparison to a consistent definition of what the term means. Let’s turn our attention to understanding more about the nature of nonfunctional requirements that make them difficult to identify.

4.3 VITAL, YET WHY SO DIFFICULT?

Errors of omission or failing to properly account for
nonfunctional requirements are generally acknowledged
to be among the most expensive errors and the most difficult
to correct following the implementation of a software system.

Nonfunctional requirements are vital to the success of software systems. If nonfunctional requirements are not properly addressed, undesirable results occur such as unsatisfied users, developers, and clients, and schedule and budget overruns to correct the software that was developed without the nonfunctional requirements in mind.



CHAPTER 4: UNDERSTANDING NONFUNCTIONAL REQUIREMENTS

Table 4-3 Overwhelmability List of Nonfunctional Types (based on [Chung, 2000])

Absorbability	Confidentiality	Guidance	Planning Cost	Space Boundedness
Access Control	Configurability	Human Engineering	Planning Time	Space Performance
Accessibility	Connectivity	Impact Analyzability	Plasticity	Specificity
Accountability	Consistency	Implementability	Portability	Stability
Accuracy	Controllability	Independence	Precision	Standardizability
Adaptability	Coordination Cost	Informativeness	Predictability	Storability
Additivity	Coordination Time	Inspection Cost	Process Management Time	Structuredness
Adjustability	Correctness	Inspection Time	Productivity	Subjectivity
Affordability	Coupling	Installability	Project Stability	Supportability
Agility	Customer Evaluation Time	Integrity	Project Tracking Cost	Surety
Appealability	Customer Loyalty	Interchangeability	Promptness	Survivability
Attractiveness	Customizability	Internal Consistency	Prototyping Cost	Susceptibility
Auditability	Data Space Performance	Interoperability	Prototyping Time	Sustainability
Augmentability	Decomposability	Intuitiveness	Quality	Tankness
Authenticity	Degradation of Service	Iterativeness	Quantitativeness	Testability
Autonomy	Deliverability	Learnability	Readability	Testing Time
Availability	Dependability	Legibility	Readiness	Throughput
Buffer Space Performance	Development Cost	Leveragability	Reconfigurability	Timeliness
Capability	Development Time	Likeability	Recoverability	Toleranceness
Capacity	Disposable	Main Memory Performance	Recovery	Traceability
Changeability	Distributivity	Maintainability	Recyclability	Trainability
Clarity	Diversity	Maintenance Cost	Reengineering Cost	Transferability
Cleanability	Domain Analysis Cost	Maintenance Time	Reliability	Transitionability
Cleanliness	Domain Analysis Time	Manageability	Repeatability	Transparency
Code Space Performance	Duplicatability	Maneuverability	Replaceability	Understandability
Cohesiveness	Ease of use	Maturity	Replicability	Uniform Performance
Commonality	Efficiency	Mean Performance	Response Time	Uniformity
Communication Cost	Elasticity	Measurability	Responsiveness	Unparochialness
Communication Time	Enhanceability	Migratability	Retirement Cost	Usability
Communicativeness	Evolvability	Mobility	Reusability	User-friendliness
Comparability	Execution Cost	Modifiability	Risk Analysis Cost	Validity
Compatibility	Expandability	Modularity	Risk Analysis Time	Variability
Completeness	Expendability	Multiness	Robustness	Verifiability
Component Integration	Extendibility	Naturalness	Safety	Versatility
Cost	Extensibility	Nomadicity	Scalability	Versionability
Component Integration	External Consistency	Observability	Security	Visibility
Time	Fault-Tolerance	Off-Peak-Period Performance	Self-containedness	Volatility
Composability	Feasibility	Operability	Self-descriptiveness	Wrappability
Comprehensibility	Flexibility	Operating Cost	Sensitivity	Workability
Computability	Foolproof-ability	Peak Period Performance	Similarity	Zero-defectness
Conceptuality	Formality	Performability	Simplicity	
Conciseness	Generality		Software Cost	



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

There are numerous contributing factors or reasons that the nonfunctional requirements are difficult to identify. Three major factors that influence the complex nature of nonfunctional requirements are as follows:

- (1) **SUBJECTIVE.** They can be viewed, interpreted, and evaluated differently from user to user. For instance, one user contends that a two-second response time is acceptable, while another user feels that a sub-second response time is necessary. Similarly, what is deemed “easy to learn” by one user may not be evaluated as such by another, less-experienced user.
- (2) **RELATIVE.** The interpretation of relevance and importance might vary depending on the specific system under consideration, as well as the products and services produced by a business. For example, organizations in the financial industry such as banks and insurance companies are likely to view precision, accuracy of information, and access control (security) of the confidential information as prominent quality attributes. Manufacturing organizations that use software to help make their products typically emphasize safety as an attribute to prevent employee accidents. On a different track, a software system developed for a one-time-use project such as a data conversion will probably not emphasize the ease of repair or the maintainability quality attribute.
- (3) **INTEGRATED.** The goals of nonfunctional requirements can conflict with one another. Nonfunctional requirements typically have a broad effect on systems. That is, nonfunctional requirements are combined to produce a whole, complete system.

For example, there is an interesting and perhaps complex relationship between flexibility requirements, which tend to influence the nature of the software, and performance (e.g., efficiency) requirements that tend to affect the hardware needed. Flexibility and efficiency are so closely intertwined that the project sponsor is often forced to make trade-off decisions between high performance/high costs and less flexibility as shown in Table 4-4. Keep in mind that this is a generalized list of examples, and as such will not always happen. Further, some trade-offs are less important now than they were previously—now that hardware is so powerful



Table 4-4 Typical Quality Factor Trade-offs [Charette, 1990]

Integrity vs. Efficiency	The additional code and processing required to control the access of the software or data usually lengthens run-time and requires additional code.
Usability vs. Efficiency	The additional code and processing requirements to ease an operator's tasks or provide more usable output usually lengthens run-time and increases storage.
Maintainability vs. Efficiency	Optimized code increases maintainer's efforts. Using components, modules, instrumentation, etc., however, will increase overhead.
Portability vs. Efficiency	The use of direct code, optimized code, or system utilities decreases the portability of the system.
Flexibility vs. Efficiency	Generally, a flexible system will increase overhead.
Interoperability vs. Efficiency	The added overhead for data conversion and interface routines decreases operating efficiency.
Flexibility vs. Integrity	Flexibility requires very general structures. Security may be harder to ensure.
Reusability vs. Integrity	As above, reusable software provides potential security problems.
Interoperability vs. Integrity	Coupled systems allow for more paths that can allow either accidental or purposeful access to data.
Reusability vs. Reliability	The generality required by reusable software makes providing for error tolerance and accuracy difficult.



and storage space is relatively inexpensive. In some instances, you can reduce the effects of the trade-offs with sophisticated software (usually associated with a sophisticated price tag).

When a trade-off (or potential conflict) between nonfunctional attributes arises, how should it be handled in the specification? At a minimum, it should point out the trade-off because there might be associated project risks.

Trade-offs or potential conflicts in nonfunctional requirements are sometimes the 800-pound gorilla in the room. Year after year on project after project, we hear the demands from project sponsors, business partners, and users for software systems that are “better, cheaper, and faster.” Unfortunately, too often the feasibility and affordability measured in terms of resources (e.g., people, software, and hardware) to deliver all that is requested on time and within budget are nearly impossible to accommodate.

4.4 CLASSIFICATION EFFORTS

Although there is no formal definition of nonfunctional requirements, there has been considerable work done on characterizing and classifying them. In this section, we’ll look at three examples of such prior works. Additionally, we’ll compare several efforts and explain the classification chosen for this book.

If the example classifications by Boehm, Sommerville, and Withall that follow serve no other purpose, I find the 30-year span of time between them to be of interest.

4.4.1 Boehm’s Software Quality Characteristics Tree

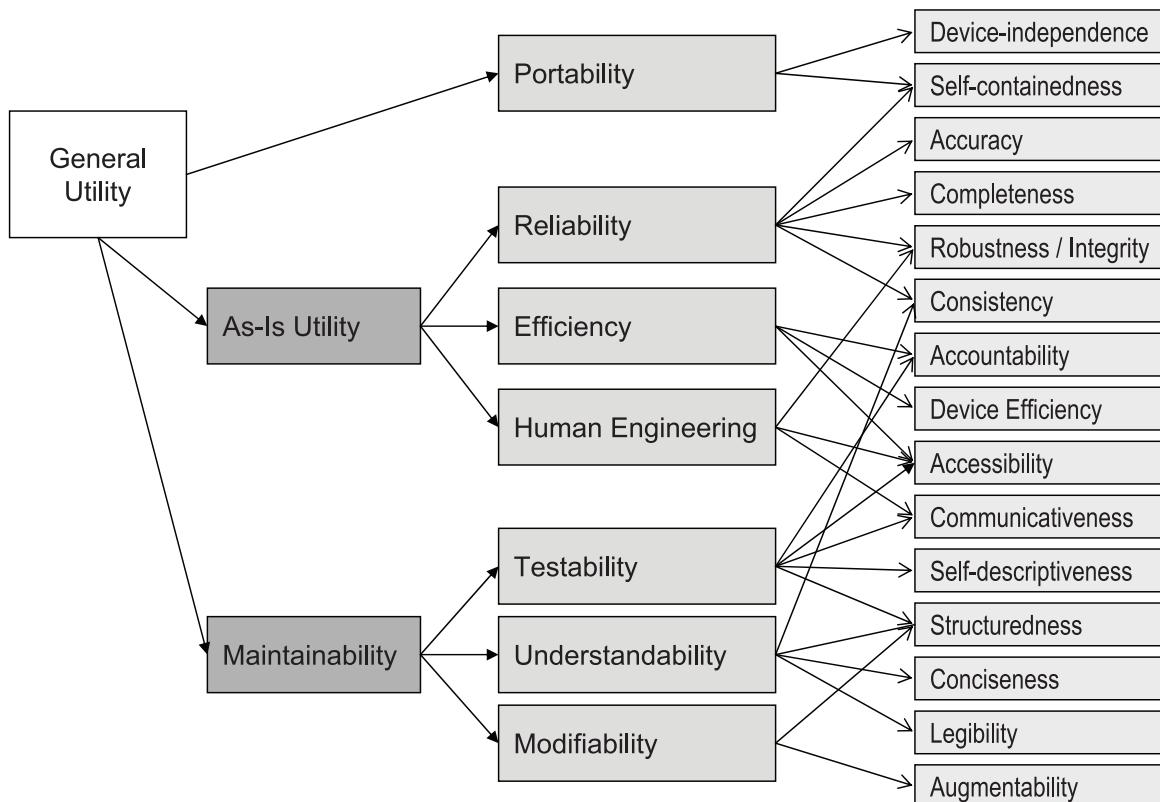
Similar to a family tree that is used to trace one’s heritage, Barry Boehm’s quality attribute tree (shown in Figure 4-2) implies that when a parent quality is met, so too are its offspring qualities. Boehm reported the results of a study regarding characteristics of software quality.

A key finding from Boehm’s study showed that explicit attention to characteristics of software quality can lead to significant savings in software lifecycle costs.



CHAPTER 4: UNDERSTANDING NONFUNCTIONAL REQUIREMENTS

Figure 4-2 Software Quality Attributes Tree [Boehm, 1976]



4.4.2 Sommerville's Nonfunctional Requirement Types

In this classification example, Ian Sommerville stresses that nonfunctional requirements arise because users need to achieve certain goals. These may be brought about by budget constraints, organizational policies, the need for interoperability with other software and hardware, the need for a certain development process to be followed, and external factors such as safety and security regulations. Figure 4-3 shows Sommerville's classification of nonfunctional requirements into three main requirement categories:

- (1) Organizational requirements are constraints placed upon the development process of the system. They may be included because the customer for a system wishes to influence this process. Organizational requirements include requirements on development standards and methods that must be followed. For example, development will conform to International Standards Organization (ISO) 9000 standards, or the system must be developed for Microsoft Windows XP®.
- (2) Product requirements specify the desired characteristics that a system or subsystem must possess. Most product requirements are concerned with specifying constraints on the behavior of the executing system. For example, the product must comply with safety regulations.
- (3) External requirements may be placed upon both the product and the process, and are derived from the environment in which the system is developed. Therefore, these requirements may be based on application domain information, legal considerations, the need for the system to work with other systems, or even basic natural laws of physics.

4.4.3 Withall's Software Requirement Patterns

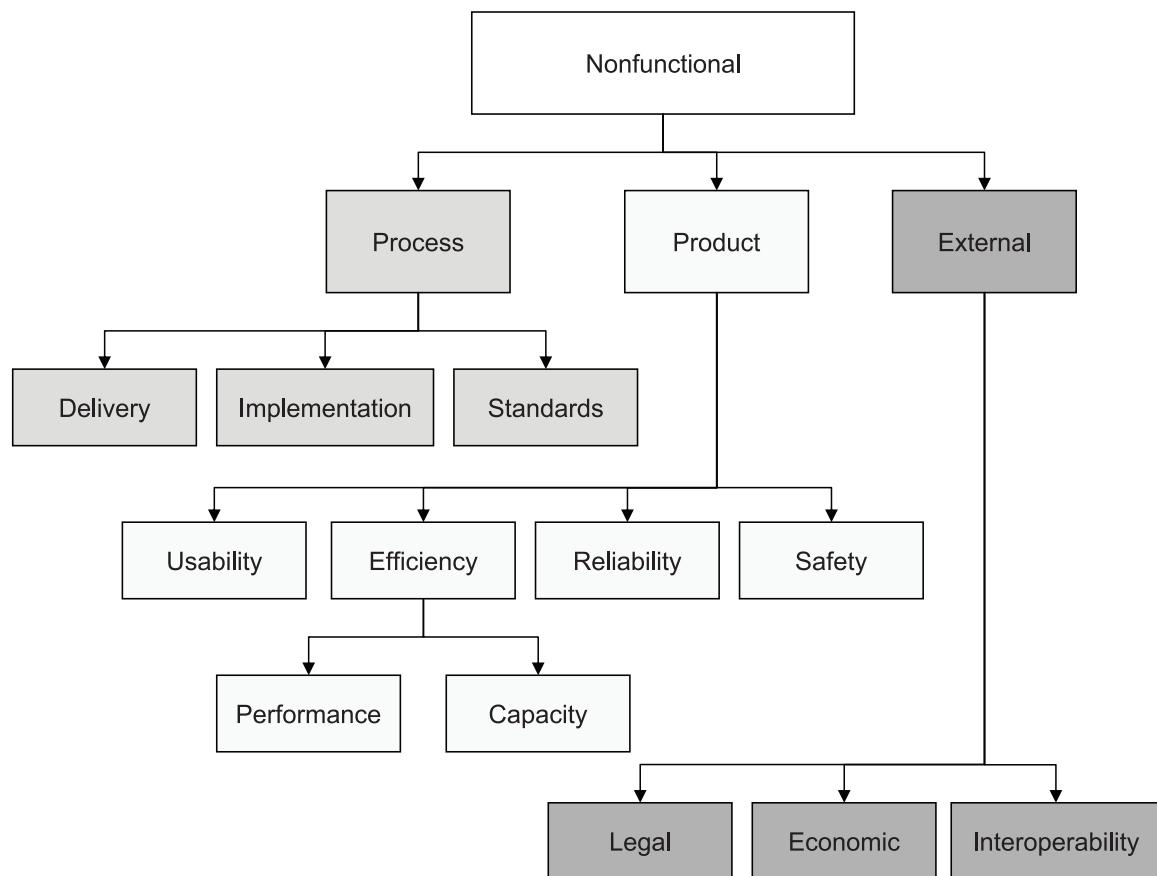
In this third example, Stephen Withall presents a classification based on patterns rather than functional and nonfunctional requirements. According to Withall, a **requirement pattern** is “*an approach to specifying a particular type of requirement.*”

Design patterns have been acknowledged for more than 15 years. Withall has brilliantly applied the concept of patterns to software requirements. He indicates that many types of



CHAPTER 4: UNDERSTANDING NONFUNCTIONAL REQUIREMENTS

Figure 4-3 Nonfunctional Requirement Types [Sommerville, 1992]



requirements crop up over and over in all kinds of systems. Requirement patterns therefore aim to let you specify better requirements more precisely with less effort. A requirement pattern applies to an individual requirement. Requirement patterns provide guidance on situations that recur in all systems such as how to tackle a requirement, what information ought to be conveyed in the requirement, as well as extra topics to consider, including development and testing.

Withall has created 37 patterns, grouped into eight domains as shown in Figure 4-4. Each domain has a theme, which all its patterns share. For readability, Figure 4-4 shows only the most significant relationships between patterns in different domains.

4.4.4 Comparing Nonfunctional Classification Efforts

In addition to the examples above from Boehm, Sommerville and Withall, there are lists provided by other authors, as well as standardization bodies. Which list should you use? The answer is simple: none of these represent the end-all perfect list. Rather, each is intended to be used as a checklist for what you might consider. You should ask yourself which nonfunctional characteristics are important to the system you encounter. Then specify the requirements related to those you identified.

I had to undergo a similar selection process in order to determine which nonfunctional categories to include in this book.

I started by studying each list independent of the others in order to understand each author's rationale behind his classification. From there I literally placed the lists side by side, as shown in Table 4-5 on page 116, and searched for similarities and differences between the lists.

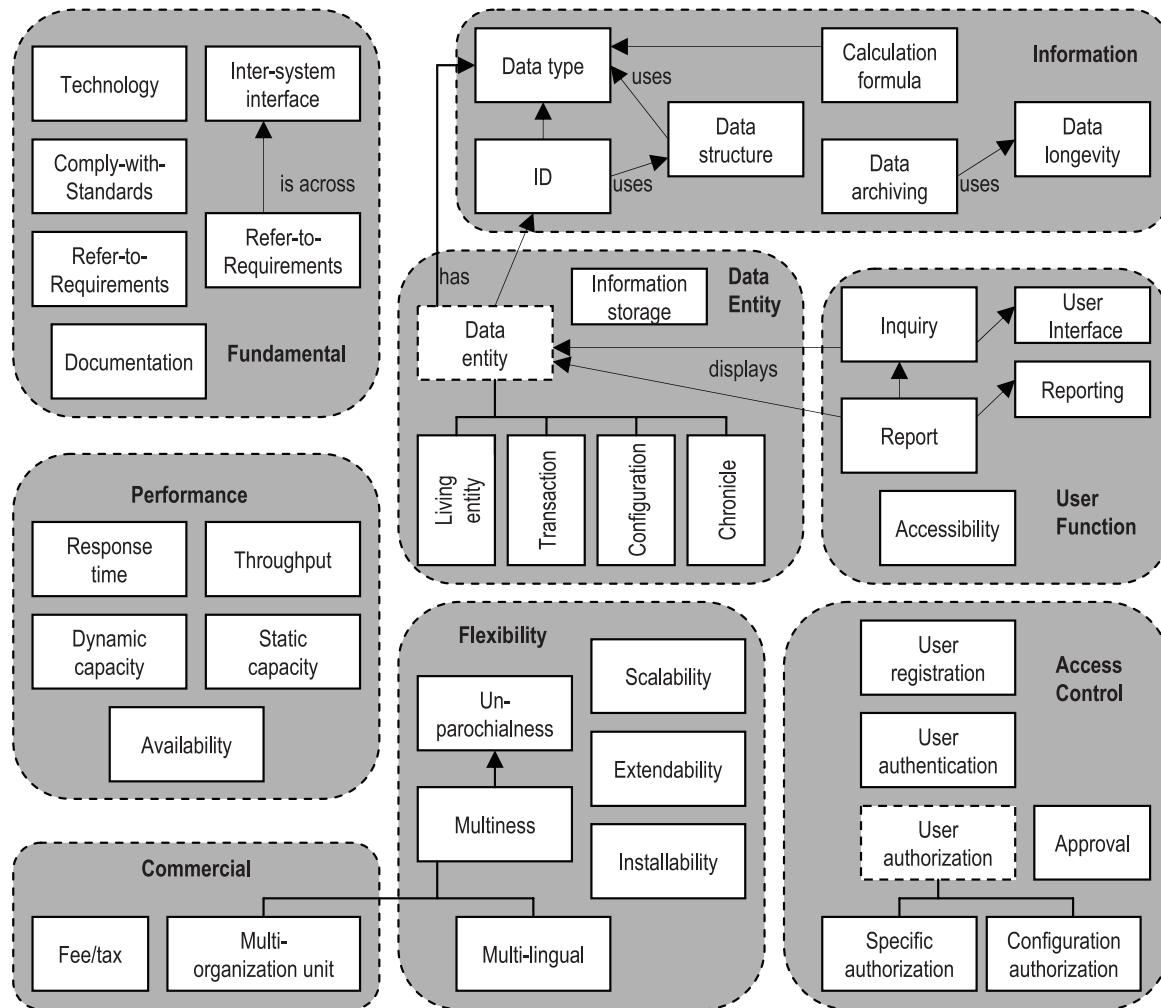
Next, I noted the frequency of occurrence of any given nonfunctional category across the various lists, as shown in Table 4-6 on page 117. Generally, only those nonfunctional categories that appeared more than once were considered. For instance, *efficiency*, *reliability*, and *usability* are categories included in all the lists.

Unsurprisingly, I found that the terms and definitions differed. For example, *testability* found in McCall and Matsumoto's list appears to match the definition of *verifiability* found in Keller's list. Furthermore, Keller placed *verifiability* in a group called *design*, while Deutsch and



CHAPTER 4: UNDERSTANDING NONFUNCTIONAL REQUIREMENTS

Figure 4-4 Software Requirement Patterns [Withall, 2007]



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Willis placed *verifiability* in a group called *management*. As you might guess, the definitions of the groups didn't exactly match.

The nonfunctional categories included in this book are indicated in the far right column of Table 4-6, and blend the works of Keller [Keller, 1990], Deutsch and Willis [Deutsch, 1988], and McCall and Matsumoto [McCall, 1980]. The user-focused classification of these common nonfunctional categories is explained in the next section, "A User-focused Approach."

Table 4-5 A Comparison of Nonfunctional Classifications

McCall & Matsumoto 1980	Keller (RADC) 1990	Deutsch & Willis 1988	ISO/IEC 9126 1991	Sommerville 1992	Gib 2005
Operation	Performance	Functional	Functionality	Product	Quality
Correctness	Efficiency	Integrity	Accurateness	Usability	Availability
Efficiency	Integrity	Reliability	Compliance	Efficiency	• Reliability
Integrity	Reliability	Survivability	Interoperability	• Performance	• Maintainability
Reliability	Survivability	Usability	Security	• Space	• Integrity
Usability	Usability	Performance	Suitability	Reliability	Adaptability
Revision	Design	Efficiency	Reliability	Portability	• Flexibility
Flexibility	Correctness	Correctness	Fault tolerance	Organizational	• Upgradeability
Maintainability	Maintainability	Safety	Maturity	Delivery	Usability
Testability	Verifiability	Interoperability	Recoverability	Implementation	• Entry-Level Experience
Transition	Adaptation	Change	Usability	Standards	• Training
Interoperability	Expandability	Maintainability	Learnability	External	• Handling Ability
Portability	Flexibility	Expandability	Operability	Interoperability	• Likeability
Reusability	Interoperability	Flexibility	Understandability	Ethical	• Demonstrability
	Portability	Portability	Efficiency	Legislative	Resource Saving
	Reusability	Reusability	Resource behavior	• Privacy	Financial
		Management	Time behavior	• Safety	Time
		Verifiability	Maintainability		Effort
		Manageability	Analyzability		Equipment
			Changeability		Workload/Capacity
			Stability		Throughput
			Testability		Response Time
			Portability		Storage Capacity
			Adaptability		
			Conformance		
			Installability		
			Replaceability		



CHAPTER 4: UNDERSTANDING NONFUNCTIONAL REQUIREMENTS

Table 4-6 Common Nonfunctional Categories

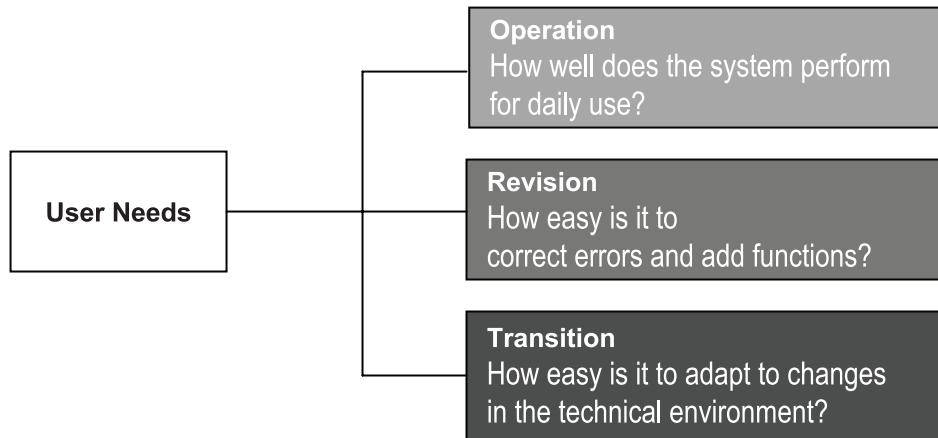
	McCall 1980	Keller 1990	Deutsch 1988	ISO/IEC 9126; 1991	Sommerville 1992	Gilb 2005	This Book
Availability						X	X
Correctness	X	X	X				
Efficiency	X	X	X	X	X	X	X
Expandability (Scalability)		X	X				X
Flexibility	X	X	X			X	X
Integrity	X	X	X			X	X
Interoperability	X	X	X	X	X		X
Maintainability	X	X	X	X		X	X
Portability	X	X	X	X	X		X
Reliability	X	X	X	X	X	X	X
Reusability	X	X	X				X
Safety			X		X		
Security				X			X
Survivability		X	X				X
Testability	X			X			
Usability	X	X	X	X	X	X	X
Verifiability		X	X				X



4.5 A USER-FOCUSED APPROACH

Nonfunctional requirements can be classified based on the user's need for software quality, which I refer to as the *user-focused approach*. Addressing a user concern will necessitate the formulation of a number of functional requirements, but the user concerns will also act to constrain other requirements that are characteristic of nonfunctional requirements. As shown in Figure 4-5, user concerns for software quality are grouped under three important aspects: its operational characteristics, its ability to undergo change, and its adaptability to new environments.

Figure 4-5 User Needs for Software Qualities [McCall, 1980]



In order to apply a user-focused approach, it is necessary to understand who the user is. The software user is any person who comes in contact with the software system. User contact with the software system might occur in any of the following ways:



- (a) **Using the functionality (*operation*)**. The user perceives the system as an electronic tool that helps to automate what would otherwise be done manually. From this point of view, the user is concerned with how well the system operates.
- (b) **Changing source code or data that drive the system (*revision*)**. The user perceives the system as a set of programmed language statements. These statements are treated as a problem that must be solved. The system must be analyzed, modified, tested, and implemented as problems arise, or the business changes the way it operates.
- (c) **Managing the upkeep of the software (*transition*)**. From this point of view, the system carries similar characteristics as hardware. That is, the user is concerned with aspects such as packaging, transport, and compatibility with other systems.

Regardless of which form of contact the user has with the system, all users are after the same thing—software quality.

4.5.1 User Needs for Software Quality

Depending on the user's contact with the software system and the purpose and perspective for using it, user needs vary. To better understand user needs, the three aspects of software quality are further subdivided, as summarized in Table 4-7 on page 120.

The three main groups of user needs for software quality are briefly introduced here. Each group, as well as the subdivision of nonfunctional categories within each group, is explained in the subsequent chapters of Part Two.

The ***operation*** group describes the user needs for a system that performs or functions well. The operation group subdivides into the following requirement categories:

- ♦ **Access Security**—how well the system is safeguarded against deliberate and intrusive faults from internal and external sources.
- ♦ **Availability**—how dependable the system is (able to function) during “normal operating times.”



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Table 4-7 A User-focused Approach

User Needs	User Concerns with the Software System	Nonfunctional Categories
Operation How well does the system perform for daily use?	How well is it guarded against unauthorized access?	Access Security (ACS)
	How dependable is it during normal operating times?	Availability (AVL)
	How fast, how many, and how well does it respond?	Efficiency (EFC)
	How accurate and authentic are the data?	Integrity (INT)
	How immune is the system to failure?	Reliability (REL)
	How resilient is the system from failure?	Survivability (SRV)
	How easy is it to learn and operate the system?	Usability (USE)
Revision How easy is it to correct errors and add on functions?	How easy is it to modify to work in different environments?	Flexibility (FLX)
	How easy is it to upkeep and repair?	Maintainability (MNT)
	How easy is it to expand or upgrade its capabilities?	Scalability (SCL)
	How easy is it to show it performs its functions?	Verifiability (VER)
Transition How easy is it to adapt to changes in the technical environment?	How easy is it to interface with another system?	Interoperability (IOP)
	How easy is it to transport?	Portability (POR)
	How easy is it to convert for use in another system?	Reusability (REU)



- ◆ **Efficiency**—how well the software system handles capacity, throughput, and response time.
- ◆ **Integrity**—how well the data are maintained by the software system in terms of accuracy, authenticity, and without corruption.
- ◆ **Reliability**—how well the software system consistently performs the specified functions without failure.
- ◆ **Survivability**—how well the software system continues to function and recovers in the presence of a system failure.
- ◆ **Usability**—how easily the user is able to learn, operate, prepare inputs, and interpret outputs through interaction with a software system.

The ***revision*** group describes the user needs for a system that is easy to correct when errors occur, and is easy to add on new functions. The revision group comprises the following requirement categories:

- ◆ **Flexibility**—how easily the software can be modified to adapt to different environments, configurations, and user expectations.
- ◆ **Maintainability**—how easily faults in the software system can be found and fixed.
- ◆ **Scalability**—how well the software system is able to expand its processing capabilities upward and outward to support business growth.
- ◆ **Verifiability**—the extent to which tests, analysis, and demonstrations are needed to prove that the software system will function as intended.

The ***transition*** group describes the user needs for ease of adaptation to changes in the technical environment. The transition group includes the following requirement categories:



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

- ◆ **Interoperability**—how well the software system is able to couple or facilitate the interface with other systems.
- ◆ **Portability**—how easily the software system can be transferred from its current hardware or software environment to another environment.
- ◆ **Reusability**—how easily a portion of the software system can be converted for use in another system.

Before moving on to the specifics of each of these groups and their associated nonfunctional requirement categories, I'd like to share my vision for the near future of software requirements.

4.5.2 A Future Classification

I hold out hope that in the future we stop using the term “nonfunctional” and its corresponding trail of confusing definitions. Let’s stop forcing everything into two “buckets” of system-level requirements: functional and nonfunctional.

Perhaps a future-state view of system-level requirements is as illustrated in Figure 4-6. Nonfunctional requirements are extinct! Rejoice!

Figure 4-6 System-Level Requirements (Future Classification)



4.6 CHAPTER SUMMARY

- ◆ Industry challenges contribute to the difficulty of understanding nonfunctional requirements including: no agreed-upon, complete list of nonfunctional characteristics, no industry-accepted definition of the term nonfunctional, and no single, uniform classification of nonfunctional attributes.
- ◆ Nonfunctional requirements are vital to the success of software systems. Yet, there are three main factors that influence the nature of nonfunctional requirements and make it difficult to elicit and define them. One factor is their subjective nature. That is, they are interpreted differently by different people and organizations. Relativity is another factor, which means that organizations will emphasize those nonfunctional categories that have the most impact on their specific business processes and products. Integration is a third factor that influences the nature of nonfunctional requirements, which often results in trade-offs or compromise during development or implementation.
- ◆ Although there is no uniform classification of nonfunctional requirements, there are several examples of classifications presented. Furthermore, these examples were compared to identify categories that are common among classifications. These common nonfunctional categories form the foundation under which categories were chosen for inclusion in this book.
- ◆ The user-focused nonfunctional classification presented in this book helps an organization combat the subjective, relative, and integrated nature of these requirements. This user-focused approach is based on three needs of the software system user:
 - ◊ **OPERATION REQUIREMENTS.** How well does the system perform for daily use?
 - ◊ **REVISION REQUIREMENTS.** How easy is it to correct errors and add-on functions?
 - ◊ **TRANSITION REQUIREMENTS.** How easy is it to adapt to changes in the technical environment?

4.7 SUGGESTED READING

Applications Strategies for Risk Analysis, by Robert Charette, [Charette, 1990]. Software engineering and the risks to the business are presented in Chapter 3, as well as the ***interacting nature of nonfunctional requirements***.

Non-functional Requirements in Software Engineering, by Lawrence Chung, et al., [Chung, 2000]. This book presents a concentrated explanation of cataloguing methods and ***non-functional requirements*** (NFRs) types using an NFR Framework.

Software Quality Engineering: A Total Technical and Management Approach, by Michael Deutsch and Ronald Willis, [Deutsch, 1988]. This book presents a “Sample Software Quality Requirements Specification,” including numerous examples of nonfunctional requirements that are separated by user concern. Chapter 3 describes the ***classification of nonfunctional requirements by user needs and concerns***.

Software Quality Metrics Enhancements, Volume 1, by James McCall and Mike Matsumoto, [McCall, 1980]. This research report presents a ***software quality framework*** consisting of ***quality factors***, criterion, and metrics.

Specifying Software Quality Requirements with Metrics, by Steven E. Keller, Laurence G. Kahn, and Roger B. Panara, [Keller, 1990]. This paper presents the Rome Air Development Center’s ***classification of software quality*** consumer-oriented attributes.





5 OPERATION REQUIREMENTS

IN THIS CHAPTER:

5.1 Access Security (ACS)	127
5.2 Availability (AVL).....	140
5.3 Efficiency (EFC)	152
5.4 Integrity (INT)	165
5.5 Reliability (REL).....	181
5.6 Survivability (SRV).....	194
5.7 Usability (USE).....	206
5.8 Suggested Reading.....	223

HOW WELL DOES THE SYSTEM PERFORM FOR DAILY USE?

Operation requirements define how well the software system performs for daily use. While functional requirements describe what tasks the system is to perform, the operation requirements describe how well the system performs the tasks. Generally, these requirements are of greatest concern to the user who comes into contact with the software system by using the functionality. Stated simply, the user's view of *automation* is a software system that either does the work for the user, or helps the user do the work better, faster, and cheaper.



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

The operation group of nonfunctional categories encompasses the concerns of the user during normal operation of the system. Operation requirements answer the user's concerns for the following software qualities:

- ◆ **ACCESS SECURITY.** How well is the system guarded against unauthorized access?
- ◆ **AVAILABILITY.** How dependable is the system during normal operating times?
- ◆ **EFFICIENCY.** How fast, how many, and how well does it respond?
- ◆ **INTEGRITY.** How accurate and authentic are the data?
- ◆ **RELIABILITY.** How immune is the system to failure?
- ◆ **SURVIVABILITY.** How resilient is the system from failure?
- ◆ **USABILITY.** How easy is it to learn and operate the system?

In the sequence listed above, this chapter presents the following for each operation category: definition, brief discussion, examples of requirements, and suggested elicitation questions. The introduction to Part Two explains the anatomy of a category, and provides guidance for how to use the suggested questions.



5.1 ACCESS SECURITY (ACS)

5.1 ACS

USER CONCERN: How well is the system safeguarded against unauthorized access?

RELATED CATEGORIES: Access Control, Authenticity, Authorization, Integrity

5.1.1 Access Security Definition

Access security is the extent to which the system is safeguarded against deliberate and intrusive faults from internal and external sources.

5.1.2 Access Security Discussion

Access security requirements protect the assets of an organization just as one would safeguard money or other forms of property. Trust is an often overlooked corporate asset; users and customers need to trust that the business will secure their private information. The business processes must provide confidentiality, privacy, and authenticity.

It is desirable to make a system ‘fool resistant.’
It is not possible to make it ‘foolproof,’ since they are always improving the quality of fools.
—David Hay, author of *Requirements Analysis: From Business Views to Architecture* [Hay, 2003]

When eliciting access security requirements, consider the following aspects:

- ♦ **USER REGISTRATION**—establishing new user accounts.
 - ◊ **Self-registration.** When users register themselves, anything entered must be treated as suspicious. Users can easily enter false information or enter



information about someone else (identity fraud). Identify the standard for authenticating the user-entered information.

- ◊ **Password format.** Setting password policies requires trade-off considerations, such as how many characters can be easily remembered by a typical user versus how easily passwords might be cracked (easily figured out by an unauthorized user).
 - ◊ **Changing passwords.** Policies for changing passwords also require trade-off considerations. For instance, should users be forced to periodically change passwords? Should there be restrictions on what the password is changed to? Can a password used previously be reused?
 - ◊ **De-registration of users.** If users can self-register, they are typically allowed to self-deregister. If self-registration isn't allowed, then someone must have authority and procedures to deregister users.
 - ◊ **User data protection.** With identity theft on the rise, users are sensitive to what data they are willing to give. The data must be protected, and the system must be in compliance with federal and state privacy laws.
 - ◊ **One-time or one-day users.** Does the organization have a need to accommodate visitors? What about special events?
-
- ♦ **USER AUTHORIZATION**—giving users permission to use system functions.
 - ◊ **Delegating authority.** Some business situations are conducive to delegating authority. For instance, the user who delegates (delegator) should be able to control the level of delegated authority. For example, an executive might have an administrative assistant who needs to manage the executive's calendar and email.
 - ◊ **Inaccessible functions.** It is common to build systems that hide or do not present functions to a specific user who does not have authority to perform such functions.



- ◊ **Authorization based on special events or conditions.** Some events or conditions might require extra authorization. For instance, insurance claim checks issued for more than \$250,000 require a second signature from a user who is a vice-president level or above.
- ◆ **USER AUTHENTICATION**—getting proof that users are who they claim to be.
 - ◊ **Ending user sessions.** Limiting the duration of a user session is recommended. From time to time, users must prove they are really them. For example, employees might be required to re-authenticate every 8 hours.
 - ◊ **User forgets password.** People are human and will forget their passwords. The system must have a way for users to re-authenticate themselves, such as a “secret phrase” or the users must answer a series of security questions.
 - ◊ **De-authentication (user logs off).** Users must have assurance that they are completely logged out. This prevents others from sitting in their seats and performing functions as if done by the users previously signed in.
 - ◊ **Re-authentication (user modifies user information).** When users change the information in their profiles, it is necessary to make notification and authenticate the information.
 - ◊ **User absence or inactivity.** It might be necessary to have users prove from time to time that it is really them. Users could walk away, leaving their session unattended. A user signed on to a web site might go to another site without logging off.
 - ◊ **Blocking or unblocking users.** Blocking users means that they are prevented from logging in to the system, for instance, if users make several invalid attempts to enter their password. If users can be blocked, there must be a means to unblock them after adequate re-authentication.
 - ◊ **Viewing session usage.** Storing information about user session usage can help to identify breaches in security (even if after the fact). Session information usually includes session start and stop date/time, terminal or workstation used, authentication method, and cause of session termination.



5.1.3 Access Security Requirement Examples

- a) Employees shall be forced to change their password the next time they log in if they have not changed it within the length of time established as “password expiration duration.”
- b) Users must change the initially assigned login authentication information (password) immediately after the first successful login. The initial password may never be reused.
- c) The payroll system shall ensure that the employee salary data can be accessed only by authorized users. The payroll system shall distinguish between authorized and non-authorized users.
- d) Employees shall not be allowed to update their own salary information, and any such attempt shall be reported to the security administrator.
- e) Only holders of current security clearance can enter the national headquarters building.
- f) The access permissions for system data may only be changed by the system’s data administrator.
- g) Passwords shall never be viewable at the point of entry or at any other time.
- h) Each unsuccessful attempt by a user to access an item of data shall be recorded on an audit trail.
- i) Users shall receive notification of profile changes via preferred communication method of record when profile information is modified.



5.1.4 Access Security Suggested Questions

5.1 ACS

DATA (WHAT?)

ID	Access Security Suggested Questions	Ask this to:
ACS1s	What information must be guarded against unauthorized update or tampering?	<i>identify data security needs.</i>
ACS2s	What information must be kept confidential?	<i>identify data security needs.</i>
ACS3s	What information is considered “secret”?	<i>identify data security needs.</i>
ACS4s	What information must be guarded against unauthorized disclosure?	<i>identify data security needs.</i>
ACS5s	What would be the impact to the user/customer if data or processing were lost due to a security breach?	<i>identify data security needs.</i>
ACS6s	What would be the impact to the company if data or processing were lost due to a security breach?	<i>identify data security needs.</i>
ACS7s	What public data are accessed by the application?	<i>identify external sources.</i>
ACS8s	How secure are infrequently used data?	<i>identify potential gaps.</i>
ACS9s	How secure are infrequently used hardware/software?	<i>identify potential gaps.</i>
ACS10s	What conditions are associated with authorization?	<i>identify relationships between events and access.</i>
ACS11s	What delegation rights exist to cover periods of absence by managers or administrators of security?	<i>identify role responsibilities.</i>
ACS12s	What user actions require approval?	<i>identify security events and conditions.</i>
ACS13s	Describe the different security levels used.	<i>identify security events and conditions.</i>
ACS14s	What are the conditions to deny user/customer access?	<i>identify security events and conditions.</i>
ACS15s	What criteria must authentication (for example, password) satisfy to be acceptable?	<i>identify security information.</i>
ACS16s	What user session information should be stored?	<i>identify security information.</i>
ACS17s	What are the sources of external data used by the application?	<i>identify types of authentication and verification.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.1 ACS

DATA (WHAT?) (continued)

ID	Access Security Suggested Questions	Ask this to:
ACS18R	What safeguards must be put in place for operational (production or run-time operation) information?	<i>identify procedures.</i>
ACS19R	What safeguards must be put in place for developmental (development stages) information?	<i>identify security events and conditions.</i>
ACS20R	What safeguards must be put in place for information that resides internally?	<i>identify security events and conditions.</i>
ACS21R	What user session information is stored currently?	<i>identify security information.</i>
ACS22R	What supplemental authentication information is needed?	<i>identify security information.</i>
ACS23R	What is expected of the database system when the application is no longer in control of the data?	<i>identify security processes and procedures.</i>
ACS24B	What else should I be asking about access security "data"?	<i>uncover additional requirements.</i>

ROLES (WHO?)

ID	Access Security Suggested Questions	Ask this to:
ACS25s	What access privileges are permitted?	<i>clarify authorization levels.</i>
ACS26s	What level of access is made available without user/customer authorization and authentication?	<i>clarify authorization levels.</i>
ACS27s	Who/what provides inputs to the system?	<i>differentiate the user classes.</i>
ACS28s	What is the unique identifier for each user class?	<i>differentiate the user classes.</i>
ACS29s	What makes the user classes different?	<i>differentiate the user classes.</i>
ACS30s	Who will write security procedures and documentation?	<i>identify additional stakeholders.</i>
ACS31s	What internal users interface with the system?	<i>identify additional user classes.</i>
ACS32s	What external users interface with the system?	<i>identify additional user classes.</i>
ACS33s	How many members of each user class are estimated to use the system each month of the first year the system is operational?	<i>identify approximate size of the user population.</i>
ACS34s	Who/what receives outputs from this system?	<i>identify role responsibilities.</i>



ROLES (WHO?) (continued)

ID	Access Security Suggested Questions	Ask this to:
ACS35s	Who will want to monitor usage?	<i>identify role responsibilities.</i>
ACS36s	Who gives authorization and under what circumstances?	<i>identify role responsibilities.</i>
ACS37s	Who has the authority to grant access to the system function?	<i>identify role responsibilities.</i>
ACS38s	Who is ultimately responsible for each system function?	<i>identify role responsibilities.</i>
ACS39s	Who should be notified when unauthorized access is recognized?	<i>identify role responsibilities.</i>
ACS40s	How many levels of user/customer access are necessary?	<i>identify security events and conditions.</i>
ACS41s	What security concerns do the customers have that must be addressed?	<i>identify security events and conditions.</i>
ACS42s	How could a user collaborate with another authorized user to gain access?	<i>identify security events and conditions.</i>
ACS43s	What levels of security apply to inactive users?	<i>identify security needs.</i>
ACS44s	What level of security is expected for access to data from outside the application?	<i>identify user classes.</i>
ACS45s	How should actions requiring levels of approval be managed?	<i>identify user classes.</i>
ACS46s	How are user classes determined?	<i>identify user classes.</i>
ACS47s	What kinds of user classes are impacted by security?	<i>identify user classes.</i>
ACS48s	Who/what should not be interacting with the system?	<i>identify user classes.</i>
ACS49s	What functions does each user class need to access?	<i>identify user classes.</i>
ACS50R	What security clearance level is required of the individuals who are constructing, enhancing, or installing the system?	<i>differentiate the user classes.</i>
ACS51R	How is access authorized?	<i>identify additional user classes.</i>
ACS52R	Who needs access to perform non-routine maintenance or emergency fixes?	<i>identify additional user classes.</i>
ACS53R	Who needs access to perform routine maintenance?	<i>identify additional user classes.</i>
ACS54B	What else should I be asking about access security "roles"?	<i>uncover additional requirements.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.1 ACS

PURPOSE (WHY?)

ID	Access Security Suggested Questions	Ask this to:
ACS55s	What business policies are in place with regard to access security?	<i>correlate business rules and access security needs.</i>
ACS56s	What business policies pertain to access security?	<i>correlate business rules and access security needs.</i>
ACS57s	What business rules apply to access security?	<i>correlate business rules and access security needs.</i>
ACS58s	What business rules exist for privacy of information?	<i>correlate business rules and access security needs.</i>
ACS59s	What privacy policy has the company declared with regard to the data and its use?	<i>correlate business rules and access security needs.</i>
ACS60s	What privacy laws are applicable with regard to the data?	<i>correlate business rules and access security needs.</i>
ACS61s	What local, state, federal, and international laws affect access security?	<i>correlate business rules and access security needs.</i>
ACS62s	What happens when access expires?	<i>correlate business rules and access security needs.</i>
ACS63s	What security audits are required to monitor application security?	<i>identify audit needs.</i>
ACS64s	How often are audits conducted by internal staff? External?	<i>identify audit needs.</i>
ACS65s	What data protection laws affect access security?	<i>identify compliance with regulations.</i>
ACS66s	What expectations do the customers have about information privacy?	<i>identify customer expectations.</i>
ACS67s	What promises and assurances have been made to the customers about access security?	<i>identify customer expectations.</i>
ACS68s	What promises and assurances have been made to the customers about information privacy?	<i>identify customer expectations.</i>
ACS69s	What measures for managing hardcopy security are necessary?	<i>identify procedural controls.</i>
ACS70s	What security measures must be extended beyond basic company security?	<i>identify security events and conditions.</i>
ACS71s	What keeps the users awake at night with regard to access security?	<i>identify security events and conditions.</i>



PURPOSE (WHY?) (continued)

ID	Access Security Suggested Questions	Ask this to:
ACS72s	What access security issues do the users have?	<i>identify security events and conditions.</i>
ACS73s	What will provide the users with the required level of confidence in security?	<i>identify security events and conditions.</i>
ACS74s	What are the access security concerns of the customers?	<i>identify security events and conditions.</i>
ACS75s	What expectations do the customers have about security?	<i>identify security events and conditions.</i>
ACS76s	What must be done to demonstrate sound security to the customers?	<i>identify security events and conditions.</i>
ACS77s	What system failures could cause significant economic damage to the business?	<i>identify security events and conditions.</i>
ACS78s	When do users/customers get blocked from access?	<i>identify security events and conditions.</i>
ACS79s	What other security issues should be addressed?	<i>identify security events and conditions.</i>
ACS80s	What security issues were experienced in the past?	<i>identify security events and conditions.</i>
ACS81s	What system failures could cause some mission to be unaccomplished (mission failure)?	<i>identify security events and conditions.</i>
ACS82s	What breach of security attempts have been thwarted in the past?	<i>identify security events and conditions.</i>
ACS83s	What is the financial cost to the company of a data security breach?	<i>identify security risk.</i>
ACS84s	What is the cost of misuse of the data or product?	<i>identify security risk.</i>
ACS85s	What is the effect on competitive advantage due to misuse?	<i>identify security risk.</i>
ACS86s	What if a competitor obtained any data?	<i>identify security risk.</i>
ACS87s	What is the public perception cost to the company of a data security breach?	<i>identify security risk.</i>
ACS88s	What rules are in place for establishing and managing user/customer authentication?	<i>identify sources of authentication.</i>
ACS89R	What design and development methods are required to ensure the entire application meets security requirements?	<i>identify methodology constraints.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?) (continued)

ID	Access Security Suggested Questions	Ask this to:
ACS90R	How is this project different from other projects with regard to security?	<i>identify security events and conditions.</i>
ACS91R	What measures have been taken to detect access security design flaws?	<i>identify security events and conditions.</i>
ACS92R	What security measures must be taken to protect the environment?	<i>identify security events and conditions.</i>
ACS93R	What hardware and equipment are the users of the system allowed to install?	<i>identify security risk.</i>
ACS94R	What software are the users of the system allowed to install?	<i>identify security risk.</i>
ACS95R	What security concerns are greater/less than those on other projects? Why?	<i>identify security risk.</i>
ACS96B	What else should I be asking about access security “purpose”?	<i>uncover additional requirements.</i>

TIMING (WHEN?)

ID	Access Security Suggested Questions	Ask this to:
ACS97s	What authorized users are authorized for access at all times?	<i>differentiate the user classes.</i>
ACS98s	Under what conditions are more than one approval required?	<i>identify security events and conditions.</i>
ACS99s	Who will be allowed to use the system when it is first installed?	<i>identify security events and conditions.</i>
ACS100s	How soon should the system time-out because of inactivity?	<i>identify security events and conditions.</i>
ACS101s	Specify conditions or circumstances that limit user authorization to access the system.	<i>identify security events and conditions.</i>
ACS102s	What events make the authorization conditional?	<i>identify security events and conditions.</i>
ACS103s	How long shall access be granted?	<i>identify security metrics.</i>
ACS104s	How promptly is approval needed?	<i>identify turnaround expectations.</i>
ACS105s	When should the system provide a user/customer lock session?	<i>identify types of usage.</i>



TIMING (WHEN?) (continued)

ID	Access Security Suggested Questions	Ask this to:
ASC106s	When are the periods of high or low usage by the user classes?	<i>identify usage patterns.</i>
ACS107s	Under what conditions could locked sessions be broken?	<i>identify types of usage.</i>
ACS108s	Specify the functions and data of the system that each user group is authorized to access.	<i>make certain all functions of the system are covered.</i>
ACS109R	When is it possible for any computers to be connected without following all company security policies?	<i>identify security events and conditions.</i>
ACS110R	At what points during use of the application should users re-authenticate themselves?	<i>identify security events and conditions.</i>
ACS111R	What system components must be locked up when not in use?	<i>identify security events and conditions.</i>
ACS112B	What else should I be asking about access security “timing”?	<i>uncover additional requirements.</i>

LOGISTICS (WHERE?)

ID	Access Security Suggested Questions	Ask this to:
ACS113s	How will access security be the same at all business locations?	<i>identify security events and conditions by location.</i>
ACS114s	What are different levels of security at each location where disaster recovery hardware, software, and other business artifacts are stored?	<i>identify security events and conditions by location.</i>
ACS115s	How is access security different for customers accessing information remotely?	<i>identify security events and conditions by location.</i>
ACS116s	What access security is needed for employees that work from home or other remote locations?	<i>identify security events and conditions by location.</i>
ACS117s	What access restrictions are necessary to the work areas where the application will be used?	<i>identify physical constraints.</i>
ACS118s	What security measures must be taken when accessing the application using public computers?	<i>identify potential holes.</i>
ACS119s	What resources are needed to secure each location?	<i>identify resources needed.</i>
ACS120s	How does access depend on the time of day or location of the user at the time of access?	<i>identify security dependencies.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.1 ACS

LOGISTICS (WHERE?) (continued)

ID	Access Security Suggested Questions	Ask this to:
ACS121R	What reasons are there for this application to run on computers that are not connected to the Internet?	<i>identify location needs.</i>
ACS122R	Where is information from the vendors of supplies for this system maintained?	<i>identify regulation compliance.</i>
ACS123R	Where is regulatory information maintained?	<i>identify regulations.</i>
ACS124R	What hardware/software is needed to secure each location?	<i>identify resources needed.</i>
ACS125R	What security measures are necessary as interfaces pass control through the system?	<i>identify security events and conditions.</i>
ACS126B	What else should I be asking about access security "logistics"?	<i>uncover additional requirements.</i>

PROCESS (How?)

ID	Access Security Suggested Questions	Ask this to:
ACS127s	What is the process flow of hardcopy from receipt to destruction?	<i>identify hardcopy life-cycle limitations.</i>
ACS128s	After denying access, what steps should be taken when a security breach is detected?	<i>identify persistent attack.</i>
ACS129s	How is access reset?	<i>identify recovery.</i>
ACS130s	What should be done when unauthorized access is recognized?	<i>identify responses.</i>
ACS131s	What procedures need to be in place to safely dispose of confidential documents, files, electronic storage devices, and other media?	<i>identify security processes and procedures.</i>
ACS132s	What happens when an approver denies approval?	<i>identify security processes and procedures.</i>
ACS133s	How do users provide identification (indicate to the system who they are)?	<i>identify security processes and procedures.</i>
ACS134s	What metrics of security need to be put in place?	<i>identify security processes and procedures.</i>
ACS135s	How do users provide authentication (indicate to the system that users are in fact who they claim to be)?	<i>identify security processes and procedures.</i>
ACS136s	If multiple means of authentication are offered, which one is the preferred or primary method?	<i>prioritize authentication methods.</i>



PROCESS (HOW?) (continued)

ID	Access Security Suggested Questions	Ask this to:
ACS137R	What different security measures must be developed?	<i>estimate implementation effort.</i>
ACS138R	How are users/customers authenticated?	<i>identify authentication procedures.</i>
ACS139R	How do the users purge confidential information?	<i>identify extended process.</i>
ACS140R	What routine audits are performed to detect improper usage, either by authorized or non-authorized users?	<i>identify inspection criteria.</i>
ACS141R	What integrity checks prevent unintentional misuse by authorized users?	<i>identify integrity checks.</i>
ACS142R	What security is in place to guard against abnormal events and conditions?	<i>identify integrity checks.</i>
ACS143R	How is hardcopy security monitored?	<i>identify physical controls.</i>
ACS144R	What means of authentication should be offered?	<i>identify possible methods of authentication.</i>
ACS145R	What security conventions apply to the development stages? Testing stages?	<i>identify security events and conditions.</i>
ACS146R	How do the users dispose of unused software? Hardware?	<i>identify security processes and procedures.</i>
ACS147R	How must the security measures be implemented?	<i>identify security processes and procedures.</i>
ACS148R	How do new users/customers self-enroll?	<i>identify security processes and procedures.</i>
ACS149R	How are users/customers removed from the system?	<i>identify security processes and procedures.</i>
ACS150R	How will this application be used on computers connected to a network or the Internet?	<i>identify security processes and procedures.</i>
ACS151R	What additional precautions should be taken with the hardware/software of the system?	<i>identify security processes and procedures.</i>
ACS152R	How can the system recognize legitimate users/customers?	<i>identify security processes and procedures.</i>
ACS153R	How do new users/customers get added to the system?	<i>identify security processes and procedures.</i>
ACS154R	What Quality Assurance (QA) measures must be followed to ensure proper security of the system?	<i>identify security processes and procedures.</i>
ACS155B	What else should I be asking about access security "processes"?	<i>uncover additional requirements.</i>



5.2 AVAILABILITY (AVL)

5.2 AVL

USER CONCERN: How dependable is the system during normal operating times?

RELATED CATEGORIES: Accessibility, Dependability, Maintainability, Reliability

5.2.1 Availability Definition

Availability is the degree to which users can depend on the system to be up (able to function) during “normal operating times.”

5.2.2 Availability Discussion

Availability requirements describe:

- (1) When the system is expected to be available for use. An enterprise should establish its “normal operating times.” This is also referred to as the “availability window.”
- (2) What constitutes an acceptable amount of outage time or period of time when the system can be unavailable to users. This outage time, also referred to as the “unavailability window,” takes into consideration periods of time for scheduled downtime, housekeeping, upgrades, and unexpected failure. The availability requirements should also define what the user can expect when the system is unavailable.

Generally, availability is measured by the probability that a system is operating satisfactorily at any point in time when used under stated conditions. Availability is usually described as a percentage as shown in some of the examples on page 142.



When eliciting availability requirements, consider the following aspects:

- ◆ **DOWNTIME IMPACT ON THE BUSINESS.** If the system is not available during normal operation, what is the disruption to business? Is a high degree of availability critical or just nice to have?
- ◆ **PARTIAL AVAILABILITY IMPACT ON THE BUSINESS.** What parts are critical to business continuity? What is the acceptable reduced level of service that the users will tolerate? What business components have an acceptable workaround?
- ◆ **TRANSPARENT UNAVAILABILITY.** What housekeeping tasks must be done during normal operation without user awareness? What degree of degraded performance is tolerable, if any?
- ◆ **MINIMIZING UNAVAILABILITY** (reducing downtime). Some might view these considerations to be reliability or maintainability or both, and that's okay with me as long as they are addressed somewhere. I've included them here because they often affect system availability.
 - ◊ **Frequency and duration of maintenance.** Many companies run an “end-of-day” maintenance cycle. Does it have to run every day? Could the maintenance be subdivided and spread out across a day? What maintenance routines can run while the system is available to users?
 - ◊ **Frequency and duration of periodic upgrades or releases.** The business nature of a particular upgrade can drive the timing of an upgrade. For example, an urgent production fix might mandate that an off-scheduled upgrade occur. It is common to execute system upgrades on a weekend when business is closed. Unfortunately, usually due to poor planning, I have seen upgrades last the entire weekend. Also included in the decision is business impact, users, and inter-related business processes.
 - ◊ **Frequency and duration of unexpected system failures.** Failures can be due to accidents or deliberate acts. A quality system must address ways to prevent both types of failure. When the system cannot prevent failure, maintenance procedures must fix it as quickly as possible.

5.2 AVL



5.2.3 Availability Requirement Examples

- a) The Online Payment System shall be available for use between the hours of 6:00 a.m. and 11:00 p.m. CST.
- b) The Online Payment System shall achieve 100 hours MTBF (mean time between failure).
- c) The CIF system shall achieve 99.5% up time.
- d) The mortgage amortization schedule shall be available to the customer within 15 seconds for 95% of the times that it is requested. The remaining times it will be available within 20 seconds.
- e) The Automated Teller Machine shall be at least 99.0 percent available on weekdays between 6:00 a.m. and 11:00 p.m. local time. The machine shall be at least 99.95 percent available on weekdays between 4:00 p.m. and 6:00 p.m. local time.
- f) Unless the system is non-operational, the system shall present a user with notification informing them that the system is unavailable.
- g) A new installation of the system shall be available for first-time use within 24 hours of the start of the install.
- h) The online registration system shall permit backing up of the registration database while other registration activities are going on. (It is estimated that this requirement reduces duration for which the online registration system would be unavailable to students for maintenance by 15 minutes each calendar day.)



5.2.4 Availability Suggested Questions

DATA (WHAT?)

5.2 AVL

ID	Availability Suggested Questions	Ask this to:
AVL1s	What notification is needed when the system is completely down?	<i>identify availability data.</i>
AVL2s	What notification is needed when the system is partially down?	<i>identify availability data.</i>
AVL3s	What processes use data for business that can be recovered following a system outage?	<i>identify availability data.</i>
AVL4s	What processes use data for business that cannot be recovered following a system outage?	<i>identify availability data.</i>
AVL5s	What challenges do users have in inputting information or materials?	<i>identify barriers.</i>
AVL6s	What challenges do users have in retrieving information or materials?	<i>identify barriers.</i>
AVL7s	When are multiple means of notification used?	<i>identify notifications.</i>
AVL8s	What means of notification should be used?	<i>identify notifications.</i>
AVL9s	Which means of notification should be used by type of system outage?	<i>identify notifications.</i>
AVL10s	When the system is unavailable, what notification should be given to users attempting to access the system?	<i>identify notifications.</i>
AVL11s	What advanced warning to users of planned outage is expected?	<i>identify notifications.</i>
AVL12R	What information or materials are archived and stored off-site?	<i>identify availability data.</i>
AVL13R	What information or materials are difficult to retrieve?	<i>identify availability data.</i>
AVL14R	How does notification depend on type of error?	<i>identify notifications.</i>
AVL15R	After an outage, how should users be notified of system availability?	<i>identify notifications.</i>
AVL16B	What else should I be asking about availability "data"?	<i>uncover additional requirements.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

ROLES (WHO?)

5.2 AVL

ID	Availability Suggested Questions	Ask this to:
AVL17s	What emergency access should be extended to each user class?	<i>identify exceptions.</i>
AVL18s	What are the expectations on system availability?	<i>identify expectations.</i>
AVL19s	What emergency access is expected for each user class?	<i>identify forms of access.</i>
AVL20s	When should multiple people be notified?	<i>identify role priority.</i>
AVL21s	What users should have limited availability (access) to the system?	<i>identify role rankings.</i>
AVL22s	Who should receive notification when there are system failures?	<i>identify role responsibilities.</i>
AVL23s	Which people need to be notified when there is a serious error?	<i>identify role responsibilities.</i>
AVL24s	Who is authorized to access what data when the system is unavailable?	<i>identify role responsibilities.</i>
AVL25s	How should acknowledgment from someone taking responsibility for the problem be communicated?	<i>identify role responsibilities.</i>
AVL26s	What is the typical number of users accessing the system simultaneously?	<i>identify usage.</i>
AVL27s	What is the greatest number of simultaneous users during peak processing? Anticipated future usage?	<i>identify usage.</i>
AVL28R	Who performs routine maintenance?	<i>identify role responsibilities.</i>
AVL29R	Who gets called when the system is down?	<i>identify role responsibilities.</i>
AVL30R	What percent of problems reported to the first level support group (for example, help desk) should be handled by the first level support group?	<i>identify role responsibilities.</i>
AVL31R	Who supports the system during "after hours" or non-normal operation hours?	<i>identify role responsibilities.</i>
AVL32B	What else should I be asking about availability "roles"?	<i>uncover additional requirements.</i>



PURPOSE (WHY?)

ID	Availability Suggested Questions	Ask this to:
AVL33s	How would users describe availability?	<i>clarify terminology.</i>
AVL34s	Define availability in terms of functions that are performed.	<i>clarify terminology.</i>
AVL35s	Define availability in terms of users that perform the work.	<i>clarify terminology.</i>
AVL36s	What are the “normal” business hours of operation?	<i>establish business rules.</i>
AVL37s	What causes the most pain when the system is not available?	<i>identify effects.</i>
AVL38s	How long can the system be down before there is permanent business loss?	<i>identify assessments.</i>
AVL39s	What is an acceptable amount of time to detect an outage?	<i>identify detection timeframes.</i>
AVL40s	When should emergency access be extended to bypass normal access restrictions?	<i>identify events.</i>
AVL41s	What are the expectations of the availability of the system?	<i>identify expectations.</i>
AVL42s	What are the associated risks to conducting business “as usual” if the system/application is unavailable?	<i>identify limitations in alternatives.</i>
AVL43s	What is the acceptable effort required of users/customers to gain access to data?	<i>identify possible trade-offs.</i>
AVL44s	How much is the business (sponsor) willing to invest to reduce the chance of downtime or failure?	<i>identify possible trade-offs.</i>
AVL45s	What problems with availability have been experienced in the past?	<i>identify problems and lessons learned.</i>
AVL46s	What are the greatest concerns about availability?	<i>identify risk.</i>
AVL47s	What is the cost to the business for system downtime?	<i>identify risk.</i>
AVL48s	What constitutes a serious error?	<i>identify risk.</i>
AVL49s	How much permanent business loss is acceptable?	<i>identify risk.</i>
AVL50s	If the system is not available, how is the business affected?	<i>identify risk.</i>
AVL51s	What is an acceptable effect on system availability of background activity?	<i>identify tolerance metrics.</i>

5.2 AVL



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.2 AVL

PURPOSE (WHY?) (continued)

ID	Availability Suggested Questions	Ask this to:
AVL52s	What cost tolerance is acceptable to maintain the required level of availability?	<i>identify tolerance metrics.</i>
AVL53s	How does routine maintenance hinder or delay productivity?	<i>identify various types of maintenance.</i>
AVL54s	What processes are considered to be absolutely mission critical to the business?	<i>prioritize processes.</i>
AVL55s	What processes must be the last to fail or be unavailable?	<i>prioritize processes.</i>
AVL56R	What level of limited availability in a degraded mode is acceptable?	<i>identify graduations.</i>
AVL57R	What current availability issues must be avoided by this project?	<i>identify lessons learned.</i>
AVL58R	What level of performance constitutes available?	<i>identify performance levels.</i>
AVL59R	How do security devices or procedures hinder or delay access of information or materials?	<i>identify possible trade-offs.</i>
AVL60R	What hardware components are common sources of availability issues?	<i>identify sources of issues.</i>
AVL61R	What is the timeframe for problem resolution by the first-level support group?	<i>identify tolerance metrics.</i>
AVL62B	What else should I be asking about availability "purpose"?	<i>uncover additional requirements.</i>



TIMING (WHEN?)

ID	Availability Suggested Questions	Ask this to:
AVL63s	How does notification depend on time of day?	<i>identify graduations in notifications.</i>
AVL64s	What is the fallback position for partial availability?	<i>identify processing sequence.</i>
AVL65s	When can upgrades be scheduled?	<i>identify schedules.</i>
AVL66s	What special processes are executed for trade-shows, customer events or promotions?	<i>identify time-triggered events.</i>
AVL67s	What days during the week should the system be available?	<i>identify time-triggered events.</i>
AVL68s	What special processes run for week-end, month-end, and year-end?	<i>identify time-triggered events.</i>
AVL69s	What time-sensitive processing occurs? (daily, monthly, quarterly, annually, month-end, quarter-end, or year-end).	<i>identify time-triggered events.</i>
AVL70s	When do the users need access to the information and materials they want?	<i>identify time-triggered events.</i>
AVL71s	What specific periods are imperative to meet business or safety objectives?	<i>identify time-triggered events.</i>
AVL72s	If the system cannot be available “24/7” or “all the time,” when are acceptable periods or conditions for the system to be unavailable?	<i>identify time-triggered events.</i>
AVL73s	How does availability vary by time of day?	<i>identify time-triggered events.</i>
AVL74s	When should the availability limitations be imposed?	<i>identify time-triggered events.</i>
AVL75s	When do the users need to prepare or enter inputs?	<i>identify time-triggered events.</i>
AVL76s	When do the users need to retrieve or get outputs?	<i>identify time-triggered events.</i>
AVL77s	When do the users need to perform their work?	<i>identify time-triggered events.</i>
AVL78s	When are users trying to do their jobs?	<i>identify time-triggered events.</i>
AVL79s	When are the most critical business periods? (time of day; busy season)	<i>identify time-triggered events.</i>
AVL80s	What time frame should be used when tracking the percentage of availability over an extended period? (day, week, month, quarter, year, or longer)	<i>identify tolerance metrics.</i>
AVL81s	When are periods of planned/expected peak usage?	<i>identify tolerance metrics.</i>
AVL82s	What is the expected availability during peak hours?	<i>identify tolerance metrics.</i>

5.2 AVL



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.2 AVL

TIMING (WHEN?) (continued)

ID	Availability Suggested Questions	Ask this to:
AVL83s	What is the acceptable availability during peak usage hours?	<i>identify tolerance metrics.</i>
AVL84s	What is the acceptable amount of time to react to an outage?	<i>identify tolerance metrics.</i>
AVL85s	When is the best time to perform maintenance?	<i>identify tolerance metrics.</i>
AVL86s	What is the accepted tolerance for system failure or unexpected downtime?	<i>identify tolerance metrics.</i>
AVL87s	What is an acceptable amount of time to correct a problem?	<i>identify tolerance metrics.</i>
AVL88s	What is a reasonable time to wait for retrieval of information or materials stored off-site?	<i>identify tolerance metrics.</i>
AVL89s	What is an acceptable amount of time to perform maintenance?	<i>identify various forms of maintenance.</i>
AVL90s	What activities or tasks are more time critical than others?	<i>prioritize activities.</i>
AVL91s	What time periods must be avoided for upgrades and maintenance?	<i>prioritize activities.</i>
AVL92s	Depending on time of day, which means of notification should be used?	<i>prioritize notifications.</i>
AVL93R	When should correction escalation be invoked?	<i>identify escalation triggers.</i>
AVL94R	When should notification escalation be invoked?	<i>identify escalations.</i>
AVL95B	What else should I be asking about availability "timing"?	<i>uncover additional requirements.</i>



LOGISTICS (WHERE?)

ID	Availability Suggested Questions	Ask this to:
AVL96s	What users access the system in a location that is susceptible to availability issues?	<i>identify availability needs by location.</i>
AVL97s	How does routine maintenance vary by business location?	<i>identify availability needs by location.</i>
AVL98s	How does system availability vary by business location?	<i>identify availability needs by location.</i>
AVL99s	How do users in varying business locations need access to the system differently?	<i>identify availability needs by location.</i>
AVL100s	What business locations must have access to the system?	<i>identify availability needs by location.</i>
AVL101s	What peak periods of usage vary by location?	<i>identify location restrictions.</i>
AVL102R	How does off-site storage hinder or delay productivity?	<i>identify location restrictions.</i>
AVL103R	What time periods must be avoided for upgrades and maintenance by business location?	<i>identify location restrictions.</i>
AVL104R	What processes are performed at each location?	<i>identify relationships between processes and location.</i>
AVL105R	What resources are needed at each business location to prevent downtime?	<i>identify resource needs.</i>
AVL106R	What hardware/software and people resources are needed at each location to perform routine maintenance?	<i>identify resource needs.</i>
AVL107R	How will system usage be monitored by location?	<i>identify usage.</i>
AVL108B	What else should I be asking about availability “logistics”?	<i>uncover additional requirements.</i>

5.2 AVL



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PROCESS (How?)

5.2 AVL

ID	Availability Suggested Questions	Ask this to:
AVL109s	What processes are legally required?	<i>classify processes.</i>
AVL110s	What processes are required by contract obligation?	<i>classify processes.</i>
AVL111s	Which processes will be used by the most customers?	<i>classify processes.</i>
AVL112s	Which processes will be used by the most users?	<i>classify processes.</i>
AVL113s	What processes are most dependent on the system being available?	<i>evaluate processes.</i>
AVL114s	What processes may have diminished availability due to degradation?	<i>evaluate processes.</i>
AVL115s	If the system/application is not available, how is business conducted?	<i>identify alternative processes.</i>
AVL116s	How long are users able to conduct business if the system/application is down?	<i>identify alternative processes.</i>
AVL117s	What processes or procedures are followed to continue doing business due to system malfunction?	<i>identify alternative processes.</i>
AVL118s	What workarounds do the users use if the system is unavailable?	<i>identify alternative processes.</i>
AVL119s	What triggers availability limitations?	<i>identify events.</i>
AVL120s	What special processes run for government purposes?	<i>identify external factors.</i>
AVL121s	What uses of the system are most apt to have availability issues?	<i>identify processes.</i>
AVL122s	What business processes cannot be performed if the system is down?	<i>identify processes.</i>
AVL123s	Which processes are compromised by the loss of related processes?	<i>identify relationships.</i>
AVL124s	What events or conditions may occur that would delay access to inputs/outputs?	<i>identify various events and conditions.</i>
AVL125s	What events or conditions may occur that would prevent access to inputs/outputs?	<i>identify various events or conditions.</i>
AVL126s	Which functions of the system are most critical to the business?	<i>prioritize processes.</i>



PROCESS (HOW?) (continued)

ID	Availability Suggested Questions	Ask this to:
AVL127s	What processes must always be available?	<i>prioritize processes.</i>
AVL128s	Which processes cause the greatest loss of company productivity when unavailable?	<i>prioritize processes.</i>
AVL129s	Which processes produce the most revenue?	<i>prioritize processes.</i>
AVL130s	Which processes produce the least revenue?	<i>prioritize processes.</i>
AVL131s	Which processes manage the most time-sensitive data?	<i>prioritize processes.</i>
AVL132s	Which processes manage the least time-critical data?	<i>prioritize processes.</i>
AVL133s	What processes have the most recoverable data?	<i>prioritize processes.</i>
AVL134s	What processes have the least recoverable data?	<i>prioritize processes.</i>
AVL135R	Which processes are the most frequently used?	<i>evaluate processes.</i>
AVL136R	When availability of a process is lost, which processes take the longest to regain a normal state of productivity?	<i>evaluate processes.</i>
AVL137R	When availability of a process is lost, which processes are the quickest to regain a normal state of productivity?	<i>evaluate processes.</i>
AVL138R	Which processes are dependent on the most system resources?	<i>evaluate processes.</i>
AVL139R	What routine maintenance is performed on the system? When is it performed?	<i>identify alternative processes.</i>
AVL140R	When the system is not available, what alternatives are there?	<i>identify alternative processes.</i>
AVL141R	What means of notification escalation should be used?	<i>identify escalation procedures.</i>
AVL142R	What means of correction escalation should be used?	<i>identify escalation procedures.</i>
AVL143R	What external communications are the source of availability issues?	<i>identify external factors.</i>
AVL144R	How does availability vary by function?	<i>identify processes.</i>
AVL145R	What processes should have availability limitations?	<i>identify restrictions.</i>
AVL146R	How should availability limitations be imposed?	<i>identify triggers.</i>
AVL147B	What else should I be asking about availability "processes"?	<i>uncover additional requirements.</i>

5.2 AVL



5.3 EFFICIENCY (EFC)

5.3 EFC

USER CONCERN:

How fast does the system function? How many can be processed? How well does the system respond?

RELATED CATEGORIES:

Capacity, Performance, Responsiveness, Throughput

5.3.1 Efficiency Definition

Efficiency is the extent to which the software system handles capacity, throughput, and response time.

5.3.2 Efficiency Discussion

Efficiency is defined by some as “workability.” Workability is the raw ability of the system to perform work. Efficiency requirements identify required functionality, indicating the strain or burden placed on current resources. These requirements identify the user needs to perform tasks in a given amount of time. Efficiency requirements express the expectations with regard to response time, throughput, and capacity:

- ◆ ***Response time*** or ***Responsiveness*** is the degree to which the system reacts to a single event or condition. Response time requirements specify how much time it takes the software system to act upon a user input or request.
- ◆ ***Throughput*** is the rate at which the system can perform input and output processing. Generally, throughput deals with how fast inputs can be taken in and how quickly the system can crank out the outputs.



- ◆ **Process capacity** is the ability to process units of work in specified units of time. Process capacity also deals with how many of a particular entity can be processed at the same time.
- ◆ **Storage capacity** is the ability of the system to store units of any defined entity.

When eliciting efficiency requirements, consider the following aspects:

5.3 EFC

- ◆ **RESPONSE TIME.**

- ◊ **Possible actions to improve response time.** Things that can be done to improve response time are dependent on the specifics of each system, and therefore not named here. It is up to the team to identify them.
- ◊ **Ways to measure response time.** What internal monitoring can be done by the system itself? What products and devices are available to monitor the system? What system responses can be watched by hand (monitored by a human)?
- ◊ **Minimizing user frustration while waiting for a response.** A quality system should notify the user of possible wait times, give progress information during the wait, and camouflage the wait if possible. For example, giving the user something to do while they wait can make the wait duration seem to take less time.

- ◆ **THROUGHPUT.**

- ◊ **Ways to monitor throughput.** Throughput measurements can be current-state (those taken to record what's happening right now) or historical (those recorded over a period of time in order to identify peak periods and trends).
- ◊ **Cause of throughput traffic and reasons to limit throughput.** Are there peak processing periods when it might be necessary to limit the number of users accessing the system? Can the business process be altered so that the work load is leveled out? For example, instead of having all employees trying to



“punch out” on the time clock at 5:00 p.m., the quitting times could be staggered every 10 minutes from 4:30 p.m. to 5:30 p.m. (assuming the start time is also adjusted accordingly).

- ◊ **Possible actions to improve throughput.** What work can be done at other times or every other day? It is recommended that background processing run on separate systems.
- ◊ **Ways to organize user inputs and outputs** to optimize system processing of data validation, storage, and retrieval.
- ♦ **PROCESS CAPACITY.**
 - ◊ **Number of simultaneous users.** When might it be necessary to limit the number of users logging in?
 - ◊ **Ways to monitor usage.** What information might be helpful in monitoring user activity? It is generally helpful to track the number of users currently active, as well as the number and class of users on each system. Tracking the number of active users over a period of time can help identify peak processing periods.
 - ◊ **Ways to free up system resources.** The development and system support teams should be on the lookout for ways to reduce system workloads.
 - ◊ **Ways to optimize system resources around patterns of use or relationships.** The development and system support teams should look to relationships of data and resources to anticipate workloads for better resource balance.
- ♦ **STORAGE CAPACITY.**
 - ◊ **Permanent or temporary storage.** Consider data longevity and data archiving needs. What procedures should be taken to prevent the system from getting over cluttered with inactive data?



- ◊ **Rate and frequency of retrieving stored entities.** Historical information can be tracked in order to identify performance metrics over time.
- ◊ **Contingency if storage capacity is exceeded.** Who should be notified? Is there a “warning” level before capacity is exceeded?

5.3 EFC

5.3.3 Efficiency Requirement Examples

- a) At least 20 percent of the processor capacity and storage space available to the system shall be unused at peak load seasonal periods.
- b) The system restart cycle must execute completely in less than 60 seconds.
- c) System shall be able to process a notification in 1 second or less, and up to and including 100 notifications in 15 seconds or less.
- d) The initial system shall be able to handle the entry of orders by customers at a minimum rate of 10 per second.
- e) The system must accommodate 300 simultaneous users or less within the peak load period from 9:00 a.m. to 11:00 a.m. Maximum simultaneous user capacity loading at non-peak periods will be 150.
- f) Any interface between a user and the automated system shall have a maximum response time of two seconds.
- g) Complete report summaries of the current business day's trading shall be available one minute after the end-of-day close of trading.
- h) Routine maintenance that is executed while users are active shall not cause a perceptible increase in response time for any function of more than 5% over the response time when no maintenance process is executing.
- i) The system shall produce a storage capacity warning notification when the 65% capacity threshold is crossed with additional notifications issued thereafter at 5% threshold increments.



5.3.4 Efficiency Suggested Questions

DATA (WHAT?)

5.3 EFC

ID	Efficiency Suggested Questions	Ask this to:
EFC1s	What is the outgoing source of throughput peak volume?	<i>identify information sources.</i>
EFC2s	What is the incoming source of throughput peak volume?	<i>identify information sources.</i>
EFC3s	How much data should be retained for each user?	<i>identify retention needs.</i>
EFC4s	How much data should be retained for each customer?	<i>identify retention needs.</i>
EFC5s	How much data should be retained for each transaction?	<i>identify retention needs.</i>
EFC6s	How much data should be retained for each product?	<i>identify retention needs.</i>
EFC7s	How long should the data be retained?	<i>identify retention needs.</i>
EFC8s	How long should the data be retained for real-time access?	<i>identify retention needs.</i>
EFC9s	When is an acknowledgment before results an expected response?	<i>identify various communications.</i>
EFC10s	Which requests require indicators of progress while processing?	<i>identify various communications.</i>
EFC11s	Which requests require estimates of time remaining before completion?	<i>identify various communications.</i>
EFC12s	What warning thresholds can be set as capacity grows?	<i>identify warnings.</i>
EFC13s	What is the balance between incoming and outgoing throughput?	<i>prioritize information or materials.</i>
EFC14s	What else should I be asking about efficiency "data"?	<i>uncover additional requirements.</i>



ROLES (WHO?)

ID	Efficiency Suggested Questions	Ask this to:
EFC15s	What source can provide an accurate count of existing end users?	<i>identify additional stakeholders.</i>
EFC16s	What source can provide an accurate count of existing customers?	<i>identify additional stakeholders.</i>
EFC17s	What source can provide an accurate count of existing processes?	<i>identify additional stakeholders.</i>
EFC18s	What source can provide an accurate volume count of existing transactions?	<i>identify additional stakeholders.</i>
EFC19s	What are the expectations on system efficiency?	<i>identify expectations.</i>
EFC20s	What are the expectations on system response time?	<i>identify expectations.</i>
EFC21s	What are the expectations on system throughput?	<i>identify expectations.</i>
EFC22s	What are the expectations on system capacity?	<i>identify expectations.</i>
EFC23s	What users should have a need for greater efficiency than others?	<i>identify expectations.</i>
EFC24s	When should multiple people be notified?	<i>identify role responsibilities.</i>
EFC25s	Who should receive notification when there are system failures?	<i>identify role responsibilities.</i>
EFC26s	Which people need to be notified when there is a serious error?	<i>identify role responsibilities.</i>
EFC27s	Who is authorized to perform what functions?	<i>identify role responsibilities.</i>
EFC28s	Who is authorized to access what data?	<i>identify role responsibilities.</i>
EFC29s	Who (role) should receive warning messages as thresholds are exceeded?	<i>identify role responsibilities.</i>
EFC30s	What is the typical number of users accessing the system simultaneously?	<i>identify usage.</i>
EFC31s	What is the greatest number of simultaneous users during peak processing? Anticipated future usage?	<i>identify usage.</i>
EFC32s	What limits to the number of simultaneous users apply?	<i>identify usage.</i>
EFC33s	What forcible eject/reject requirements exist for users?	<i>identify user behavior.</i>
EFC34s	What types of users can be affected by traffic metering?	<i>identify users.</i>
EFC35R	Who performs routine performance tuning?	<i>identify role responsibilities.</i>

5.3 EFC



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.3 EFC

ROLES (WHO?) (continued)

ID	Efficiency Suggested Questions	Ask this to:
EFC36R	Who gets called when the system is not performing as expected?	<i>identify role responsibilities.</i>
EFC37B	Who should receive system performance audit reports?	<i>identify role responsibilities.</i>
EFC38B	What else should I be asking about efficiency "roles"?	<i>uncover additional requirements.</i>

PURPOSE (WHY?)

ID	Efficiency Suggested Questions	Ask this to:
EFC39s	How many things can the system cope with and work on at once?	<i>identify expectations.</i>
EFC40s	How critical is this performance factor?	<i>identify metrics.</i>
EFC41s	What demand spike can be anticipated?	<i>identify metrics.</i>
EFC42s	What response time expectations are there by function?	<i>identify metrics.</i>
EFC43s	What class or category levels of performance can be defined?	<i>identify metrics.</i>
EFC44s	Where did the tolerable response time come from?	<i>identify metrics.</i>
EFC45s	What level of throughput is expected of the system?	<i>identify metrics.</i>
EFC46s	What is the minimum number of events the system must store?	<i>identify metrics.</i>
EFC47s	What performance is measured?	<i>identify metrics.</i>
EFC48s	What timeframe does this performance requirement need to meet?	<i>identify metrics.</i>
EFC49s	What is the maximum acceptable response time end-to-end?	<i>identify metrics.</i>
EFC50s	What percentage of time is the stated response time expected?	<i>identify metrics.</i>
EFC51s	How will volume of activity be measured?	<i>identify metrics.</i>
EFC52s	What volume levels will be measured, and in what timeframes?	<i>identify metrics.</i>



PURPOSE (WHY?) (continued)

ID	Efficiency Suggested Questions	Ask this to:
EFC53s	How can performance be measured?	<i>identify metrics.</i>
EFC54s	When does measurement start and stop?	<i>identify metrics.</i>
EFC55s	How should performance be monitored?	<i>identify metrics.</i>
EFC56s	How much resource should be allocated to monitoring?	<i>identify metrics.</i>
EFC57s	What performance indicators are most important to the business?	<i>identify metrics.</i>
EFC58s	On what basis is this performance goal specified?	<i>identify metrics.</i>
EFC59s	What is a tolerable length of response time?	<i>identify metrics.</i>
EFC60s	Please fill in the following: What is the expected response time in [<i>unit of measure</i>]?	<i>identify metrics.</i>
EFC61s	Please fill in the following: What is the expected throughput in [<i>unit of measure</i>]?	<i>identify metrics.</i>
EFC62s	Please fill in the following: What is the expected capacity in [<i>unit of measure</i>]?	<i>identify metrics.</i>
EFC63s	What is the expected (accepted) average response time?	<i>identify metrics.</i>
EFC64s	What is the expected (accepted) maximum response time?	<i>identify metrics.</i>
EFC65s	What is the expected (accepted) average throughput?	<i>identify metrics.</i>
EFC66s	What is the expected (accepted) minimum throughput?	<i>identify metrics.</i>
EFC67s	What is the expected (accepted) maximum throughput?	<i>identify metrics.</i>
EFC68s	What is the expected speed to complete the task?	<i>identify metrics.</i>
EFC69s	Why is this level of performance needed?	<i>identify metrics.</i>
EFC70s	What is an acceptable mode of operation when the system has been degraded in some manner?	<i>identify performance levels.</i>
EFC71s	At what load level can response begin to degrade?	<i>identify performance levels.</i>
EFC72s	What is acceptable degradation during peak times?	<i>identify performance levels.</i>
EFC73s	What have been barriers to expected performance?	<i>identify performance levels.</i>

5.3 EFC



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.3 EFC

PURPOSE (WHY?) (continued)

ID	Efficiency Suggested Questions	Ask this to:
EFC74s	What is the worst that can happen if the requirement is not met?	<i>identify results (outcomes).</i>
EFC75s	What are risks associated with performance problems?	<i>identify risk.</i>
EFC76s	How critical to the business are the performance requirements?	<i>identify risk.</i>
EFC77s	How much is acceptable to pay for this level of performance?	<i>identify value.</i>
EFC78s	What is an acceptable cost of monitoring?	<i>identify value.</i>
EFC79R	What is the capacity of the system to store events of a particular type?	<i>identify events.</i>
EFC80R	What functions are candidates for metering response time?	<i>identify events.</i>
EFC81R	What are the predictors of high volume activity?	<i>identify events.</i>
EFC82R	What circumstances or conditions must be met from the operating environment?	<i>identify events.</i>
EFC83R	What efficiency conditions might exist from vendor hardware?	<i>identify resource needs.</i>
EFC84R	What efficiency conditions might exist from vendor software?	<i>identify resource needs.</i>
EFC85B	What are the known constraints/restrictions imposed by the system?	<i>identify possible trade-offs.</i>
EFC86B	What time-accuracy trade-offs have been considered?	<i>identify possible trade-offs.</i>
EFC87B	What time-space trade-offs have been considered?	<i>identify possible trade-offs.</i>
EFC88B	What is the trade-off between efficiency and scalability?	<i>identify possible trade-offs.</i>
EFC89B	What is the trade-off between efficiency and maintainability?	<i>identify possible trade-offs.</i>
EFC90B	What is the trade-off between efficiency and reliability?	<i>identify possible trade-offs.</i>
EFC91B	What is the trade-off between efficiency and flexibility?	<i>identify possible trade-offs.</i>
EFC92B	What else should I be asking about efficiency "purpose"?	<i>uncover additional requirements.</i>



TIMING (WHEN?)

ID	Efficiency Suggested Questions	Ask this to:
EFC93s	When should inactive data be removed?	<i>identify data characteristics.</i>
EFC94s	How often should performance be measured?	<i>identify metrics.</i>
EFC95s	When should performance be measured?	<i>identify metrics.</i>
EFC96s	What is typical off-peak throughput? Peak throughput?	<i>identify metrics.</i>
EFC97s	When during the week, month, quarter, and year will peak times occur?	<i>identify system load cycles.</i>
EFC98s	What is the expected duration of peak use activity periods?	<i>identify system load durations.</i>
EFC99s	What time of day will peak usage occur?	<i>identify system load time periods.</i>
EFC100s	Which requests have maximum response time limits?	<i>identify time limitations.</i>
EFC101s	Which requests have minimum response time limits?	<i>identify time limitations.</i>
EFC102s	Which requests have maximum sequential response time limits? (The request is followed by sequentially timed responses.)	<i>identify time limitations.</i>
EFC103s	Which requests have minimum sequential response time limits?	<i>identify time limitations.</i>
EFC104s	Which requests are sequential with maximum time limits? (For example, user must enter password within <i>x time</i> of entering user identification.)	<i>identify time limitations.</i>
EFC105s	Which requests are sequential with minimum time limits? (For example, user cannot enter password within <i>x time</i> of entering user identification.)	<i>identify time limitations.</i>
EFC106s	Which requests require completion by the user within a set period of time? (For example, entry of data must complete from first to last within <i>x time</i> .)	<i>identify time limitations.</i>
EFC107s	Which requests require completion by the user after a set period of time? (For example, entry of menu choice cannot occur until <i>x time</i> has lapsed from menu display.)	<i>identify time limitations.</i>
EFC108s	What timeframe is this capacity expected?	<i>identify time limitations.</i>
EFC109s	How do peak activity periods compare to normal activity periods of use?	<i>identify various time periods.</i>
EFC110B	What should the balance be between primary and secondary processes?	<i>identify possible trade-offs.</i>
EFC111B	What are the most important activities to process during peak load times?	<i>identify possible trade-offs.</i>

5.3 EFC



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.3 EFC

TIMING (WHEN?) (continued)

ID	Efficiency Suggested Questions	Ask this to:
EFC112B	What activities can be reduced during peak load times?	<i>identify possible trade-offs.</i>
EFC113B	What else should I be asking about efficiency “timing”?	<i>uncover additional requirements.</i>

LOGISTICS (WHERE?)

ID	Efficiency Suggested Questions	Ask this to:
EFC114s	What users access the system in a location that is susceptible to efficiency issues?	<i>identify efficiency needs by location.</i>
EFC115s	How does routine performance tuning vary by business location?	<i>identify efficiency needs by location.</i>
EFC116s	How does system efficiency vary by business location?	<i>identify efficiency needs by location.</i>
EFC117s	What users in varying business locations need access to the system differently?	<i>identify efficiency needs by location.</i>
EFC118s	What business locations interface with the system?	<i>identify efficiency needs by location.</i>
EFC119s	What peak periods of usage vary by location?	<i>identify efficiency needs by location.</i>
EFC120s	Where is the greatest concentration of users?	<i>identify locations.</i>
EFC121s	How widely spread are the system users?	<i>identify user locations.</i>
EFC122s	What users spread across time zones?	<i>identify user locations.</i>
EFC123s	What users spread across the world?	<i>identify user locations.</i>
EFC124s	What part of the system do each of the performance requirements pertain?	<i>identify various components.</i>
EFC125R	How does off-site storage capacity hurt or help performance?	<i>identify location restrictions.</i>
EFC126R	What time periods must be avoided for performance tuning by business location?	<i>identify location restrictions.</i>
EFC127R	What processes are performed at each location?	<i>identify relationships between processes and location.</i>



LOGISTICS (WHERE?) (continued)

ID	Efficiency Suggested Questions	Ask this to:
EFC128R	What resources are needed at each business location to increase efficiency performance?	<i>identify resource needs.</i>
EFC129R	What hardware/software and people resources are needed at each location to execute performance tuning?	<i>identify resource needs.</i>
EFC130R	How will system throughput be monitored by location?	<i>identify usage.</i>
EFC131R	How will system capacity be monitored by location?	<i>identify usage.</i>
EFC132R	How will system response time be monitored by location?	<i>identify usage.</i>
EFC133B	What else should I be asking about efficiency “logistics”?	<i>uncover additional requirements.</i>

5.3 EFC

PROCESS (How?)

ID	Efficiency Suggested Questions	Ask this to:
EFC134s	What are acceptable response times during exception processing?	<i>identify exceptions.</i>
EFC135s	What is the anticipated growth?	<i>identify metrics.</i>
EFC136s	What peak load is anticipated?	<i>identify metrics.</i>
EFC137s	What time-critical processes are expected?	<i>identify processes.</i>
EFC138s	What makes these processes time critical?	<i>identify processes.</i>
EFC139s	What happens to the system during the time-critical processes when too much time is taken?	<i>identify processes.</i>
EFC140s	What processes include complex calculations?	<i>identify processes.</i>
EFC141s	What processes require large volumes of data?	<i>identify processes.</i>
EFC142s	What part of the system does this performance target apply to?	<i>identify processes.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PROCESS (HOW?) (continued)

5.3 EFC

ID	Efficiency Suggested Questions	Ask this to:
EFC143s	What is the rate at which the system can process things?	<i>identify processing rates.</i>
EFC144s	How shall the system respond when warning thresholds are hit?	<i>identify system responses.</i>
EFC145s	What messages should be sent when warning thresholds are hit?	<i>identify system responses.</i>
EFC146s	What response is expected when stated capacities are exceeded?	<i>identify system responses.</i>
EFC147s	How long does it take the system to satisfy user requests?	<i>identify system responses.</i>
EFC148R	What exception processes must be considered?	<i>identify exceptions.</i>
EFC149R	When can traffic be metered for limiting access?	<i>identify metrics.</i>
EFC150R	What development techniques may be used to improve retrieval time?	<i>identify possible process improvement areas.</i>
EFC151R	What processing considerations (encryption, protocols, message sets) must be included in the application?	<i>identify processes.</i>
EFC152R	What operations are dependent upon vendor software?	<i>identify processes.</i>
EFC153R	How much main memory is needed to support the system?	<i>identify resource needs.</i>
EFC154R	What is the average expected hardware configuration?	<i>identify resource needs.</i>
EFC155R	What vendor software is under the company's control?	<i>identify resource needs.</i>
EFC156R	When should text versus graphical design considerations be used?	<i>identify resource needs.</i>
EFC157R	How much secondary storage space is needed to support the system?	<i>identify resource needs.</i>
EFC158R	What are the minimum hardware specifications expected for implementation to support the performance requirements?	<i>identify resource needs.</i>
EFC159R	What hardware could constrain development?	<i>identify resource needs.</i>
EFC160R	What is the lowest common denominator for design consideration of user equipment?	<i>identify resource needs.</i>
EFC161R	What are known resource limitations?	<i>identify resource needs.</i>
EFC162R	What can be done if the performance cannot be met?	<i>identify types of adjustments.</i>
EFC163R	What can be done if the performance is not met?	<i>identify types of adjustments.</i>
EFC164B	What else should I be asking about efficiency "processes"?	<i>uncover additional requirements.</i>



5.4 INTEGRITY (INT)

USER CONCERN: How accurate and authentic are the data?

RELATED CATEGORIES: Accuracy, Completeness, Confidentiality, Data Authenticity, Data Integration, Data Security

5.4 INT

5.4.1 Integrity Definition

Integrity is the degree to which the data maintained by the software system are accurate, authentic, and without corruption.

5.4.2 Integrity Discussion

The term “integrity” is used in this book to mean consistency, accuracy, and correctness of the data. Integrity is an indicator of the user’s trust in the system’s data. This book treats the terms *data* and *information* as synonymous. The integrity of information must be preserved (identically maintained) during operations in which information is transferred, stored, and retrieved.

Integrity requirements describe the ability of the system to survive threats to its data integrity. Threats are potential attacks on the system integrity, both accidental and intentional. A threat to the system integrity is measured by an estimated probability of an attack of a specific type occurring within a particular time frame. Meanwhile, the potential to counteract threats on system integrity can be measured as the probability of counteracting attacks of a particular type.

Integrity is also viewed as the degree to which the system is free from any sorts of intentional attacks (for example, thieves and saboteurs) or non-malicious events (for example, failing hardware, transmission errors, and accidental physical file destruction by the operator). Integrity is also the degree to which data are both untouched by any sort of corruption, removal, change, or addition, and are not penetrated by unauthorized people or processes for any purpose.



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Integrity is a close relative of security. Some authors use the term integrity to encompass a variety of *security* issues, such as blocking unauthorized access to system functions (also called *access security* or *access control*, as explained earlier in subsection 5.1), and ensuring that the software is protected from virus infection, as well as disaster and recovery of software and hardware. As stated earlier, these are not included in the scope of integrity as referenced in this book.

5.4 INT

When eliciting integrity requirements, consider the following aspects:

- ◆ **REGULAR AND CONSISTENT BACKUPS OF THE SYSTEM'S DATA HELP TO PREVENT DATA LOSS.** In the event of system failure (due to any reason), the challenge lies in being able to restore all of the lost information to its original state with 100-percent accuracy.
- ◆ **BACKING UP TO THE SAME DRIVE AS THE NATIVE DATA IS USELESS;** a hard-drive crash can mean losing everything. Backup software should allow the possibility for storing data to a variety of media (external to the original data store). Backup archives are also dependent on the type of medium chosen and its long-term shelf-life.
- ◆ **DATA RESTORE PROCEDURES SHOULD BE TESTED FREQUENTLY TO VERIFY THAT STORAGE HARDWARE IS WORKING PROPERLY.** A faulty hard-drive may reproduce corrupt backup archives no matter how good the backup software is.
- ◆ **DATA AUTHENTICITY:** maintaining the data and representing the data in accord with the original data source.
 - ◊ **Data precision:** specifying the degree of numeric precision in calculations, storage, and presentation.
 - ◊ **Accuracy of information portrayed:** presenting information as intended by the originating source.
 - ◊ **Authentic relationships:** maintaining data relationships as defined by the originating sources.



- ◊ **True to the source of the data**, ensuring the data are consistently stored, presented, and maintained with respect to the originating source.
- ◊ **Data encryption methods**.

5.4.3 Integrity Requirement Examples

- a) All monetary amounts must be accurate to two decimal places.
- b) Accuracy of warehouse temperature readings will be within plus or minus two degrees Celsius.
- c) Whenever a change is made to information stored in Microsoft Word®, the fact of the change shall be recorded in a database or equivalent technology that is routinely backed up. This is intended to identify changed documents in the event of the loss of a disk.
- d) The loan origination system shall perform all calculations with rounding to five (5) decimal places before rounding for presentation to two decimal places (dollars and cents).
- e) The integrity of the system data area must be checked by the internal audit system twice per second; if inconsistencies in the data are detected, the system operation should be disabled.
- f) The precision of calculations with derived data shall be at the same degree of precision as the originating source data.
- g) Derived totals and sub-totals shall be considered transient data and never committed to permanent storage.
- h) Tallies and totals shall acknowledge the source of record when presented without supporting detail.
- i) Presentation of earned premium shall be relative to presentation date, policy renewal date, and payment posted date calculated to a whole day.

5.4 INT



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.4.4 Integrity Suggested Questions

DATA (WHAT?)

5.4 INT

ID	Integrity Suggested Questions	Ask this to:
INT1s	What causes information to be created?	<i>correlate information and events.</i>
INT2s	What causes information to be updated?	<i>correlate information and events.</i>
INT3s	What causes information to be deleted?	<i>correlate information and events.</i>
INT4s	What causes a change in status to information?	<i>correlate information and events.</i>
INT5s	What is the response from the event?	<i>determine information that is output from the event.</i>
INT6s	What is the reaction of the system to each business event?	<i>determine information that is output from the event.</i>
INT7s	What is the response to each business event?	<i>determine information that is output from the event.</i>
INT8s	How does the business event get triggered?	<i>determine what information is received to initiate the event.</i>
INT9s	Who owns the existing information?	<i>identify additional stakeholders.</i>
INT10s	What do internal users need to do their jobs?	<i>identify correlations between information and users.</i>
INT11s	What information is each user looking for?	<i>identify correlations between information and users.</i>
INT12s	What are concerns for the wholeness or completeness of the information?	<i>identify data security needs.</i>
INT13s	If information exists within another system, what is the condition of the data?	<i>identify effort to clean existing data.</i>
INT14s	What information already exists within another system?	<i>identify effort to create data.</i>
INT15s	How does the existing data need to change in the system under development?	<i>identify effort to create data.</i>
INT16s	Who triggers the business event?	<i>identify information input sources.</i>



DATA (WHAT?) (continued)

ID	Integrity Suggested Questions	Ask this to:
INT17s	What information is needed to perform the process?	<i>identify information input sources.</i>
INT18s	Who is the recipient of the business event?	<i>identify information output sources.</i>
INT19s	Who should be notified if a failure occurs?	<i>identify information output sources.</i>
INT20s	What is the output of the process?	<i>identify information output sources.</i>
INT21s	What is the relationship of each information group with other groups?	<i>identify information relationships.</i>
INT22s	What kinds of user activities cause data corruption?	<i>identify problematic uses.</i>
INT23s	What multi-part information entry applications should the user/customer be allowed to recommente entry?	<i>identify process activities.</i>
INT24s	How does each business policy affect information groups?	<i>identify relationships between data and rules.</i>
INT25s	What information is needed to enforce business policies?	<i>identify relationships between data and rules.</i>
INT26s	How long should backed-up data be retained?	<i>identify retention levels.</i>
INT27s	What data must be archived?	<i>identify retention levels.</i>
INT28s	How long will data be stored to allow user/customer to complete entry in a multi-part entry system?	<i>identify retention levels.</i>
INT29s	How long should transaction history be retained?	<i>identify retention periods.</i>
INT30s	How long should archived data be retained?	<i>identify retention periods.</i>
INT31s	What multi-part information should be retained if a failure occurs partway through the entry?	<i>identify retention types.</i>
INT32s	How long should the system retain partial data from interrupted entry?	<i>identify retention types.</i>
INT33s	What records of events happening are needed?	<i>identify retention.</i>
INT34s	What history of changed data is needed?	<i>identify retention.</i>
INT35s	When data are changed, how long should the previous value be retained?	<i>identify retention.</i>

5.4 INT



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

DATA (WHAT?) (continued)

ID	Integrity Suggested Questions	Ask this to:
INT36s	When data are changed, how many generations of previous values need to be retained?	<i>identify retention.</i>
INT37s	What errors must be tracked?	<i>identify retention.</i>
INT38s	In addition to the company security, what must this application do to protect the data?	<i>identify security events and conditions.</i>
INT39s	What are concerns for the authenticity of the information?	<i>identify security events and conditions.</i>
INT40s	What are information concerns for the genuineness or faithfulness of true representation?	<i>identify security events and conditions.</i>
INT41s	What other steps must be taken to protect information?	<i>identify security events and conditions.</i>
INT42s	What availability or assured service of information must be maintained? Assured service is guarding against interruption of service.	<i>identify security risk.</i>
INT43s	What data are not made available for hardcopy reports?	<i>identify types of access.</i>
INT44s	What data should be backed up?	<i>identify types of backup.</i>
INT45s	What are the expectations for data backup? Backing up is the act of making a copy at a specific point in time.	<i>identify types of backups.</i>
INT46s	What are the expectations for data recovery? Recovery is the act of bringing the restored data up to date, usually using an update log of the changes made since the back-up was taken.	<i>identify types of recovery.</i>
INT47s	What is needed to do each user's job?	<i>identify user information needs.</i>
INT48s	What kinds of audit trails or logs are kept?	<i>identify various controls and retention.</i>
INT49R	What data are stored locally (user's computer) while using the application?	<i>identify data security issues.</i>
INT50R	What is the cardinality between related information? (Cardinality is the number—minimum or maximum—of times a relationship occurs.)	<i>identify optional data and volume.</i>
INT51R	What controls over data mining of the application data must be put in place?	<i>identify possible access combinations.</i>
INT52R	What derived data are stored?	<i>identify sources of data.</i>

5.4 INT



DATA (WHAT?) (continued)

ID	Integrity Suggested Questions	Ask this to:
INT53R	What data are received from other systems, stored by this system, and must not be changed by this system?	<i>identify sources of data.</i>
INT54R	Which data elements must always be encrypted to all users?	<i>identify types of controls.</i>
INT55R	Which data elements must always be encrypted to all access attempts?	<i>identify types of controls.</i>
INT56R	Which data elements are completely blocked from unauthorized access?	<i>identify types of data.</i>
INT57R	Which data elements are partially blocked from unauthorized access?	<i>identify types of data.</i>
INT58R	What if data became corrupted?	<i>identify types of monitoring and verification.</i>
INT59R	What safeguards must be put in place for information that resides externally?	<i>identify uncontained information.</i>
INT60B	What else should I be asking about integrity "data"?	<i>uncover additional requirements.</i>

5.4 INT

ROLES (WHO?)

ID	Integrity Suggested Questions	Ask this to:
INT61s	How does each business policy affect each user class?	<i>correlate users and business rules.</i>
INT62s	Who should not trigger this event?	<i>identify access security requirements.</i>
INT63s	What would be the risk to the business if an unauthorized user triggered the event?	<i>identify access security requirements.</i>
INT64s	Who should be denied access through each communication type?	<i>identify access security requirements.</i>
INT65s	Who has authority to trigger the process?	<i>identify access security requirements.</i>
INT66s	Once verified, how is externally sourced data changed?	<i>identify authority.</i>
INT67s	Which users initiate which business events?	<i>identify correlations between users and events.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

ROLES (WHO?) (continued)

ID	Integrity Suggested Questions	Ask this to:
INT68s	Under what conditions can approval decisions be overridden?	<i>identify levels of authority.</i>
INT69s	What should happen when another component/system has not completed its responsibility?	<i>identify levels of responsibility.</i>
INT70s	Who triggers each event?	<i>identify relationships between users and events.</i>
INT71s	Who gets the response from the event?	<i>identify relationships between users and events.</i>
INT72s	How does the process differ by user class?	<i>identify restrictions by user.</i>
INT73s	Who (user/customer role) is authorized to delete data?	<i>identify role authority levels.</i>
INT74s	What component/system is authorized to change data?	<i>identify role authority levels.</i>
INT75s	What component/system is authorized to delete data?	<i>identify role authority levels.</i>
INT76s	When a multiple data store application is used, who (what component/system) is responsible for monitoring the other components/systems?	<i>identify role authority levels.</i>
INT77s	What combinations of data elements must not be shown to any group of people except the highest authorized?	<i>identify role levels of authority.</i>
INT78s	Who (user role) is responsible for the data?	<i>identify role responsibilities.</i>
INT79s	Who should know about changes made to data?	<i>identify role responsibilities.</i>
INT80s	Who has the authority to verify derived data?	<i>identify role responsibilities.</i>
INT81s	Who has the authority to override approval decisions?	<i>identify role responsibilities.</i>
INT82s	What users must see data only related to their roles?	<i>identify role responsibilities.</i>
INT83s	Who owns the interface between components/systems?	<i>identify role responsibilities.</i>
INT84s	What roles are constrained to particular high-level analytics?	<i>identify role responsibilities.</i>
INT85s	What roles have access to reporting system data but will be restricted from access to analytic system data?	<i>identify role responsibilities.</i>
INT86s	What component/system is responsible for the data?	<i>identify role responsibilities.</i>
INT87s	What information must be protected from which user classes at all costs?	<i>identify security issues.</i>

5.4 INT



ROLES (WHO?) (continued)

ID	Integrity Suggested Questions	Ask this to:
INT88s	How are high-level analytics constrained by role?	<i>identify usage.</i>
INT89s	What is the impact on the customer's business if the information is out of date?	<i>identify usage.</i>
INT90s	What consequences are there for two different users having two different versions of the data?	<i>identify usage.</i>
INT91s	How will the information be used?	<i>identify usage.</i>
INT92s	What data elements must be presented in combination and not as individual elements?	<i>identify usage.</i>
INT93s	How many users need access at each location?	<i>identify users.</i>
INT94s	How are data levels of access granted?	<i>identify various authorizers.</i>
INT95s	Which data elements require a higher level of security to access?	<i>identify various levels of access.</i>
INT96R	How many levels of data access exist?	<i>identify various levels of access.</i>
INT97R	What control is required to authorize changes to data?	<i>identify various types of authorization.</i>
INT98R	How many concurrent users are there for each communication type?	<i>identify need to add resources.</i>
INT99R	Who will need access during backup/restore procedures?	<i>identify role responsibilities.</i>
INT100R	Who has authority to override backup/restore procedures?	<i>identify role responsibilities.</i>
INT101R	What user audit processes are in place currently?	<i>identify role responsibilities.</i>
INT102R	What could be done to improve user audit controls?	<i>identify role responsibilities.</i>
INT103B	What else should I be asking about integrity "roles"?	<i>uncover additional requirements.</i>

5.4 INT



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?)

5.4 INT

ID	Integrity Suggested Questions	Ask this to:
INT104s	What are legal concerns in performing this process? Not performing the process?	<i>correlate processes and business rules.</i>
INT105s	How does this event conflict with business policies?	<i>correlate processes and business rules.</i>
INT106s	How is this event triggered by business policies?	<i>correlate processes and business rules.</i>
INT107s	How does each business policy affect business events?	<i>correlate processes and business rules.</i>
INT108s	What restrictions need to be applied to each user class?	<i>correlate users and business rules.</i>
INT109s	What are the effects on business if a user class does not have access to requested information?	<i>identify business risks.</i>
INT110s	How is the business policy dependent on another policy?	<i>identify relationships among business rules.</i>
INT111s	How does the satisfaction of this business policy affect another policy?	<i>identify relationships among business rules.</i>
INT112s	What makes that data necessary to protect?	<i>identify risk.</i>
INT113s	What would be the effect on business if a user class does not have access for a few days, or a few weeks?	<i>identify tolerance of system downtime.</i>
INT114s	What audit trail of access to sensitive or confidential data should be kept?	<i>identify types of monitoring and verification.</i>
INT115s	What audit trail of unsuccessful data access should be maintained?	<i>identify types of monitoring and verification.</i>
INT116s	What audit trail of failed user actions should be kept?	<i>identify types of monitoring and verification.</i>
INT117s	What security violations that the system detects should be recorded?	<i>identify types of monitoring and verification.</i>
INT118s	What record of incorrect authentication (for example, password) entry should be kept?	<i>identify types of monitoring and verification.</i>
INT119s	What combination of data would cause serious damage to the business if made available?	<i>identify various data combinations.</i>



PURPOSE (WHY?) (continued)

ID	Integrity Suggested Questions	Ask this to:
INT120s	What data must always be presented together for an accurate representation of the business?	<i>identify various data relationships.</i>
INT121s	How much data can be lost?	<i>identify various levels of data.</i>
INT122s	What restrictions should be placed on modifying timed changes?	<i>identify various types of controls.</i>
INT123s	What audit trail of user actions should be kept?	<i>identify various types of monitors.</i>
INT124s	What record should be kept of the frequency that data are viewed? Changed?	<i>identify various types of monitors.</i>
INT125s	What audit trail of error conditions detected by the system shall be recorded?	<i>identify various types of monitors.</i>
INT126s	What information must be captured when approval decisions are overridden?	<i>identify various types of monitors.</i>
INT127s	What log of changes should be maintained?	<i>identify various types of monitors.</i>
INT128s	What evidence of processes being run or executed are necessary?	<i>identify various types of monitors.</i>
INT129s	What transaction history is needed? A transaction is a representation of an event or occurrence that happens at a point in time.	<i>identify various types of monitors.</i>
INT130s	What audit trail of significant system events should be kept?	<i>identify various types of monitors.</i>
INT131s	How will data access be audited?	<i>identify various types of monitors.</i>
INT132s	What audits must be in place to detect unintentional or unauthorized changes to data?	<i>identify various types of monitors.</i>
INT133s	What should be known about data changes?	<i>identify various types of monitors.</i>
INT134s	What steps should be taken to audit data quality?	<i>identify various types of monitors.</i>
INT135s	What data functions must have time of activity recorded?	<i>identify various types of monitors.</i>
INT136s	What audits must be in place to detect improper usage of data by either authorized or unauthorized users/customers?	<i>identify various types of monitors.</i>
INT137s	What audits must be in place to detect improper usage of data by either authorized or unauthorized components/systems?	<i>identify various types of monitors.</i>
INT138s	What integrity checks must be in place to validate data after a restore?	<i>identify various types of verification.</i>

5.4 INT



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.4 INT

PURPOSE (WHY?) (continued)

ID	Integrity Suggested Questions	Ask this to:
INT139s	What integrity checks must be in place to validate data after recovery?	<i>identify various types of verification.</i>
INT140s	What integrity checks must be in place to validate data after an abnormal event?	<i>identify various types of verification.</i>
INT141s	What data must be available for high-level analytics of the business?	<i>identify various usages.</i>
INT142s	How should the system prompt the user to verify data when an interrupted entry is resumed?	<i>identify verification types.</i>
INT143s	What are the greatest concerns regarding access to reports produced by the analytic system?	<i>identify vulnerabilities.</i>
INT144s	What information will provide user with confidence that the system is working?	<i>prioritize processes.</i>
INT145s	In what business areas would failure to perform hurt the business the most?	<i>prioritize requirements.</i>
INT146s	Where would users hate to see something go wrong?	<i>prioritize requirements.</i>
INT147s	If the users were away for two weeks, what would be the first things they checked upon returning?	<i>prioritize requirements.</i>
INT148s	What else should I be asking about integrity “purpose”?	<i>uncover additional requirements.</i>

TIMING (WHEN?)

ID	Integrity Suggested Questions	Ask this to:
INT149s	What should happen if a user is in the middle of entering a timed change when its time is reached?	<i>correlate processes and business rules.</i>
INT150s	What should happen to a timed change that needs approval but hasn't been approved by the time it's due?	<i>correlate processes and business rules.</i>
INT151s	When the user is interrupted during data entry, where does the user expect to resume entry?	<i>correlate processes and business rules.</i>



TIMING (WHEN?) (continued)

ID	Integrity Suggested Questions	Ask this to:
INT152s	Where should users recommence entering information later on, from the point they had reached previously?	<i>correlate processes and business rules.</i>
INT153s	When should data be backed up?	<i>identify business cycles.</i>
INT154s	How frequently should data backups be taken?	<i>identify business cycles.</i>
INT155s	How frequently should data be archived?	<i>identify business cycles.</i>
INT156s	Explain the order, if any, of the list of business events.	<i>identify event dependencies.</i>
INT157s	What are the expectations for data restore? (Restore is the act of copying from a back-up when the main data are lost.)	<i>identify levels of restore.</i>
INT158s	What sequence must the process follow?	<i>identify process dependencies.</i>
INT159s	Why would each user interact with the system?	<i>identify standard triggers of events.</i>
INT160s	What stimulates the business event to occur?	<i>identify standard triggers of events.</i>
INT161R	At what logical points or triggers should the user be aware that information entered has been retained?	<i>identify communication types.</i>
INT162R	When will data quality be measured?	<i>identify metrics.</i>
INT163R	When will data security be measured?	<i>identify metrics.</i>
INT164R	How long is the current communication response time?	<i>identify need to add resources.</i>
INT165R	What timed changes to information occur? (A timed change is a change to information at a precise, predetermined moment in time.)	<i>identify process activities.</i>
INT166R	What coordinated changes to information occur? (A coordinated change is a collection of related changes that all need to be applied at exactly the same time.)	<i>identify process activities.</i>
INT167R	What incremental changes of data should be stored between backups?	<i>identify process cycles.</i>
INT168R	When should changes to data be committed?	<i>identify process cycles.</i>
INT169B	What else should I be asking about integrity “timing”?	<i>uncover additional requirements.</i>

5.4 INT



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

LOGISTICS (WHERE?)

5.4 INT

ID	Integrity Suggested Questions	Ask this to:
INT170s	Who needs access to the information at each business location?	<i>identify communication locations.</i>
INT171s	What users access the system in a location that is susceptible to data corruption?	<i>identify integrity needs by location.</i>
INT172s	How does data transmission vary by business location?	<i>identify integrity needs by location.</i>
INT173s	How do integrity needs vary by business location?	<i>identify integrity needs by location.</i>
INT174s	How do users in varying business locations access information differently?	<i>identify integrity needs by location.</i>
INT175s	What business locations must have access to the system?	<i>identify integrity needs by location.</i>
INT176s	What information audits should be performed at each business location?	<i>identify integrity needs by location.</i>
INT177s	What are the peak periods of usage that vary by location?	<i>identify location restrictions.</i>
INT178s	What geographic dependencies are there to the business events?	<i>identify relationships between locations and events.</i>
INT179s	How are business policies geographically dependent?	<i>identify relationships between locations and rules.</i>
INT180s	How does the process vary by geographic location?	<i>identify restrictions by location.</i>
INT181s	What data are stored in multiple places?	<i>identify storage locations.</i>
INT182s	What data will be stored locally?	<i>identify various data types.</i>
INT183R	How should the coordination of data stored in multiple data stores be managed?	<i>identify location controls.</i>
INT184R	How should data stored in multiple data stores be recovered?	<i>identify location controls.</i>
INT185R	How does off-site storage help or hurt integrity?	<i>identify location restrictions.</i>
INT186R	What time periods must be avoided for backup/restore procedures by business location?	<i>identify location restrictions.</i>
INT187R	What data manipulation processes are performed at each location?	<i>identify relationships between processes and location.</i>
INT188R	What resources are needed at each business location to safeguard integrity?	<i>identify resource needs.</i>



LOGISTICS (WHERE?) (continued)

ID	Integrity Suggested Questions	Ask this to:
INT189R	What hardware/software and people resources are needed at each location to perform audit processes?	<i>identify resource needs.</i>
INT190R	How will system usage be monitored by location?	<i>identify usage.</i>
INT191B	What else should I be asking about integrity “logistics”?	<i>uncover additional requirements.</i>

5.4 INT

PROCESS (HOW?)

ID	Integrity Suggested Questions	Ask this to:
INT192S	How does processing differ between deleting and canceling information?	<i>clarify terminology.</i>
INT193S	What functions should each user class be allowed to perform?	<i>correlate users and processing.</i>
INT194S	What types of technology equipment are currently used?	<i>define current resources.</i>
INT195S	What types of communication are currently used?	<i>define current resources.</i>
INT196S	What is the relationship between business events?	<i>determine workflow dependencies.</i>
INT197S	Where does the output from the process go?	<i>identify additional processes or processes that aren't needed.</i>
INT198S	What happens to the output from the process?	<i>identify additional processes or processes that aren't needed.</i>
INT199S	What are the current responsibilities of the users?	<i>identify additional processes.</i>
INT200S	Who can best describe the process?	<i>identify additional stakeholders.</i>
INT201S	What steps are needed to ensure data are represented in the manner intended by the source/authority?	<i>identify integrity checks.</i>
INT202S	What steps are needed to ensure data are maintained in the manner intended by the source/authority?	<i>identify integrity checks.</i>
INT203S	How are derived data verified before presentation?	<i>identify integrity checks.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PROCESS (HOW?) (continued)

5.4 INT

ID	Integrity Suggested Questions	Ask this to:
INT204s	What areas cause bottlenecks or issues?	<i>identify possible process improvement areas.</i>
INT205s	How would the users like the system to work?	<i>identify possible process improvement areas.</i>
INT206s	What information must be updated by a particular user class while other users are viewing the information?	<i>identify process relationships.</i>
INT207s	What entry of information could be disrupted?	<i>identify processes.</i>
INT208s	How is access of sensitive data controlled?	<i>identify types of control.</i>
INT209s	How is externally sourced data verified?	<i>identify types of monitoring and verification.</i>
INT210s	How should coordinated changes be managed? (multi-site, multi-platform, multi-system)	<i>identify various activities or actions.</i>
INT211R	What happens if either the system or an external interface system fails during the transmission of data?	<i>correlate processes and business rules.</i>
INT212R	What controls must be in place when transforming data from reporting systems to analytic data stores?	<i>correlate processes and business rules.</i>
INT213R	What steps should be taken to deal with loss of data from multi-part entry?	<i>correlate processes and business rules.</i>
INT214R	What processes or procedures need to be in place to protect software from viruses that can cause data loss or corruption?	<i>correlate processes and business rules.</i>
INT215R	What processes or procedures need to be in place to protect hardware?	<i>correlate processes and business rules.</i>
INT216R	When a failed component/system returns, what initiates recovery?	<i>correlate processes and business rules.</i>
INT217R	How will data security be measured?	<i>identify metrics.</i>
INT218R	What support processes might be affected by the system?	<i>identify process dependencies.</i>
INT219B	What else should I be asking about integrity “processes”?	<i>uncover additional requirements.</i>



5.5 RELIABILITY (REL)

USER CONCERN:	How immune is the system to failure?
RELATED CATEGORIES:	Availability, Dependability, Fault Tolerance, Maintainability, Survivability

5.5.1 Reliability Definition

Reliability is the extent to which the software system consistently performs the specified functions without failure.

5.5 REL

5.5.2 Reliability Discussion

Reliability might be the most important dynamic characteristic of a software system. The reason for this is that the costs associated with system failure often far exceed development costs.

Informally, the reliability of a software system is the degree to which users think the system provides the services that they require. Reliability is the extent to which the system is doing what it ought to or doing the right thing, as opposed to something else such as producing a wrong answer or producing nothing. “Ought to” or “right thing” must be clearly defined before trying to measure reliability.

Fault avoidance is a term that might be used instead of reliability. Fault avoidance is a strategy applied to the design and development of a system that is *fault-free*. Fault-free software means that the software conforms to its requirement specifications. Because there may be errors in the specifications, or the requirements may not reflect the user’s real needs, fault-free software doesn’t necessarily mean that the system works as the user wants. In statistical terms, reliability is defined as the probability of failure-free operation of a software system in a specified environment for a specified time.



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Reliability requirements describe the degree to which the system operates as expected to deliver the service to the end-users. As such, reliability requirements are closely related to availability and maintainability. For instance, if the system does experience a failure, the system is *unavailable* for its intended work tasks. The system remains unavailable based on the ability to find and fix the error quickly (*maintainability*).

The zeroth law of reliability generalized:
You can reach almost any ambitious level
if you are willing to sacrifice all the other attributes.
—Tom Gilb, author of *Principles of Software Engineering Management* [Gilb, 1988]

5.5 REL

When eliciting reliability requirements, consider the following aspects:

- ◆ **POSSIBLE CAUSES OF SYSTEM FAILURE.**
 - ◊ Poor development or testing practices.
 - ◊ Missed requirements.
 - ◊ Incorrect assumptions regarding system requirements.
 - ◊ Poor user interface.
 - ◊ Faulty hardware.
 - ◊ Inadequate user training (user error).
- ◆ **PREVENTATIVE ACTIONS OR PROCEDURES NECESSARY TO AVOID FAILURE.**
 - ◊ Acceptable data values are defined.
 - ◊ Exception handling for invalid data values is defined.
 - ◊ Functions: algorithms, calculations, and computations are defined and verified.
 - ◊ Combinations, sequences, or series of events are defined.



- ◆ **FAILURE CLASSES** [Sommerville, 2007].
 - ◊ Transient—occurs only with certain inputs.
 - ◊ Permanent—occurs with all inputs.
 - ◊ Recoverable—system can recover without operator intervention.
 - ◊ Unrecoverable—operator intervention needed to recover from failure.
 - ◊ Non-corrupting—failure does not corrupt system state or data.
 - ◊ Corrupting—failure corrupts system state or data.

5.5 REL

- ◆ **RELIABILITY METRICS.**

- ◊ **POFOD** (Probability of failure on demand): a measure of the likelihood that the system will fail when a service request is made. For example, a POFOD of 0.002 means that 2 out of 1000 service requests may result in failure.
- ◊ **ROCOF** (Rate of failure occurrence): a measure of the frequency of occurrence with which unexpected behavior is likely to occur. For example, a ROCOF of 5/100 means that 5 failures are likely to occur in each 100 operational time units.
- ◊ **MTTF** (Mean time to failure): a measure of the time between observed system failures. For example, an MTTF User forgets password.



5.5.3 Reliability Requirement Examples

- a) The Automated Teller Machine (ATM) probability of failure on demand (POFOD) shall be 0.001 (1 out of 1000) when reading the magnetic stripe data on an undamaged card.
- b) The rate of failure occurrence (ROCOF) per ATM shall be 1/1000 (1 occurrence in 1000 days). Failure means the ATM fails to operate with any card inserted, and the software must be restarted to correct the failure.
- c) The mean time to failure (MTTF) of the ATM timing out due to user inactivity shall be 1/1000 (1 occurrence in 1000 transactions). Failure means the ATM must cancel the transaction, and the software must allow the user to start over.
- d) The account update process shall roll back all related updates when any update fails to commit.
- e) The authorization transaction match process shall require a 100-percent match to post a transaction.
- f) The data transmission process shall confirm the receiving terminal is in a ready state prior to the start of transmission.
- g) The point-of-sale terminal shall have a mean time to failure (MTTF) of 1/10,000 (1 occurrence in 10,000 transactions) in a rate of failure occurrence (ROCOF) of 1/30 (1 occurrence in 30 days). Failure is defined as an invalid transaction presented for processing.



5.5.4 Reliability Suggested Questions**DATA (WHAT?)**

ID	Reliability Suggested Questions	Ask this to:
REL1s	What are the classifications for found defects?	<i>classify defects.</i>
REL2s	What is considered a critical bug or defect?	<i>classify defects.</i>
REL3s	What acknowledgment is required indicating that someone has taken responsibility for the problem?	<i>identify communication needs.</i>
REL4s	What notifications are necessary at the time of system failure?	<i>identify communication needs.</i>
REL5s	What means of notification are currently used?	<i>identify communication needs.</i>
REL6s	What training and documentation must be provided for the critical components?	<i>identify communication needs.</i>
REL7s	What types of reports or validations are available or necessary to monitor system accuracy?	<i>identify communication needs.</i>
REL8s	What system failure reports are needed?	<i>identify information.</i>
REL9s	What system failure data are captured currently?	<i>identify information.</i>
REL10s	What system performance data are needed by auditors?	<i>identify information.</i>
REL11s	What error messages and error detection information trails are provided by the system?	<i>identify resources.</i>
REL12s	What alternate, cross-check calculations can be performed to validate derived data?	<i>identify verifications.</i>
REL13s	What components manage unrecoverable data?	<i>identify weaknesses or gaps.</i>
REL14R	What should the system maintain on an event log?	<i>identify information.</i>
REL15R	What data exist for use in rigorous testing to achieve the level of desired reliability?	<i>identify testing techniques.</i>
REL16B	What else should I be asking about reliability “data”?	<i>uncover additional requirements.</i>

5.5 REL

PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

ROLES (WHO?)

ID	Reliability Suggested Questions	Ask this to:
REL17s	What source can provide an accurate count of existing end-users?	<i>identify additional stakeholders.</i>
REL18s	What source can provide an accurate count of existing customers?	<i>identify additional stakeholders.</i>
REL19s	What source can provide an accurate count of existing processes?	<i>identify additional stakeholders.</i>
REL20s	What source can provide an accurate volume count of existing transactions?	<i>identify additional stakeholders.</i>
REL21s	What are the user expectations for system reliability?	<i>identify expectations.</i>
REL22s	What are the customer expectations for system reliability?	<i>identify expectations.</i>
REL23s	What users have a need for greater reliability than others?	<i>identify expectations.</i>
REL24s	Who is responsible for monitoring system errors and outages?	<i>identify responsibilities.</i>
REL25s	Who is responsible for reliability of purchased software and hardware?	<i>identify responsibilities.</i>
REL26s	Who should receive notification when there are system outages?	<i>identify role responsibilities.</i>
REL27s	Who should receive notification when there are system errors?	<i>identify role responsibilities.</i>
REL28s	Which user classes need to be notified when there is a serious error?	<i>identify role responsibilities.</i>
REL29s	Who is authorized to perform what functions on the system?	<i>identify role responsibilities.</i>
REL30s	Who is authorized to access what data?	<i>identify role responsibilities.</i>
REL31s	Who should receive warning messages when errors occur?	<i>identify role responsibilities.</i>
REL32s	What is the typical number of users accessing the system simultaneously?	<i>identify usage.</i>
REL33s	What is the greatest number of simultaneous users during peak processing? Anticipated future usage?	<i>identify usage.</i>
REL34s	What limits apply to the number of simultaneous users?	<i>identify usage.</i>
REL35s	What types of users can be affected by error and failure metering?	<i>identify users.</i>
REL36R	What skill level of technician is required in an outage?	<i>identify responsibilities.</i>
REL37R	How much in-house support for purchased software and hardware is necessary?	<i>identify responsibilities.</i>

5.5 REL



ROLES (WHO?) (continued)

ID	Reliability Suggested Questions	Ask this to:
REL38R	Who performs system upgrades and releases?	<i>identify role responsibilities.</i>
REL39R	Who gets called when system outages occur?	<i>identify role responsibilities.</i>
REL40B	Who receives system error/failure audit reports?	<i>identify role responsibilities.</i>
REL41B	What else should I be asking about reliability "roles"?	<i>uncover additional requirements.</i>

5.5 REL

PURPOSE (WHY?)

ID	Reliability Suggested Questions	Ask this to:
REL42s	Define "system failure."	<i>clarify terminology.</i>
REL43s	Define "repaired."	<i>clarify terminology.</i>
REL44s	What is the acceptable level of known bugs or defect rate allowed to migrate into production?	<i>classify defects.</i>
REL45s	What service agreements are in place with vendors of purchased hardware and software?	<i>identify agreements.</i>
REL46s	What is the cost of lost/incorrect data?	<i>identify business impacts.</i>
REL47s	What is the business benefit of the expected level of reliability?	<i>identify business value.</i>
REL48s	What is the cost of lost business when the expected level of reliability is not achieved?	<i>identify business value.</i>
REL49s	What system components must run for extended periods of time without failure?	<i>identify components.</i>
REL50s	At the time of a problem, what component(s) must receive immediate attention?	<i>identify components.</i>
REL51s	What degree of recovery or restart is expected of failed components?	<i>identify deliverables.</i>
REL52s	Under what circumstances would the whole system be required to shut down?	<i>identify events.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?) (continued)

ID	Reliability Suggested Questions	Ask this to:
REL53s	What is the expected rate of reliability?	<i>identify metrics for the expected level of reliability.</i>
REL54s	What are the success factors for the level of desired reliability?	<i>identify metrics for the expected level of reliability.</i>
REL55s	What is the severity of impact if failures occur?	<i>identify metrics.</i>
REL56s	What is the measurement of defect rate?	<i>identify metrics.</i>
REL57s	What is the acceptable spending limit for this level of reliability?	<i>identify possible trade-offs.</i>
REL58s	What components are most critical to the business?	<i>identify priorities.</i>
REL59s	What components manage the most expensive/critical business elements?	<i>identify priorities.</i>
REL60s	What functions do customers rely on the most?	<i>identify priorities.</i>
REL61s	What functions do users rely on the most?	<i>identify priorities.</i>
REL62s	What components can be sacrificed, temporarily, for the more critical components?	<i>identify priorities.</i>
REL63s	What are the criteria defining classes of errors?	<i>identify tolerance levels.</i>
REL64s	What tolerance for error is acceptable?	<i>identify tolerance levels.</i>
REL65R	What purchased software and hardware are being used?	<i>identify components.</i>
REL66R	What different measures (expectations) exist for the hardware than for the software?	<i>identify failure types.</i>
REL67R	How wide is the exposure to error on the network?	<i>identify integration.</i>
REL68R	What has been the history of releases from the vendors?	<i>identify lessons learned.</i>
REL69R	What is the history/track record for similar installations of vendor components?	<i>identify lessons learned.</i>
REL70R	Why are there different measures needed for hardware than software?	<i>identify metrics.</i>
REL71R	How is the monitoring system audited?	<i>identify possible trade-offs.</i>
REL72R	What software and hardware is considered outdated?	<i>identify resources.</i>

5.5 REL



PURPOSE (WHY?) (continued)

ID	Reliability Suggested Questions	Ask this to:
REL73R	What hardware components should be replaced?	<i>identify resources.</i>
REL74B	What are business consequences when system failure occurs?	<i>identify deliverables.</i>
REL75B	What is the trade-off between reliability and efficiency?	<i>identify possible trade-offs.</i>
REL76B	What is the trade-off between reliability and availability?	<i>identify possible trade-offs.</i>
REL77B	What is the business cost of less than that level of reliability?	<i>identify possible trade-offs.</i>
REL78B	What is the cost of achieving that level of reliability?	<i>identify possible trade-offs.</i>
REL79B	What else should I be asking about reliability "purpose"?	<i>uncover additional requirements.</i>

5.5 REL

TIMING (WHEN?)

ID	Reliability Suggested Questions	Ask this to:
REL80s	What is the required interval for system checking?	<i>identify intervals.</i>
REL81s	What is the minimum expected time duration between outages caused by defects?	<i>identify metrics.</i>
REL82R	What is the maximum acceptable time duration to fix defects?	<i>identify metrics.</i>
REL83s	When is the system <i>most</i> reliable?	<i>identify metrics.</i>
REL84s	When is the system <i>least</i> reliable?	<i>identify metrics.</i>
REL85s	What is the expected mean time between system failures?	<i>identify metrics.</i>
REL86s	What is the expected mean time to repair the system?	<i>identify metrics.</i>
REL87s	What components must be available during an outage?	<i>prioritize components.</i>
REL88s	During what time periods is system reliability most critical?	<i>prioritize usage.</i>
REL89R	When are system upgrades and fixes migrated into production?	<i>identify timing restrictions.</i>
REL90B	What else should I be asking about reliability "timing"?	<i>uncover additional requirements.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

LOGISTICS (WHERE?)

5.5 REL

ID	Reliability Suggested Questions	Ask this to:
REL91s	What are the peak periods of usage that vary by location?	<i>identify location restrictions.</i>
REL92s	What expected users of the application are localized? Dispersed?	<i>identify locations.</i>
REL93s	What locations require reliability the most?	<i>identify locations.</i>
REL94s	What users access the system in a location that is susceptible to reliability issues?	<i>identify reliability needs by location.</i>
REL95s	How does system reliability vary by business location?	<i>identify reliability needs by location.</i>
REL96s	How do users in varying business locations experience system failure more than other locations?	<i>identify reliability needs by location.</i>
REL97s	What business locations must have access to the system?	<i>identify reliability needs by location.</i>
REL98s	What remote access facilities are in place/necessary?	<i>identify types of accesses.</i>
REL99s	What environmental conditions must be created to simulate the planned installation environment for testing to achieve that level of reliability?	<i>identify verifications.</i>
REL100R	How does off-site processing help or hurt reliability?	<i>identify location restrictions.</i>
REL101R	What time periods must be avoided for upgrades and maintenance by business location?	<i>identify location restrictions.</i>
REL102R	How will system errors and outages be monitored by location?	<i>identify monitoring controls.</i>
REL103R	Where are known weak points in the network?	<i>identify possible process improvement areas.</i>
REL104R	What processes are performed at each location?	<i>identify relationships between processes and location.</i>
REL105R	How do releases of production fixes vary by location?	<i>identify reliability needs by location.</i>
REL106R	What resources are needed at each business location to restore a system?	<i>identify resource needs.</i>
REL107R	What resources are needed at each business location to prevent system failure?	<i>identify resource needs.</i>
REL108R	What hardware/software and people resources are needed at each location to perform routine maintenance?	<i>identify resource needs.</i>
REL109R	What is the measure of accuracy with which a given algorithm is being implemented for a specific software and hardware environment?	<i>identify verifications.</i>
REL110B	What else should I be asking about reliability "logistics"?	<i>uncover additional requirements.</i>



PROCESS (How?)

ID	Reliability Suggested Questions	Ask this to:
REL111s	What monitoring tools must be in place?	<i>identify methods of monitoring.</i>
REL112s	What measures of reliability need to be tracked?	<i>identify metrics.</i>
REL113s	What ensures the accuracy of range and value checks?	<i>identify procedural checks.</i>
REL114s	What metrics are used to ensure that all the numeric fields are the right (same) size?	<i>identify procedures.</i>
REL115s	What ensures that all the artificial limits on size and number are eliminated?	<i>identify procedures.</i>
REL116s	What ensures the completeness of test cases?	<i>identify procedures.</i>
REL117s	What escalation procedures are required during an outage?	<i>identify procedures.</i>
REL118s	What alternatives are available during an outage?	<i>identify procedures.</i>
REL119s	What fault-tolerance does the application require?	<i>identify procedures.</i>
REL120s	What off-site redundant implementation procedures are required?	<i>identify procedures.</i>
REL121s	What access control extensions are necessary and permissible to address the problem?	<i>identify procedures.</i>
REL122s	What vendor access is necessary for problem resolution?	<i>identify procedures.</i>
REL123s	How should the system respond under stress?	<i>identify procedures.</i>
REL124s	How will the algorithm be tested to confirm it works as intended?	<i>identify procedures.</i>
REL125s	What configuration management steps are in place for production control?	<i>identify procedures.</i>
REL126s	What procedures need to be in place to address any failures that occur?	<i>identify procedures.</i>
REL127s	What ensures a high degree of confidence in the accuracy of the information?	<i>identify process controls.</i>
REL128s	What should be in place for the system to detect and then prevent unauthorized access?	<i>identify process controls.</i>
REL129s	What procedures are used to ensure that all the data is edited properly?	<i>identify process controls.</i>
REL130s	What process steps can be taken to verify information accuracy?	<i>identify process controls.</i>
REL131s	What will show that mathematical functions are within real number tolerances?	<i>identify process controls.</i>

5.5 REL



PROCESS (HOW?) (continued)

ID	Reliability Suggested Questions	Ask this to:
REL132s	Compared to how it works in the real world, how will users know an algorithm does what it should?	<i>identify process controls.</i>
REL133s	What sources of information are available to validate the values for derived data?	<i>identify process controls.</i>
REL134s	What is the measure of adequacy with which the implemented algorithm relates to real-world interactions?	<i>identify process controls.</i>
REL135s	What test plan should the Quality Assurance testing group follow?	<i>identify process controls.</i>
REL136s	What can be done to reduce exposure to components that run for extended periods of time and cannot fail?	<i>identify process controls.</i>
REL137s	What are the appropriate locations and intervals to check critical data elements for accuracy?	<i>identify process controls.</i>
REL138s	How frequently are system enhancements expected?	<i>identify rates of change.</i>
REL139s	What response time to outages is required?	<i>identify responses.</i>
REL140s	What response time to defects (errors not causing outages) is required?	<i>identify responses.</i>
REL141s	What functions in the production environment should be replicated in a test environment?	<i>identify verifications.</i>
REL142s	What amount of regression testing is minimal when repairing defects?	<i>identify verifications.</i>
REL143s	What amount of regression testing is minimal when making enhancements?	<i>identify verifications.</i>
REL144R	What are known weak points in the network?	<i>evaluate design.</i>
REL145R	How can the known weak points in the network be avoided by this system?	<i>identify alternate processes.</i>
REL146R	How is the system monitored for detection of errors that do not cause an outage?	<i>identify monitoring techniques.</i>
REL147R	How is the system precision and accuracy monitored?	<i>identify monitoring techniques.</i>
REL148R	What systematic error detection and correction monitoring is required?	<i>identify monitoring techniques.</i>
REL149R	How fail safe is the monitoring system?	<i>identify monitoring techniques.</i>
REL150R	How should defects be handled, as "fixed as found" or packaged into releases?	<i>identify processes.</i>

5.5 REL



PROCESS (HOW?) (continued)

ID	Reliability Suggested Questions	Ask this to:
REL151R	What tools do users need to monitor the system?	<i>identify processes.</i>
REL152R	What procedures must be in place to ensure that the data are backed up correctly and are recoverable?	<i>identify processes.</i>
REL153R	What control over vendor software do users have?	<i>identify verifications.</i>
REL154R	What is the plan to manage vendor upgrade releases?	<i>identify verifications.</i>
REL155R	What measures of control are in place on vendor software update releases?	<i>identify verifications.</i>
REL156R	What stress tests can be applied?	<i>identify verifications.</i>
REL157B	What else should I be asking about reliability "processes"?	<i>uncover additional requirements.</i>

5.5 REL



5.6 SURVIVABILITY (SRV)

USER CONCERN:	How resilient is the system from failure?
RELATED CATEGORIES:	Availability, Dependability, Fault Avoidance, Maintainability, Reliability, Robustness

5.6.1 Survivability Definition

Survivability is the extent to which the software system continues to function and recovers in the presence of a system failure.

5.6 SRV

5.6.2 Survivability Discussion

Survivability requirements identify the ability of the system to respond reasonably to unexpected events. The system must have a degree of stability to withstand unexpected business situations and conditions. Survivability of a system is demonstrated by its ability to continue to deliver essential, mission-critical functionality to authorized users while it is under attack or after a portion of the system has been damaged as a result of an attack or system failure.

Robert Ellison, Software Engineering Institute (SEI), and colleagues [Ellison, 2002] define survivability as “the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents.” Ellison, et al. created a method called Survivable Systems Analysis to assess vulnerabilities in systems. This method of achieving survivability depends on three strategies:

- ◆ **RESISTANCE** involves avoiding faults or failures by building capabilities into the system to thwart off attacks. For example, a system might use voice recognition to authenticate users as a way to make it more difficult, albeit not impossible, for unauthorized users to gain entry.



- ◆ **RECOGNITION** consists of detecting faults or failures by building capabilities into the system to do so and assess the resulting damage. For example, the system might add check digits on account numbers so that corruption to the data can be detected.
- ◆ **RECOVERY** means tolerating faults or failures by building capabilities into the system to continue providing essential functionality while under attack and to restore full functionality after an attack. For example, business-critical systems might have software and hardware that execute in parallel in order to pick up processing in the event of a system outage.

Survivability is also referred to as *fault tolerance*, which indicates the degree to which techniques are applied to prevent system faults from resulting in system errors, or system errors from resulting in system failures. Moreover, survivability requirements may also be called *robustness* requirements, as they reflect the ability to continue to deliver essential business-critical services to legitimate users while the system is under attack, or after part of the system has been damaged as a consequence of an attack or a system failure. As such, survivability requirements are closely related to reliability, availability, and maintainability as these nonfunctional requirements are compromised by system failures.

A *fault-tolerant* system can continue in operation after some system faults have occurred. According to Ian Sommerville [Sommerville, 2007], there are four aspects to fault-tolerance:

- ◆ **FAULT DETECTION** involves checking that the system state is consistent, and identifying a failure has occurred or will result in a system failure.
- ◆ **DAMAGE ASSESSMENT** consists of assessing the parts of the system state that have been affected by a fault.
- ◆ **FAULT RECOVERY** includes actions to modify the state of the system so that the effects of the fault are minimized. Forward recovery involves correcting the damaged system, while backward recovery restores the system to its normal “safe” state.

5.6 SRV



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

- ◆ **FAULT REPAIR** involves making modifications to the system so that the fault does not occur again.

When eliciting survivability requirements, consider the following aspects:

- ◆ **FAILURE PREVENTION AND DAMAGE ASSESSMENT TECHNIQUES.**
 - ◊ Exception handling.
 - ◊ Check-sums in data exchange.
 - ◊ Check digits in numeric data.
- ◆ **FAULT RECOVERY TECHNIQUES.**
 - ◊ Forward error recovery (correcting the damaged system).
 - ◊ Backward error recovery (restoring the system).

5.6 SRV



5.6.3 Survivability Requirement Examples

- a) If the audit trail function fails before the user saves updates to the contract, the system shall be able to recover all changes made in the contract being updated up to one minute prior to the failure.
- b) When an update failure is detected, all updates performed during the failed session shall be rolled back to restore the data to pre-session condition.
- c) All data recovered in a roll-back condition shall be recorded for use in forward recovery under user control.
- d) When operating after a failure, the user shall be informed the application is operating in a "safe mode" and all data are available for review without update.
- e) The system shall prevent access to failed functions while providing access to all currently operational functions.
- f) All transactions shall pass three-way hash routine validation before committing transaction update.
- g) All hardware components of the assembly operation shall be replicated, such that failure of any one hardware component shall not render the assembly operation unavailable to end-users. It is acceptable for system performance to be poorer than normal for up to 3 business days following the failure and replacement of a piece of hardware.
- h) All software modules shall pass check-sum verification every time the module is initiated.
- i) The system shall prevent subsequent updates to an account when an update failure has been detected until the failure has been corrected.
- j) The system shall verify all assigned confirmation numbers using dual-verify check digit routines. All confirmation numbers are unique in the system and never re-used.

5.6 SRV



5.6.4 Survivability Suggested Questions

DATA (WHAT?)

ID	Survivability Suggested Questions	Ask this to:
SRV1s	What is the most likely source of corrupt data?	<i>identify problem sources.</i>
SRV2s	What measures can be taken to reduce, eliminate, or cleanse corrupt data?	<i>identify recovery measures.</i>
SRV3s	What is the predictability of corrupt data?	<i>identify validations.</i>
SRV4s	What is the expected level of precision on calculations?	<i>identify validations.</i>
SRV5s	What ranges of valid values are known?	<i>identify validations.</i>
SRV6s	How can the calculations be validated?	<i>identify validations.</i>
SRV7s	What are the applicable limitations for the formulas?	<i>identify validations.</i>
SRV8s	What validation edits are in place for the calculation variables?	<i>identify validations.</i>
SRV9s	What are the numeric data attributes that must be enforced consistently?	<i>identify various attributes.</i>
SRV10s	What date, time, date and time attributes must be enforced consistently?	<i>identify various attributes.</i>
SRV11s	What time standard has been established?	<i>identify various attributes.</i>
SRV12s	What time zone is the standard for the system?	<i>identify various attributes.</i>
SRV13s	What are the data type attributes that must be enforced consistently?	<i>identify various attributes.</i>
SRV14s	What default values must be applied for variables when invalid/missing data are found?	<i>identify various attributes.</i>
SRV15s	How much data can be recovered when problems occur?	<i>qualify data loss.</i>
SRV16s	How much data can be lost when problems occur?	<i>qualify data loss.</i>
SRV17s	How should the system correct/reject nonconforming data?	<i>qualify data loss.</i>
SRV18s	How many attempts at resolution of bad data should be provided?	<i>quantify responses.</i>
SRV19R	What date management and terminology will be used consistently by the system?	<i>identify date specific details.</i>
SRV20R	What check digit routines will be used by the system?	<i>identify fault detection procedures.</i>



DATA (WHAT?) (continued)

ID	Survivability Suggested Questions	Ask this to:
SRV21R	What volatile interfaces can be eliminated?	<i>identify interfaces.</i>
SRV22R	What are the test plans for the calculations?	<i>identify validations.</i>
SRV23B	What else should I be asking about survivability “data”?	<i>uncover additional requirements.</i>

ROLES (WHO?)

ID	Survivability Suggested Questions	Ask this to:
SRV24s	What is the expected growth rate of the number of users, customers, transactions, and products?	<i>assess magnitude of usage patterns.</i>
SRV25s	What authorization is necessary to use default values?	<i>identify access security requirements.</i>
SRV26s	When default values are substituted for invalid data, who needs to be alerted?	<i>identify access security requirements.</i>
SRV27s	Who can validate the calculations?	<i>identify additional stakeholders.</i>
SRV28s	What is the source of calculation formulas?	<i>identify additional stakeholders.</i>
SRV29s	What are the user expectations for system survivability?	<i>identify expectations.</i>
SRV30s	What are the customer expectations for system survivability?	<i>identify expectations.</i>
SRV31s	Who should receive notification when there are system faults?	<i>identify role responsibilities.</i>
SRV32s	Who should receive notification when there are system recoveries or repairs?	<i>identify role responsibilities.</i>
SRV33s	Who needs to be notified when there are serious damages?	<i>identify role responsibilities.</i>
SRV34s	Who is authorized to perform recovery procedures?	<i>identify role responsibilities.</i>
SRV35s	Who should receive warning messages when system faults are encountered?	<i>identify role responsibilities.</i>
SRV36s	What are the conditions under which more than one approval is required?	<i>identify role responsibilities.</i>
SRV37s	Who will be allowed to restore or recover the system?	<i>identify role responsibilities.</i>
SRV38s	What faults, conditions, or events make the authorization conditional?	<i>identify role responsibilities.</i>

5.6 SRV



ROLES (WHO?) (continued)

ID	Survivability Suggested Questions	Ask this to:
SRV39s	What types of extreme user behaviors can be expected?	<i>identify usage.</i>
SRV40s	What is the typical number of users accessing the system simultaneously?	<i>identify usage.</i>
SRV41s	What is the greatest number of simultaneous users during peak processing? Anticipated future usage?	<i>identify usage.</i>
SRV42s	What types of users can be affected by fault and repair monitoring?	<i>identify users.</i>
SRV43R	Who is responsible for monitoring system faults, recoveries, and repairs?	<i>identify responsibilities.</i>
SRV44R	Who is responsible for survivability aspects of purchased software and hardware?	<i>identify responsibilities.</i>
SRV45R	What skill level of technician is required to recover or restore the system?	<i>identify responsibilities.</i>
SRV46R	Who is authorized to perform system backups and restores?	<i>identify role responsibilities.</i>
SRV47R	Who gets called when system failures occur?	<i>identify role responsibilities.</i>
SRV48B	Who receives system fault/repair audit reports?	<i>identify role responsibilities.</i>
SRV49B	What else should I be asking about survivability "roles"?	<i>uncover additional requirements.</i>

5.6 SRV



PURPOSE (WHY?)

ID	Survivability Suggested Questions	Ask this to:
SRV50s	Define “recovery.”	<i>clarify terminology.</i>
SRV51s	Define “system restore.”	<i>clarify terminology.</i>
SRV52s	What are the business rules for the calculations used?	<i>identify applicable standards.</i>
SRV53s	How much can be afforded to make the system as sturdy as requested?	<i>identify possible trade-offs.</i>
SRV54s	What is the cost to the company if there are some failures due to survivability issues?	<i>identify possible trade-offs.</i>
SRV55s	What aspects must be “fool resistant”?	<i>identify vulnerable components.</i>
SRV56s	What is the impact of a failed device?	<i>qualify devices.</i>
SRV57R	What extreme conditions of implementation can be expected?	<i>identify implementations.</i>
SRV58R	What is the expected operating environment of the system?	<i>identify implementations.</i>
SRV59R	What technology is planned for this project?	<i>identify technical environments.</i>
SRV60B	What is the acceptable trade-off between survivable and available?	<i>identify possible trade-offs.</i>
SRV61B	What is the acceptable trade-off between survivable and maintainable?	<i>identify possible trade-offs.</i>
SRV62B	What is the acceptable trade-off between survivable and reliable?	<i>identify possible trade-offs.</i>
SRV63B	What is the acceptable trade-off between survivable and flexible?	<i>identify possible trade-offs.</i>
SRV64B	What else should I be asking about survivability “purpose”?	<i>uncover additional requirements.</i>

5.6 SRV



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

TIMING (WHEN?)

ID	Survivability Suggested Questions	Ask this to:
5.6 SRV	SRV65s When are the periods of high or low usage by the user classes?	<i>identify fault patterns.</i>
	SRV66s When are the periods of high or low occurrences of system faults?	<i>identify fault patterns.</i>
	SRV67s When can system restores be run?	<i>identify time sensitive processes.</i>
	SRV68s When are disaster recovery plans tested?	<i>identify time sensitive processes.</i>
	SRV69s How frequently should fault detection processes run?	<i>identify time-triggered processes.</i>
	SRV70s When can system backups be run?	<i>identify time-triggered processes.</i>
	SRV71R What types of preventative maintenance schedule need to be established for the software and hardware?	<i>identify fault detection procedures.</i>
	SRV72R How often are inspections of hardware and software conducted?	<i>identify routine inspections.</i>
	SRV73R What hardware and software must be locked up when not in use?	<i>identify security events and conditions.</i>
	SRV74R How frequently are repairs migrated into production?	<i>identify time-triggered processes.</i>
	SRV75R How much time is required to recover the system?	<i>identify turnaround expectations.</i>
	SRV76R How much time is required to do a system restore?	<i>identify turnaround expectations.</i>
	SRV77B What else should I be asking about survivability “timing”?	<i>uncover additional requirements.</i>



LOGISTICS (WHERE?)

ID	Survivability Suggested Questions	Ask this to:
SRV78s	How will access security be the same at all business locations?	<i>identify fault risks by location.</i>
SRV79s	What are the different levels of security at the location where disaster recovery hardware, software, and other business artifacts are stored?	<i>identify fault risks by location.</i>
SRV80s	How is access security different for users accessing information remotely?	<i>identify fault risks by location.</i>
SRV81s	What access security is needed for employees that work from home or other remote locations?	<i>identify fault risks by location.</i>
SRV82s	What access restrictions are necessary to the work areas where this application will be used?	<i>identify fault risks by location.</i>
SRV83s	What security measures must be taken when accessing the application using public computers?	<i>identify fault risks by location.</i>
SRV84s	Where is regulatory information maintained?	<i>identify regulations.</i>
SRV85R	What resources are needed to recover/restore each location?	<i>identify resources needed.</i>
SRV86R	Where is backup software and hardware kept?	<i>identify resources needed.</i>
SRV87R	Where is backup information stored?	<i>identify resources needed.</i>
SRV88B	What else should I be asking about survivability “logistics”?	<i>uncover additional requirements.</i>

5.6 SRV



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PROCESS (HOW?)

ID	Survivability Suggested Questions	Ask this to:
SRV89s	In similar applications, where have there been points of failure that should have been more robust?	<i>identify alternative processes.</i>
SRV90s	What alternatives exist for processing without the corrupt data?	<i>identify alternative processes.</i>
SRV91s	What does the current technology prohibit or hinder?	<i>identify capabilities.</i>
SRV92s	What recovery capability from errors is expected?	<i>identify capabilities.</i>
SRV93s	When extreme input is encountered, how fast must the system respond and recover?	<i>identify fault detection procedures.</i>
SRV94s	How should the system respond to input errors?	<i>identify fault detection procedures.</i>
SRV95s	How should the system react to unusual error conditions?	<i>identify fault detection procedures.</i>
SRV96s	Which interfaces are critical to business continuity when faults occur?	<i>identify interfaces.</i>
SRV97s	What is the practical minimum number of transactions and events the system must handle?	<i>identify measures.</i>
SRV98s	What is the practical minimum number of users and customers the system must handle?	<i>identify measures.</i>
SRV99s	What procedures are necessary to "show your work" on calculations?	<i>identify procedures.</i>
SRV100s	Which processes must continue to function or survive with reduced capability when faults occur?	<i>identify processes.</i>
SRV101s	Which processes should terminate gracefully when faults occur?	<i>identify processes.</i>
SRV102s	What should the process do to terminate gracefully?	<i>identify processes.</i>
SRV103s	If there are failures due to unexpected corrupt data or hardware failures, what lost business can be recovered?	<i>identify recovery measures.</i>
SRV104s	What workaround is available for a failed hardware device?	<i>identify recovery measures.</i>
SRV105s	Which processes performed by the system have manual workarounds?	<i>identify risk areas.</i>
SRV106s	How should the system respond to extreme conditions?	<i>identify types of responses.</i>
SRV107s	What types of error conditions might the system encounter?	<i>identify various conditions.</i>

5.6 SRV



PROCESS (HOW?) (continued)

ID	Survivability Suggested Questions	Ask this to:
SRV108R	What can be done to make the hardware devices more robust?	<i>identify alternative processes.</i>
SRV109R	What software and hardware has a history of failures?	<i>identify alternative processes.</i>
SRV110R	What alternatives exist for software and hardware that have a history of failures?	<i>identify alternative processes.</i>
SRV111R	What alternatives exist for any devices currently used?	<i>identify alternative processes.</i>
SRV112R	What design considerations should be made to reduce the impact of anticipated trouble spots?	<i>identify configurations.</i>
SRV113R	What design considerations should be made to improve the robustness of anticipated trouble spots?	<i>identify configurations.</i>
SRV114R	What hardware fault tolerance is acceptable?	<i>identify configurations.</i>
SRV115R	What hardware redundancy is necessary?	<i>identify configurations.</i>
SRV116R	Which hardware devices must be the most robust?	<i>identify configurations.</i>
SRV117R	How can the system predict a hardware device failure?	<i>identify configurations.</i>
SRV118R	What is the response plan for a failed device?	<i>identify fault detection procedures.</i>
SRV119R	Which interfaces have the highest degree of volatility?	<i>identify interfaces.</i>
SRV120R	What alternates are there for the volatile interfaces?	<i>identify interfaces.</i>
SRV121R	How can the volatile interfaces be reduced in priority?	<i>identify interfaces.</i>
SRV122R	What is the cause of volatility in any interfaces?	<i>identify interfaces.</i>
SRV123R	What measures are in place to safeguard software and hardware from natural disasters?	<i>identify security needs.</i>
SRV124R	What measures are in place to safeguard software and hardware from unauthorized access?	<i>identify security needs.</i>
SRV125R	What measures are in place to recover software and hardware from natural disasters?	<i>identify security needs.</i>
SRV126R	What measures are in place to recover software and hardware from unauthorized access?	<i>identify security needs.</i>
SRV127B	What else should I be asking about survivability "processes"?	<i>uncover additional requirements.</i>

5.6 SRV



5.7 USABILITY (USE)

USER CONCERN:	How easy is it to learn and operate the system?
RELATED CATEGORIES:	Human Engineering, Operability, User-friendliness

5.7.1 Usability Definition

Usability is the ease with which the user is able to learn, operate, prepare inputs, and interpret outputs through interaction with a software system.

5.7 USE

5.7.2 Usability Discussion

Usability requirements indicate how well people are able and willing to use the system. Usability requirements are concerned with the user interface and end-user interaction with the system. *User-friendliness* and *human engineering* are terms often associated with usability, and describe the user's desire for a system that is easy to interact with and use. The usability of a system has an effect on productivity, error rates, and acceptance of the system. Usability requirements are often compromised by trade-offs between what the project sponsor is trying to achieve (business objective) and what the users expect (functionality to get their work done).

If the system does not work as intended by the programmer, we have a program error—a bug. If it is impossible to carry out the task, we have a requirement defect—missing functionality.

If the system works as intended by the programmer, and it can support the task, yet the user cannot figure out how to do it or doesn't like the system, we have a usability problem.

—Soren Lauesen, author of *Software Requirements: Styles and Techniques* [Lauesen, 2002]



When eliciting usability requirements, consider the following aspects:

- ◆ **EASE OF ENTRY:** the extent that the system accounts for the human requirements for success in inputting information or materials into the system. It is often expressed in physical requirements (such as vision, hearing, and manual dexterity), and in terms of educational, intelligence, vocational, or cultural requirements.
- ◆ **EASE OF LEARNING:** the degree to which the system provides resources to help the user attain some measurable level of capability. The learning ability would be expected to vary with the capability of the individual user (such as previous experience with similar systems) and other human factors.
- ◆ **EASE OF HANDLING:** the extent to which the system aids the user in productivity. This is usually measured over time by a reduction in end-user errors. The ease of handling also will vary for specific classes of users.
- ◆ **LIKABILITY:** the measure of how well people like to use the system. One common way of measuring likability is user opinion surveys.
- ◆ **POSSIBLE USABILITY METRICS** [Lauesen, 2002]:
 - ◊ **Problem counts:** metric for ease of learning.
 - ◊ **Task time:** metric for ease of learning and task efficiency.
 - ◊ **Keystroke counts:** metric for task efficiency for experienced users.
 - ◊ **Opinion poll:** subjective metric for user satisfaction.
 - ◊ **Score for understanding:** subjective metric for understandability.
 - ◊ **Design-level requirements:** metric descriptions regarding the user interface.
 - ◊ **Product-level requirements:** metric for ease of remembering entry values.
 - ◊ **Guideline adherence:** metric for the general appearance and response on the user interface.
 - ◊ **Development process requirements:** metric for specifics applied to the development process.

5.7 USE



5.7.3 Usability Requirement Examples

- a) The new product shall be easy to use by adult members (age 18 to 80) of the public who may only have one hand free.
- b) The vending product shall be able to be used by adult members of the public without training. A panel representative of at least 95 percent of the general public shall successfully purchase a product from the vending product on their first encounter.
- c) The product shall be self-explanatory and intuitive such that a service agent shall be able to produce a price quote within 10 minutes of encountering the product for the first time.
- d) The new policy management system shall be evaluated by 90 percent of the user community to be at least as easy to use as the existing system.
- e) A trained order-entry clerk shall have the ability to submit a complete order for a product chosen from a supplier catalog in a maximum of 7 minutes, with an average order entry time of 4 minutes.
- f) People with no training and no understanding of English shall be able to use the product.
- g) A new warehouse clerk shall be able to enter a customer order on the system within a typical 8-hour business day.
- h) The system shall be useable by program developers after five weeks of training.



5.7.4 Usability Suggested Questions

DATA (WHAT?)

ID	Usability Suggested Questions	Ask this to:
USE1s	What are the institutional shorthand notations for common data elements?	<i>identify various types of data.</i>
USE2s	What natural clustering of information exists in the business area?	<i>identify business areas.</i>
USE3s	Where can the specific wording for the error messages, descriptions, and definitions be obtained?	<i>identify communications sources.</i>
USE4s	What confirmation of progress is expected?	<i>identify levels of communication.</i>
USE5s	What levels of user help are necessary?	<i>identify levels of help.</i>
USE6s	What levels of user interactive help are expected?	<i>identify levels of help.</i>
USE7s	What degree of interactive help is expected?	<i>identify levels of help.</i>
USE8s	What level(s) of error message information is expected?	<i>identify message levels.</i>
USE9s	How interactive is the process for data entry between data input and data contained in the system?	<i>identify processes.</i>
USE10s	What options should be available to turn on or turn off help features?	<i>identify types of help.</i>
USE11s	When is it acceptable to use the common shorthand notations?	<i>identify usage habits.</i>
USE12s	When is it not acceptable to use the common shorthand notations?	<i>identify usage habits.</i>
USE13s	How frequently do current help features get used?	<i>identify usage patterns.</i>
USE14s	What is the maximum number of keystrokes and presentation controls acceptable during data input?	<i>identify user interfaces.</i>
USE15s	What informative warning messages and error messages are needed?	<i>identify various communications.</i>
USE16s	What input/output devices will be supported?	<i>identify various devices.</i>
USE17s	What sort of input/output devices for the human interface are available and what are their characteristics?	<i>identify various devices.</i>

5.7 USE



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

DATA (WHAT?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE18s	What specific terms, symbols, or expressions are commonly used by the users that need to be incorporated?	<i>identify various dialects.</i>
USE19s	How many types of help are expected?	<i>identify various forms of help.</i>
USE20s	Where do the users typically look first for help?	<i>identify various forms of help.</i>
USE21s	What "help" facilities or features are needed?	<i>identify various forms of help.</i>
USE22s	Where do the users expect to find help?	<i>identify various forms of help.</i>
USE23s	How appealing is the presentation of the information?	<i>identify various presentation styles.</i>
USE24s	Specify the existence and required features of online help systems, wizards, tool tips, context-sensitive help, user manuals, and other forms of documentation and assistance.	<i>identify various types of communication.</i>
USE25s	What expectations are there for error messages, descriptions, and definitions in varying lengths depending upon user?	<i>identify various types of communications.</i>
USE26s	What are the different lengths of error messages, descriptions, and definitions?	<i>identify various types of communications.</i>
USE27s	What data elements are repetitive and may be shortcut during data input?	<i>identify various types of data.</i>
USE28s	How complex is the combination of data during data entry?	<i>identify various types of data.</i>
USE29s	What "help" documentation is available to the user?	<i>identify various types of documentation.</i>
USE30s	What information or materials do the end users need in order to do their jobs?	<i>identify various types of information or material.</i>
USE31s	What history records of user injuries exist?	<i>identify various types of injuries.</i>
USE32s	How much data must be presented in common presentations?	<i>identify various types of presentations.</i>
USE33s	What clues for use are the users accustomed to having in their applications?	<i>identify various types of user guides.</i>
USE34R	What historical data are never retrieved by users?	<i>identify data cleansing areas.</i>
USE35R	What duplication of input could be eliminated or reduced?	<i>identify data cleansing areas.</i>
USE36R	What data labels are inconsistent or confusing?	<i>identify data cleansing areas.</i>

5.7 USE



DATA (WHAT?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE37R	Regarding what data and functions does the help desk get repeated calls from users?	<i>identify types of help.</i>
USE38R	What additional data could be pre-populated for the user and reduce data entry time and amount?	<i>identify types of help.</i>
USE39R	What data are most frequently retrieved by each user class?	<i>identify various types of data.</i>
USE40R	What data are viewed the most by each user class?	<i>identify various types of data.</i>
USE41B	What else should I be asking about usability "data"?	<i>uncover additional requirements.</i>

ROLES (WHO?)

ID	Usability Suggested Questions	Ask this to:
USE42s	What type of users are the end users?	<i>classify users.</i>
USE43s	What prior experience are users expected to have?	<i>classify users.</i>
USE44s	What is the range in years of experience of the end users?	<i>classify users.</i>
USE45s	What is the range in education level of the end users?	<i>classify users.</i>
USE46s	What different access levels will be offered to power users and casual users?	<i>classify users.</i>
USE47s	Who is the intended audience of the system?	<i>classify users.</i>
USE48s	What are the data throughput expectations per user role in the department?	<i>classify users.</i>
USE49s	What aspects of the system must be foolproof?	<i>identify access security risks.</i>
USE50s	What are the expectations regarding training on the new system?	<i>identify levels of training.</i>
USE51s	How much training is expected?	<i>identify levels of training.</i>
USE52s	What level of training will be provided, offered, or made available?	<i>identify levels of training.</i>
USE53s	What are the expectations for allowing users to alter their presentation formats?	<i>identify levels of usage.</i>
USE54s	What user access control preferences must be retained?	<i>identify preference types.</i>

5.7 USE



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

ROLES (WHO?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE55s	What user access control profiles must be retained?	<i>identify profile types.</i>
USE56s	What experience does the staff have in developing solutions for users with specific needs?	<i>identify resource needs.</i>
USE57s	What is the skill level of the target user group?	<i>identify skill levels.</i>
USE58s	What is the skill level of the majority of the primary users?	<i>identify skill levels.</i>
USE59s	What percentage of users will have specific needs?	<i>identify specific needs.</i>
USE60s	How does using the system change between skill levels of the users?	<i>identify system usages.</i>
USE61s	What training will be necessary before using the system?	<i>identify types of training.</i>
USE62s	What is the accessibility for handicapped users?	<i>identify types of usage.</i>
USE63s	What are the expectations for allowing users to filter out data?	<i>identify types of usage.</i>
USE64s	Describe the typical usage of novice users. Expert users.	<i>identify user capabilities.</i>
USE65s	How rapidly are users expected to advance from one category to the next?	<i>identify user capabilities.</i>
USE66s	How important is it for the system to be understandable by the users?	<i>identify user capabilities.</i>
USE67s	How many categories of user types are there? (novice, experienced, power)	<i>identify user categories.</i>
USE68s	Who is the expected user audience?	<i>identify user groups.</i>
USE69s	What do the users consider to be "stressful" aspects of their work?	<i>identify user perceptions.</i>
USE70s	What do users commonly express as "frustrations" with using the system?	<i>identify user perceptions.</i>
USE71s	Describe the typical user of the system.	<i>identify user types.</i>
USE72s	Describe the "novice" or "beginner" user.	<i>identify user types.</i>
USE73s	Describe the "expert" or "power" user.	<i>identify user types.</i>
USE74s	How much user to user collaboration happens while using the system?	<i>identify user usages.</i>
USE75s	What are the expectations for users to learn the system without providing training?	<i>identify user usages.</i>

5.7 USE



ROLES (WHO?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE76s	How do different levels of users view the relationships within the data?	<i>identify user views.</i>
USE77s	How well will the users remember how to use the low-frequency features?	<i>identify various features.</i>
USE78s	How do the users "feel" about the system?	<i>identify various perceptions.</i>
USE79s	What specific needs of the prospective users must be met?	<i>identify various user needs.</i>
USE80R	What user classes require more help than others?	<i>identify levels of training.</i>
USE81R	What particular sequence should be used to train the users?	<i>identify levels of training.</i>
USE82R	What user classes have the most volume of data entry errors?	<i>identify levels of training.</i>
USE83R	What volume of users are there currently? Expected in the future?	<i>identify resource needs.</i>
USE84R	What users input the most data?	<i>identify user usages.</i>
USE85R	What usability characteristics are requested by novice users? By expert users?	<i>identify user usages.</i>
USE86R	What help features are most frequently used? What features are not used?	<i>identify various features.</i>
USE87R	What user classes use existing help features?	<i>identify various features.</i>
USE88B	Who is responsible for training the users?	<i>identify role responsibilities.</i>
USE89B	Who is responsible for supporting the users?	<i>identify role responsibilities.</i>
USE90B	What else should I be asking about usability "roles"?	<i>uncover additional requirements.</i>

5.7 USE



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?)

ID	Usability Suggested Questions	Ask this to:
USE91s	What similar systems are available that the users are familiar with and like to use?	<i>identify comparatives.</i>
USE92s	What similar systems are available that the users are familiar with and do not like to use?	<i>identify comparatives.</i>
USE93s	What other interfaces are similar to that used in this system with which the users are familiar?	<i>identify comparatives.</i>
USE94s	What similar systems do the users complain about?	<i>identify comparatives.</i>
USE95s	Describe the usability of the proposed system with other state-of-the-art systems that the user community knows and likes.	<i>identify comparatives.</i>
USE96s	What other system could be used as a model for "ease of use"?	<i>identify comparatives.</i>
USE97s	What other system could be used as a model for "easy to learn"?	<i>identify comparatives.</i>
USE98s	What recognizable system can be used as a model or pattern?	<i>identify comparatives.</i>
USE99s	What are examples of good designs to emulate?	<i>identify comparatives.</i>
USE100s	What must be included beyond company usability design standards?	<i>identify exceptional standards.</i>
USE101s	What current company usability standards are not adequate?	<i>identify existing standards.</i>
USE102s	What usability laws or standards must be adhered to?	<i>identify laws or standards.</i>
USE103s	What usability improvements have been implemented in the past?	<i>identify lessons learned.</i>
USE104s	What is the importance that the system is easy to learn?	<i>identify metrics.</i>
USE105s	What is an allowable amount of time needed before users can successfully use the product?	<i>identify metrics.</i>
USE106s	Specify the required training time for users to become minimally productive (able to accomplish simple tasks) and operationally productive (able to accomplish normal, day-to-day tasks).	<i>identify metrics.</i>
USE107s	What are learning rate requirements?	<i>identify metrics.</i>
USE108s	How will learning rate be measured?	<i>identify metrics.</i>

5.7 USE



PURPOSE (WHY?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE109s	Describe characteristics to measure "easy to learn."	<i>identify metrics.</i>
USE110s	How are users measured on "ability to learn quickly"?	<i>identify metrics.</i>
USE111s	Once users have been trained to perform routine tasks, how long should it take them to perform a typical task or transaction?	<i>identify metrics.</i>
USE112s	How will the usability of the system be evaluated?	<i>identify metrics.</i>
USE113s	What metrics for user productivity are used?	<i>identify metrics.</i>
USE114s	What are entry rate requirements?	<i>identify metrics.</i>
USE115s	How will entry rate be measured?	<i>identify metrics.</i>
USE116s	What productivity gains are expected from using the system?	<i>identify metrics.</i>
USE117s	What productivity rate increase, in what timeframe, is expected from the system?	<i>identify metrics.</i>
USE118s	Specify measurable task times for typical tasks or transactions.	<i>identify metrics.</i>
USE119s	Please fill in the following: A user shall produce a [<i>specified result</i>] within [<i>specified time</i>] of beginning to use the product, without having to reference the user's manual.	<i>identify metrics.</i>
USE120s	Please fill in the following: After receiving [<i>number of hours</i>] training a user shall be able to produce [<i>quantity of specified outputs</i>] per [<i>unit of time</i>].	<i>identify metrics.</i>
USE121s	Please fill in the following: [<i>Agreed percentage</i>] of a test panel shall successfully complete [<i>specified task</i>] within [<i>specified time limit</i>].	<i>identify metrics.</i>
USE122s	Please fill in the following: After [<i>specified time duration, e.g., one month</i>], usage of the product shall result in a total error rate of less than [<i>an agreed-upon percentage</i>].	<i>identify metrics.</i>
USE123s	Please fill in the following: An anonymous survey shall show that [<i>an agreed percentage</i>] of the users are regularly using the product after [<i>an agreed time</i>] familiarization period.	<i>identify metrics.</i>

5.7 USE



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE124s	Please fill in the following: The users shall achieve [<i>agreed percentage</i>] pass rate from the final examination of the training.	<i>identify metrics.</i>
USE125s	What are handling or error rate requirements?	<i>identify metrics.</i>
USE126s	What error rate decrease, in what timeframe, is expected from the users of the system?	<i>identify metrics.</i>
USE127s	What is an acceptable user error rate when working at normal speed?	<i>identify metrics.</i>
USE128s	What reduction rate of user errors is expected?	<i>identify metrics.</i>
USE129s	How should acceptance of usability be measured?	<i>identify metrics.</i>
USE130s	How will user satisfaction be measured?	<i>identify metrics.</i>
USE131s	How is "likability" measured?	<i>identify metrics.</i>
USE132s	How are user errors measured?	<i>identify metrics.</i>
USE133s	What is the rate of errors by the users?	<i>identify metrics.</i>
USE134s	How will error rate be measured?	<i>identify metrics.</i>
USE135s	What standards/guidelines for navigation apply?	<i>identify navigation standards.</i>
USE136s	What complaints are heard most often from the users about systems they use on a regular basis?	<i>identify pain points.</i>
USE137s	What is the trade-off between "easy to use" and "easy to learn"?	<i>identify perspectives.</i>
USE138s	What standards/guidelines for amount of data on a presentation or the number of presentations apply?	<i>identify presentation standards.</i>
USE139s	What government or other regulatory compliance standards must be applied regarding user safety?	<i>identify safety standards.</i>
USE140s	What style guidelines should be followed?	<i>identify styles.</i>
USE141s	What will it take to make a successful system for the user?	<i>identify success criteria.</i>
USE142s	What conventions and standards developed for the human-to-machine interface must be applied?	<i>identify types of interfaces.</i>

5.7 USE



PURPOSE (WHY?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE143s	What accident prevention requirements must be considered?	<i>identify types of safety.</i>
USE144s	What global safety requirements or constraints must the system satisfy?	<i>identify types of safety.</i>
USE145s	What safety issues and concerns must users be educated about?	<i>identify types of safety.</i>
USE146s	How important is it to develop the system so that it minimizes user errors? What is the importance that users be protected from making errors?	<i>identify various conditions / events.</i>
USE147s	What are examples of bad designs to avoid? .	<i>identify various designs.</i>
USE148s	What will aid in user orientation and recognition when first exposed to the new system?	<i>identify various presentation styles.</i>
USE149s	What safety risks need to be addressed?	<i>identify various risks.</i>
USE150s	In terms of usability, what would make this system better than a competing one?	<i>identify various usability comparatives.</i>
USE151s	What is the user trying to achieve by using the system?	<i>identify various user goals.</i>
USE152R	What advanced technology could help users be more productive?	<i>identify improvement areas.</i>
USE153R	What usability design changes are recommended?	<i>identify improvement areas.</i>
USE154R	What could be done to improve usability?	<i>identify improvement areas.</i>
USE155R	In what way does training hinder usability?	<i>identify improvement areas.</i>
USE156R	In what way do user help components hinder usability?	<i>identify improvement areas.</i>
USE157R	In what way do current system features hinder usability?	<i>identify improvement areas.</i>
USE158R	What business policies put constraints on usability design?	<i>identify pain points.</i>
USE159R	What design standards constrain usability design?	<i>identify pain points.</i>
USE160R	What usability issues are potential business risks?	<i>identify pain points.</i>
USE161B	What design standards must be applied to ensure consistency?	<i>identify design standards.</i>
USE162B	What else should I be asking about usability “purpose”?	<i>uncover additional requirements.</i>

5.7 USE



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

5.7 USE

TIMING (WHEN?)

ID	Usability Suggested Questions	Ask this to:
USE163s	What is the rate of acceptance by the users?	<i>assess acceptance.</i>
USE164s	What is a typical learning curve?	<i>assess learning.</i>
USE165s	What duration of time is an acceptable learning period?	<i>assess learning.</i>
USE166s	How long should it take to complete a process?	<i>identify process guidelines.</i>
USE167s	While entering data, how much time do the users spend reviewing, analyzing, or verifying the data?	<i>identify process interactions.</i>
USE168s	How important is it to complete a process quickly and easily?	<i>identify process performance levels.</i>
USE169s	When processing work, where are the breaks in sequence?	<i>identify process sequences.</i>
USE170s	What sequencing of activities is common to the business area?	<i>identify related activities.</i>
USE171s	What amount of time is required for a reply?	<i>identify response to events.</i>
USE172s	How long before re-prompt and time out?	<i>identify response to events.</i>
USE173s	What on-going user training must be provided?	<i>identify training needs.</i>
USE174s	What incremental user training should be provided?	<i>identify training needs.</i>
USE175s	How long should it take first-time users to learn the system to a proficient level?	<i>identify types of users.</i>
USE176s	How long should it take experienced users to learn the system?	<i>identify types of users.</i>
USE177s	What timing constraints are there on completion of a transaction?	<i>identify usage events.</i>
USE178s	How frequently will most functions be used?	<i>identify usage frequencies.</i>
USE179s	Which functions will be used the least frequently?	<i>identify usage frequencies.</i>
USE180s	What tasks are performed infrequently?	<i>identify usage patterns.</i>
USE181s	When there are breaks in the sequence of the work, how long are the breaks?	<i>identify usage patterns.</i>
USE182s	When there are breaks in the sequence of the work, how is the system expected to resume processing?	<i>identify usage patterns.</i>



TIMING (WHEN?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE183s	How quickly must users learn to use the most frequented features?	<i>identify various features.</i>
USE184R	When should users be trained?	<i>assess learning.</i>
USE185R	How long would it take the average user to get up to speed?	<i>assess learning.</i>
USE186R	When do most data entry errors occur?	<i>identify usage patterns.</i>
USE187R	When does the help desk receive the most calls?	<i>identify usage patterns.</i>
USE188B	During which peak periods is usability critical?	<i>identify usage patterns.</i>
USE189B	How often is user help updated?	<i>assess acceptance.</i>
USE190B	What else should I be asking about usability “timing”?	<i>uncover additional requirements.</i>

5.7 USE**LOGISTICS (WHERE?)**

ID	Usability Suggested Questions	Ask this to:
USE191s	What are user needs to switch between multiple currencies?	<i>identify conversion combinations.</i>
USE192s	What multiple currencies must be supported?	<i>identify currencies.</i>
USE193s	What expectations are there for the system to be used by people who do not speak the language of the country where the system is used?	<i>identify distribution.</i>
USE194s	Describe the work environment adjacent to the specific area being considered.	<i>identify environment types.</i>
USE195s	Describe the typical environment in which the users do their work.	<i>identify environments.</i>
USE196s	Describe the ergonomic usage of the system.	<i>identify ergonomic levels.</i>
USE197s	What multiple languages must be supported?	<i>identify languages.</i>
USE198s	What is the sound quality of the system?	<i>identify sound levels.</i>



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

LOGISTICS (WHERE?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE199s	What special environmental conditions must be considered?	<i>identify types of environments.</i>
USE200s	What special lighting needs are there?	<i>identify types of lighting.</i>
USE201s	What multiple currencies must be available in the same instance of the system?	<i>identify various combinations.</i>
USE202s	What external devices must be used in concert to provide accessibility?	<i>identify various combinations.</i>
USE203s	What geographical or logistical constraints impact the user's ability to use the system?	<i>identify various constraints.</i>
USE204s	What textual and graphical visual interfaces are needed?	<i>identify visual interface types.</i>
USE205R	What level of user support is needed at each business location?	<i>identify resource needs.</i>
USE206R	What usability issues vary by business location?	<i>identify types of environments.</i>
USE207R	Where does each user class do its work?	<i>identify types of environments.</i>
USE208R	What different levels of help information are needed at different locations?	<i>identify usability needs.</i>
USE209R	What could be done to improve usability at each location?	<i>identify usability needs.</i>
USE210B	What locations can accommodate user training?	<i>identify resource needs.</i>
USE211B	What else should I be asking about usability "logistics"?	<i>uncover additional requirements.</i>

5.7 USE



PROCESS (How?)

ID	Usability Suggested Questions	Ask this to:
USE212s	What features of the current technology constrain usability?	<i>assess technology.</i>
USE213s	What characteristics are needed to make the system “easy” to use?	<i>classify characteristics.</i>
USE214s	What features/functions are critical to enabling users to perform their jobs?	<i>classify features.</i>
USE215s	Which features are used for high volume data input?	<i>classify features.</i>
USE216s	What makes the product safe to use? Unsafe?	<i>classify safety.</i>
USE217s	How necessary is it for the system to lead users through the task?	<i>identify activities.</i>
USE218s	What must be put in place to avoid hazards?	<i>identify hazards.</i>
USE219s	What must be put in place to minimize accidents if hazards arise?	<i>identify hazards.</i>
USE220s	What is necessary to ensure safe operation?	<i>identify levels of safety.</i>
USE221s	What makes the information easy to read?	<i>identify presentation styles.</i>
USE222s	How many variations in presentation exist within user roles?	<i>identify presentation types.</i>
USE223s	What procedures must be taken in the event of an accident?	<i>identify procedures.</i>
USE224s	What are the routines (process flow) followed in the business area?	<i>identify processes.</i>
USE225s	How does the business area react to disruption of their normal routine?	<i>identify scenarios.</i>
USE226s	How does the system conform to any established user interface standards?	<i>identify standards.</i>
USE227s	What user tasks show the highest incidences of accidents?	<i>identify types of accidents.</i>
USE228s	What workers compensation claims have been made in the past?	<i>identify types of accidents.</i>
USE229s	What hazard identification and analysis is needed?	<i>identify types of hazards.</i>
USE230s	What should the system do to “remember” where users have been and assist in their return?	<i>identify usage patterns.</i>
USE231s	What different components have specific access procedures?	<i>identify various access processes.</i>
USE232s	Describe unacceptable or undesirable system behavior.	<i>identify various behaviors.</i>

5.7 USE



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PROCESS (How?) (continued)

ID	Usability Suggested Questions	Ask this to:
USE233s	What benchmark system or metric will be used?	<i>identify various benchmarks.</i>
USE234s	What user-defined control functions are needed?	<i>identify various controls.</i>
USE235s	How many different ways are expected in order to accomplish the same function?	<i>identify various function usages.</i>
USE236s	What are the expectations for multiple ways to complete the same task?	<i>identify various scenarios.</i>
USE237s	What type of navigation is expected?	<i>identify various types of navigation.</i>
USE238s	What current workarounds are performed to improve usability?	<i>identify various types of workarounds.</i>
USE239s	What user workarounds are performed to offset system deficiencies?	<i>identify various types of workarounds.</i>
USE240R	What processes should be changed to improve usability?	<i>identify process improvement areas.</i>
USE241R	What process areas experience the most data entry errors?	<i>identify process improvement areas.</i>
USE242R	What processes or procedures affect usability?	<i>identify process improvement areas.</i>
USE243R	What manual workarounds could be automated?	<i>identify process improvement areas.</i>
USE244R	What processes are typically performed by novice users? Expert users?	<i>identify processes.</i>
USE245R	What are procedures for users to get help?	<i>identify processes.</i>
USE246R	What are procedures for users to get help desk support?	<i>identify processes.</i>
USE247R	What processes present user safety concerns?	<i>identify processes.</i>
USE248B	What processes or procedures require the most user training?	<i>identify processes.</i>
USE249B	What else should I be asking about usability “processes”?	<i>uncover additional requirements.</i>

5.7 USE



5.8 SUGGESTED READING

Handbook of Software Reliability Engineering, Michael R. Lyu (editor), [Lyu, 1996]. This is a comprehensive collection of articles on the subject of **software reliability**, and an article on **fault-tolerance** system architectures.

Non-functional Requirements in Software Engineering, by Lawrence Chung, et al., [Chung, 2000]. This book includes a concentrated discussion on cataloguing methods and non-functional requirement types, and a thorough perspective on accuracy (**integrity**), security, and performance (**efficiency**).

Requirements Engineering: Processes and Techniques, by Gerald Kotonya and Ian Sommerville, [Kotonya, 1998]. Non-functional requirements are discussed in Chapter 8 including **reliability**, security, safety, **usability**, and performance (**efficiency**).

Security in Computing, 3rd Edition, by Charles P. Pfleeger and Shari Lawrence Pfleeger, [Pfleeger, 2003]. This book is devoted entirely to the exploration of **security** for computer-based systems.

Software Engineering, 8th Edition, by Ian Sommerville, [Sommerville, 2007]. Chapter 3 deals with critical systems and discusses four dimensions of system dependability: **availability**, **reliability**, safety, and security. Chapter 20 covers the development of fault tolerant systems and some techniques for **fault avoidance**, while Chapter 30 deals with system **survivability**.

Software Fault Tolerance Techniques and Implementation, by Laura Pullum, [Pullum, 2001]. This is a comprehensive book on the subject of software fault tolerance (**survivability**).

Software Quality Engineering, by Jeff Tian, [Tian, 2005]. Chapter 22 introduces the general topic of software **reliability** engineering.

Software Reliability Engineering: More Reliable Software Faster and Cheaper, 2nd Edition, by John D. Musa, [Musa, 2004]. This book is a classic guide to the time-saving practice of **reliability**.

Software Requirement Patterns, by Stephen Withall, [Withall, 2007]. Chapter 11 of this book presents **access control** as its own domain category consisting of five requirement patterns: user registration, user authentication, user authorization (specific authorization and configurable authorization), and approval. Chapter 9, section 5, includes a discussion on **availability**.



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Software Requirements, 2nd Edition, by Karl Wiegers, [Wiegers, 2003]. Chapter 12 provides an overview of several software quality attributes including robustness (**survivability**).

Software Requirements: Objects, Functions, & States, by Alan M. Davis, [Davis, 1993]. Davis discusses portability, **reliability**, **efficiency**, and human engineering (**usability**) in Chapter 5, “Specifying Nonbehavioral Requirements.”

Software Requirements: Styles and Techniques, by Soren Lauesen, [Lauesen, 2002]. Chapter 6 presents a variety of categories, including **efficiency** and **usability**.

System Requirements Analysis, Jeffrey O. Grady, [Grady, 2006]. Although Grady’s book focuses on system requirements instead of business requirements, Chapter 3.7, “Specialty Engineering Requirements Analysis,” includes a discussion on **reliability** and maintainability engineering.





6

REVISION REQUIREMENTS

IN THIS CHAPTER:

6.1 Flexibility (FLX)	227
6.2 Maintainability (MNT)	242
6.3 Scalability (SCL)	255
6.4 Verifiability (VER)	265
6.5 Suggested Reading.....	284

HOW EASY IS IT TO CORRECT ERRORS AND ADD ON FUNCTIONS?

Revision requirements define how efficiently the software system can be corrected or fixed when errors occur, and how easily new features can be added. Revision requirements are generally of greater concern to the users who read the software system documentation to understand the design and usage of the system, and change source code or data that drive the system. As such, the user's view of *quality* is a system that is easy to maintain and easy to demonstrate that performance requirements are being met.

The revision group of nonfunctional categories encompasses the concerns of the users supporting a system in a production environment. Revision requirements answer user concerns for the following software qualities:



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

- ◆ **FLEXIBILITY.** How easy is it to modify the system to work in different environments?
- ◆ **Maintainability.** How easy is it to upkeep and repair the system?
- ◆ **Scalability.** How easy is it to expand or upgrade the capabilities of the system?
- ◆ **Verifiability.** How easy is it to show the system performs its functions?

In the same order as listed above, this chapter presents the following for each revision category: definition, brief discussion, examples of requirements, and suggested elicitation questions. The introduction to Part Two explains the anatomy of a category and provides guidance for how to use the suggested questions.



6.1 FLEXIBILITY (FLX)

USER CONCERN: How easy is it to modify the system to work in different environments?

RELATED CATEGORIES: Adaptability, Extendability, Interchangeability, Multiplicability, Tailorability

6.1.1 Flexibility Definition

Flexibility is the ease with which the software can be modified to adapt to different environments, configurations, and user expectations.

6.1.2 Flexibility Discussion

Flexibility concerns the built-in ability of the system to adapt or to be adapted by the users to meet business conditions without major reconstruction of the core system.

Flexibility is often subdivided into the following attributes:

- ◆ **EXTENDABILITY:** concerned with the ability to extend the existing software by “plugging in” (also called “plug-and-play”) extra components. Examples are as follows:
 - a) It shall be possible to add a new delivery option for customer mailing method by developing and “plugging in” the functionality necessary to support that delivery option. The new delivery option shall not require changes to the core software of the system to allow its introduction.
 - b) It shall be possible to deliver mail to customers via United Parcel Service (UPS). To send mail via UPS, the customer must have a valid residential or business address. (UPS will not deliver to post office boxes.)
 - c) There shall be a maintenance function to allow the entry of details about a new delivery option and the editing of existing delivery options.

6.1 FLX



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

- ◆ **INTERCHANGEABILITY:** concerned with the ability to modify the system to switch from using a specific set of components to using another set. For example, the security software shall be configured to switch from workday mode to weekend mode.
- ◆ **MULTIPLICABILITY:** concerned with the ability of the system to handle multiple objects at the same time, each of which typically has its own user interface with data that are distinctly separate from the others. For example, a multi-lingual system will have different user interfaces for each language supported.

Flexibility is generally more applicable when building commercial-off-the-shelf (COTS) software or a software product that is intended to be sold to multiple organizations. It makes it easier to tailor (hence, the term “tailorability”) or customize the software for each organization without affecting core functionality. For example, a word processing software package should be able to adapt to different kinds of printers.

6.1 FLX

When eliciting flexibility requirements, consider the following aspects:

- ◆ **DIFFERENT ORGANIZATIONS.** This might literally be organizations that the software is sold to. It might also be various divisions within a large corporation. Even internal departments or business silos within an organization could have difference business practices. Consider the following differences:
 - ◊ Data types and display format
 - ◊ Organization structure or hierarchy
 - ◊ Employee roles and job descriptions
 - ◊ Accounting and General Ledger entries
 - ◊ Interfaces with external systems
 - ◊ Terminology



- ◆ **DIFFERENT INDUSTRIES.** Is the system being built going to be used in multiple industries? Minor differences such as terminology can affect the flexibility of the software.
- ◆ **DIFFERENT COUNTRIES.** Not only do different languages and currencies exist, but processing differences occur across physical land boundaries and governances. Consider the following differences:
 - ◊ Currency
 - ◊ Numeric display format
 - ◊ Date display format
 - ◊ Language
 - ◊ Dialect
 - ◊ Time zone
 - ◊ Financial year
 - ◊ Calendar
 - ◊ Geographic regions (for example, states, territories, and provinces)
 - ◊ Postal codes
 - ◊ Address format
 - ◊ Telephone numbers
 - ◊ Measurement system
 - ◊ Grammatical punctuation
 - ◊ Cultural differences
 - ◊ Tax, regulatory, and legal regimes
 - ◊ Calculations (for example, bank interest)

6.1 FLX



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

- ◆ **SINGLE SITE.** If you are building software for one specific site or location, this must be clearly communicated or documented as an assumption. In other words, call attention to it to confirm this.
- ◆ **DIFFERENT SITES.** If the software is intended to be used in multiple departments, business silos, divisions, and so on, it is important to identify any diversity or difference between them.

6.1.3 Flexibility Requirement Examples

- a) Provisions shall be made for the future usage of multiple languages. Provision shall include at least the following: 1) The structure of the data store shall be such that multi-lingual support shall not necessitate additional components or the need to replace current components, and 2) A user shall be able to nominate their preferred language when entering their personal information.
- b) No piece of text that might be displayed to a user shall reside in program source code. Every piece of text that a user might see must be modifiable without changing source code. That is, no user-visible text will be “hard-coded.”
- c) The billing system shall be able to process invoices and payments in multiple different currencies. (Currency conversion calculations are to be detailed in business rules and enforced by functional requirements.)
- d) The course curriculum management system shall allow multiple independent courses to be offered with multiple scheduled offerings. Information about courses shall be rigorously separated from each other, and no user shall be able to view or otherwise access information about a course with which they are not connected.
- e) The employee benefits system shall be suitable for use by any ABC Corporation office in any country in which ABC Corporation operates. (It is assumed that all users of the system speak a common language so translation into multiple languages for user interfaces, reports, documentation, and other business materials is not necessary.)
- f) The system shall have the ability to add a new user notification method by developing and “plugging in” the software necessary to support the new method. A new user notification method shall not require changes to the core software of the system to allow its introduction.

6.1 FLX



6.1.4 Flexibility Suggested Questions

DATA (WHAT?)

ID	Flexibility Suggested Questions	Ask this to:
FLX1s	What information is updated based on the event?	<i>correlate information and events.</i>
FLX2s	How long must the information be available to the user?	<i>determine amount of history storage needs.</i>
FLX3s	What information history is customized by the user?	<i>determine customization needs.</i>
FLX4s	How does the quantity of instances vary by specific user?	<i>determine customization needs.</i>
FLX5s	In what level of detail is history needed?	<i>determine if summarization history is needed.</i>
FLX6s	What information is needed to monitor adherence to business policies?	<i>identify processing.</i>
FLX7s	What types of notification methods can be anticipated?	<i>identify response rates.</i>
FLX8R	What information is passed with the system response?	<i>correlate information and events.</i>
FLX9R	What information is heavily accessed?	<i>determine flexibility metrics.</i>
FLX10R	How often can a calculated result change?	<i>identify need to refresh information.</i>
FLX11R	What calculations on the data are performed?	<i>identify calculations that can be stored.</i>
FLX12R	What special information is needed to add a new interface?	<i>identify configurations.</i>
FLX13R	How would users like to see or view information?	<i>identify formats for displaying information.</i>
FLX14R	How would users like to access information?	<i>identify formats for displaying information.</i>
FLX15R	Under what circumstances should information <i>not</i> be transferred?	<i>identify geographic or device restrictions in transmitting data.</i>
FLX16R	How is the information used?	<i>identify how information is manipulated.</i>
FLX17R	What information is needed about each business object?	<i>identify information attributes.</i>

6.1 FLX



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

DATA (WHAT?) (continued)

ID	Flexibility Suggested Questions	Ask this to:
FLX18R	Where does information come from?	<i>identify information input sources.</i>
FLX19R	Where does information go?	<i>identify information output sources.</i>
FLX20R	Who views information?	<i>identify information output sources.</i>
FLX21R	What information is needed to facilitate user customization?	<i>identify information to personalize the user's experience.</i>
FLX22R	What information is needed about each user class?	<i>identify information to request from users.</i>
FLX23R	How often is the calculated result needed?	<i>identify need to store information.</i>
FLX24R	What additional information about each business object would be nice to know?	<i>identify optional data.</i>
FLX25R	What information requires quick access?	<i>identify response time.</i>
FLX26R	What encryption algorithms can be anticipated?	<i>identify security requirements.</i>
FLX27R	What quantity of data can be transmitted at one time?	<i>identify the maximum data capacity by device type or function.</i>
FLX28R	What information do users need to perform a process?	<i>identify user information needs.</i>
FLX29R	What information is captured or modified by this process?	<i>identify user information needs.</i>
FLX30R	What error, informational, and help messages do users need?	<i>identify user information needs.</i>
FLX31R	What additional input file formats can be anticipated?	<i>identify various interfaces.</i>
FLX32R	What additional output file formats can be anticipated?	<i>identify various interfaces.</i>
FLX33R	What other processes receive the same information requests from the users?	<i>minimize the need to enter information multiple times.</i>
FLX34B	What else should I be asking about flexibility "data"?	<i>uncover additional requirements.</i>

6.1 FLX



ROLES (WHO?)

ID	Flexibility Suggested Questions	Ask this to:
FLX35s	Why should users be allowed to set their preferred date presentation?	<i>identify controls extended to users.</i>
FLX36s	Why should users be allowed to set their preferred presentation standards?	<i>identify controls extended to users.</i>
FLX37s	How diverse are individual users allowed to make their system profile?	<i>identify controls extended to users.</i>
FLX38s	What system settings could be added at the user level?	<i>identify controls extended to users.</i>
FLX39s	Who enforces the adherence to business policies?	<i>identify role responsibilities.</i>
FLX40s	Who is responsible for knowing the special information?	<i>identify role responsibilities.</i>
FLX41s	What are the expectations of the role for the one entering special information about the configuration?	<i>identify training needs.</i>
FLX42s	What dialects within languages must be supported?	<i>identify users.</i>
FLX43s	What types of new customers might be added?	<i>identify users.</i>
FLX44s	What is the expectation for adding new types of users?	<i>identify users.</i>
FLX45s	How do these new types of users differ from current users?	<i>identify users.</i>
FLX46s	What types of new users might be added?	<i>identify users.</i>
FLX47s	What is the skill level of those users in remote locations for managing installations and upgrades?	<i>identify users.</i>
FLX48R	Who determines the amount of history to be kept?	<i>determine amount of history storage needs.</i>
FLX49R	Who can validate the information during a test?	<i>identify additional stakeholders.</i>
FLX50R	Who can validate that the process works correctly?	<i>identify additional stakeholders.</i>
FLX51R	What configuration settings are established at the user level?	<i>identify controls extended to users.</i>
FLX52R	Who (role) is responsible for the underlying infrastructure?	<i>identify role responsibilities.</i>
FLX53R	When doing new installations, what is the expected level of expertise for the installer?	<i>identify role responsibilities.</i>
FLX54R	Who (role) is responsible for entering the special information in the configuration?	<i>identify role responsibilities.</i>
FLX55B	What else should I be asking about flexibility “roles”?	<i>uncover additional requirements.</i>

6.1 FLX



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?)

ID	Flexibility Suggested Questions	Ask this to:
FLX56s	How mature is the business for this application?	<i>assess maturity.</i>
FLX57s	How does adherence to business policies vary by user class?	<i>determine customization needs.</i>
FLX58s	What is the range of the degree a business policy might be broken?	<i>determine flexibility metrics.</i>
FLX59s	During what times can the business policy be broken with no action required?	<i>identify business rules.</i>
FLX60s	What is the extent to which the business policy might be broken?	<i>identify business rules.</i>
FLX61s	Which features/functions could be affected by regulatory control?	<i>identify change severities.</i>
FLX62s	Which features/functions must be most flexible to respond to business changes?	<i>identify change velocity.</i>
FLX63s	How frequently can additions be expected?	<i>identify change velocity.</i>
FLX64s	How quickly must the company respond to customer feedback?	<i>identify channels of communication.</i>
FLX65s	How aggressive are the competitors in the same market space?	<i>identify competitors.</i>
FLX66s	What interfaces with external systems can be anticipated?	<i>identify configurations.</i>
FLX67s	How fast must enhancements be completed?	<i>identify enhancements.</i>
FLX68s	What external factors cause business change?	<i>identify external factors.</i>
FLX69s	Which features/functions are most susceptible to external changes?	<i>identify external factors.</i>
FLX70s	As the business evolves, how might this system be affected?	<i>identify longevities.</i>
FLX71s	What is the value of being first to market with an innovation?	<i>identify market pressure.</i>
FLX72s	What user frustrations must be addressed, from the current release, to promote wide-spread use of the system?	<i>identify obstacles/barriers to implementation.</i>
FLX73s	What types of new products might be added?	<i>identify possibilities.</i>
FLX74s	What types of new services might be added?	<i>identify possibilities.</i>
FLX75s	What is the performance expectation for extending the system?	<i>identify possible extensions.</i>

6.1 FLX



PURPOSE (WHY?) (continued)

ID	Flexibility Suggested Questions	Ask this to:
FLX76s	What enhancement capabilities can be expected?	<i>identify potential requests.</i>
FLX77s	How many different types of features/functions are anticipated?	<i>identify processing.</i>
FLX78s	How large could the change be as a result of a regulatory change?	<i>identify regulatory controls.</i>
FLX79s	What is the anticipated rate of requests for the addition of new features/functions?	<i>identify scope of change.</i>
FLX80s	What should be the measurement of a feature added to the product?	<i>identify sizing metrics.</i>
FLX81s	What is the date presentation standard?	<i>identify standards.</i>
FLX82R	How does the additional infrastructure alter the project scope?	<i>assess impact.</i>
FLX83R	How different will these new features/functions be from those expected in this release?	<i>evaluate solutions.</i>
FLX84R	What levels of restricted access to any information exist?	<i>identify access security requirements.</i>
FLX85R	What additional devices can be anticipated?	<i>identify configurations.</i>
FLX86R	What considerations need to be given for configuration information?	<i>identify configurations.</i>
FLX87R	How much configuration special information is managed by the system?	<i>identify configurations.</i>
FLX88R	How quickly should new devices be made available to users?	<i>identify configurations.</i>
FLX89R	How much effort is expected for adding new, common devices?	<i>identify configurations.</i>
FLX90R	What expectations are there for uninstalling the whole system?	<i>identify configurations.</i>
FLX91R	What expectations are there for uninstalling system components?	<i>identify configurations.</i>
FLX92R	What expectations are there for re-installing alternate configurations?	<i>identify configurations.</i>
FLX93R	When external factors cause change, which factors require a system response?	<i>identify external factors.</i>
FLX94R	What is expected for adding a new instance?	<i>identify implementations.</i>
FLX95R	Provide details of hardware and software that are currently used.	<i>identify interfaces.</i>

6.1 FLX



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?) (continued)

ID	Flexibility Suggested Questions	Ask this to:
FLX96R	How will the system manage software license expiration?	<i>identify licensing.</i>
FLX97R	What licensing controls should be implemented?	<i>identify limitations.</i>
FLX98R	What simplifications of multiples are acceptable?	<i>identify limitations.</i>
FLX99R	What means for extending features are available in the current design?	<i>identify limitations.</i>
FLX100R	What design considerations need to be made for adding physical devices?	<i>identify possible devices.</i>
FLX101R	How easy should it be to "plug in" a new interface?	<i>identify possible interfaces.</i>
FLX102R	What is the trade-off between innovative and incremental releases?	<i>identify types of change.</i>
FLX103R	What is the expected combination of modifying existing features and adding new features?	<i>identify types of change.</i>
FLX104R	How difficult is it to modify or extend the application to meet specific needs?	<i>identify various extensions.</i>
FLX105R	What will be the user acceptance test criteria?	<i>prioritize requirements and identify test considerations.</i>
FLX106R	What is the trade-off between flexibility and performance?	<i>relate categories.</i>
FLX107B	What else should I be asking about flexibility "purpose"?	<i>uncover additional requirements.</i>

6.1 FLX



TIMING (WHEN?)

ID	Flexibility Suggested Questions	Ask this to:
FLX108s	What fiscal reporting periods will the system be expected to support?	<i>identify business cycles.</i>
FLX109s	How long is the system expected to be in use?	<i>identify life expectancy.</i>
FLX110s	How often are new sites added for installation?	<i>identify pace of change.</i>
FLX111s	How rapidly is the business changing or evolving?	<i>identify rate of change.</i>
FLX112s	How tolerant of change are the users/customers of the system?	<i>identify rates of change.</i>
FLX113s	What is the trade-off between small, quick changes and large, more time-consuming feature changes?	<i>identify rates of change.</i>
FLX114s	How frequently will different currencies be added?	<i>identify rates of change.</i>
FLX115s	How quickly must new features/functions be completed?	<i>identify rates of change.</i>
FLX116s	What is the lead time notice of regulation change?	<i>identify regulatory compliance.</i>
FLX117s	What is the rate of regulation change?	<i>identify regulatory controls.</i>
FLX118s	When should users be notified that a business policy might potentially be broken?	<i>identify time-triggered events.</i>
FLX119R	How long should information be kept?	<i>determine amount of history storage needs.</i>
FLX120R	When is the expected peak period for each user class?	<i>determine capacity requirements.</i>
FLX121R	When is the expected peak period for each business event?	<i>determine capacity requirements.</i>
FLX122R	When is the expected peak period for each business location?	<i>determine capacity requirements.</i>
FLX123R	What would be the expected performance for peak periods?	<i>determine flexibility metrics.</i>
FLX124R	When is information needed?	<i>determine information supply strategy.</i>
FLX125R	How soon is the history needed for analysis?	<i>determine optional archival methods.</i>
FLX126R	How often is information transmitted?	<i>determine traffic patterns.</i>
FLX127R	How often must the information be uploaded?	<i>determine traffic patterns.</i>
FLX128R	When is the peak period that data transmission occurs?	<i>determine traffic patterns.</i>

6.1 FLX



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

TIMING (WHEN?) (continued)

ID	Flexibility Suggested Questions	Ask this to:
FLX129R	At what specific time must downloaded information be available?	<i>determine traffic patterns.</i>
FLX130R	What information transmissions vary by user class?	<i>determine traffic patterns.</i>
FLX131R	What information transmissions vary by device type?	<i>determine traffic patterns.</i>
FLX132R	What is the expected volume growth of information over the next year? Five years? Ten years?	<i>identify expandability requirements.</i>
FLX133R	How often must the information be updated?	<i>identify how often information should be refreshed.</i>
FLX134R	How fast must installations be completed?	<i>identify installations.</i>
FLX135R	During what time periods is information accessed more than others?	<i>identify patterns in retrieving information.</i>
FLX136R	What history information will be accessed immediately?	<i>identify patterns in retrieving information.</i>
FLX137R	What are the considerations for making upgrades by skipping release levels?	<i>identify rates of change.</i>
FLX138R	What is the sequence in which the events must occur?	<i>identify time-triggered events.</i>
FLX139R	At what point in the process is the information needed?	<i>minimize the need to carry information throughout the process.</i>
FLX140B	What else should I be asking about flexibility "timing"?	<i>uncover additional requirements.</i>

6.1 FLX

LOGISTICS (WHERE?)

ID	Flexibility Suggested Questions	Ask this to:
FLX141S	What kinds of things should the system be able to support multiples of?	<i>identify configurations.</i>
FLX142S	What level of support for multiple features is expected?	<i>identify configurations.</i>
FLX143S	What is different (how diverse) about each new site?	<i>identify diversity.</i>
FLX144S	What other related business areas might this system support?	<i>identify extensions.</i>



LOGISTICS (WHERE?) (continued)

ID	Flexibility Suggested Questions	Ask this to:
FLX145s	How does the system need to support currency conversion?	<i>identify interface dependencies.</i>
FLX146s	What are the plans for installation at remote locations?	<i>identify location complexities.</i>
FLX147s	In what sites will the system be installed?	<i>identify possible breadth.</i>
FLX148s	How rapidly will languages be added?	<i>identify speed of changes.</i>
FLX149s	What current languages are supported?	<i>identify users.</i>
FLX150s	What nationalities and languages must be supported?	<i>identify users.</i>
FLX151s	How broad (and how limited) is the set of environments for which the system should be suitable?	<i>identify various environments.</i>
FLX152s	What are possible new sites in the same industry?	<i>identify various extensions.</i>
FLX153s	What currencies must be supported?	<i>identify various installations.</i>
FLX154s	What are possible new sites in the same country?	<i>identify various location combinations.</i>
FLX155s	What are possible new sites within the organization?	<i>identify various locations.</i>
FLX156R	What would be the priority by geographic location and business processes?	<i>determine traffic patterns.</i>
FLX157R	What is the availability time frame at each location?	<i>determine when backups can be initiated.</i>
FLX158R	How much information would be retrieved by each business location?	<i>identify amount of data transmitted by device type.</i>
FLX159R	How much information is uploaded by each business location?	<i>identify amount of data transmitted by device type.</i>
FLX160R	What are the possible variations within a single installation?	<i>identify configurations.</i>
FLX161R	What additional/alternate database platforms can be anticipated?	<i>identify configurations.</i>
FLX162R	What type of information is transferred by each business location?	<i>identify graphical and textual information transmitted.</i>
FLX163B	What else should I be asking about flexibility "logistics"?	<i>uncover additional requirements.</i>

6.1 FLX



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PROCESS (How?)

ID	Flexibility Suggested Questions	Ask this to:
FLX164s	Describe the current workflow.	<i>define current processes.</i>
FLX165s	How does the process work today?	<i>define current processes.</i>
FLX166s	Why does the process work that way?	<i>define current processes.</i>
FLX167s	Where will the system need agility to remain viable with the expected business growth?	<i>identify areas of rapid response.</i>
FLX168s	What additional functions/features can be anticipated?	<i>identify possible extensions.</i>
FLX169s	How will the system monitor adherence to business policies?	<i>identify processing.</i>
FLX170s	What should occur if a business policy is not followed?	<i>identify processing.</i>
FLX171R	How will the system know that users are who they claim to be?	<i>identify access security requirements.</i>
FLX172R	What combinations of information and processing requires special access privileges?	<i>identify access security requirements.</i>
FLX173R	What are the exceptions to user actions?	<i>identify additional processes.</i>
FLX174R	What is done when an exception occurs?	<i>identify additional processes.</i>
FLX175R	What are exceptions to the normal workflow?	<i>identify alternative processes.</i>
FLX176R	How does the process vary by user?	<i>identify alternative processes.</i>
FLX177R	What alternatives are there to this process?	<i>identify alternative processes.</i>
FLX178R	What other processes are there?	<i>identify alternative processes.</i>
FLX179R	What specific processing limits need to apply to each user class?	<i>identify capacity metrics.</i>
FLX180R	What conditions must be met for this event to occur?	<i>identify event dependencies.</i>
FLX181R	What happens if the conditions are not met?	<i>identify event dependencies.</i>
FLX182R	What would users like changed in the current workflow?	<i>identify possible process improvement areas.</i>

6.1 FLX



PROCESS (HOW?) (continued)

ID	Flexibility Suggested Questions	Ask this to:
FLX183R	When is the process reusable?	<i>identify possible process improvement areas.</i>
FLX184R	How are users added, changed, deleted, reviewed, and monitored?	<i>identify processing restrictions.</i>
FLX185R	What is the system response from the event?	<i>identify processing.</i>
FLX186R	What actions take place when the event is triggered?	<i>identify relationships between events and processes.</i>
FLX187R	How does the activity differ in terms of which user initiates the event?	<i>identify relationships between users and events.</i>
FLX188R	During what processes is information reusable?	<i>identify reusable components.</i>
FLX189R	What would prevent implementing changes in the current workflow?	<i>identify risks and issues.</i>
FLX190B	What else should I be asking about flexibility “processes”?	<i>uncover additional requirements.</i>

6.1 FLX



6.2 MAINTAINABILITY (MNT)

USER CONCERN:	How easy is it to upkeep and repair the system?
RELATED CATEGORIES:	Correctiveness, Modifiability, Reliability, Repairability, Supportability

6.2.1 Maintainability Definition

Maintainability is the ease with which faults in a software system can be found and fixed.

6.2.2 Maintainability Discussion

Maintainability is an indicator of how quickly an unreliable system (one in a failed state) can be brought to a reliable state. Maintainability depends on how easily the software can be understood, changed, and tested.

In general, maintainability requirements need to consider not only the repair of the defect or fault, but recovery from the effects of the fault, so that the system is ready again to do its intended work (see also *reliability* in Chapter 5, “Operation Requirements”).

Software engineers usually differentiate between four kinds of maintenance requirements:

- ◆ **CORRECTIVE MAINTENANCE:** concerned with correcting known defects in the system. Changes are made to the software to remove faults (defects in the code).
- ◆ **PREVENTATIVE MAINTENANCE:** concerned with correcting defects that haven't caused problems yet, but might later on. Changes are made to the software to make it more maintainable.
- ◆ **PERFECTIVE MAINTENANCE:** concerned with expanding the system to meet additional business needs. Changes and enhancements are made to the software as a result of user requests.



- ◆ **ADAPTIVE MAINTENANCE:** concerned with modifying the system to accommodate technology. Changes are made to the software as a consequence of operated system, hardware, or other technical-oriented changes.

The intent of maintainability engineering is to integrate maintainability requirements with system requirements and design parameters to increase the likelihood that the software and hardware are readily maintainable within the designated performance levels at the lowest possible system life-cycle cost. The maintainability requirements are often design criteria or constraints on how the system is to be developed. Such design criteria includes modularization, standardization, accessibility, interchangeability, repair guidance, discard guidance, and test checkpoint placement and quantity. These criteria are stated in both qualitative and quantitative parameters, and are used as guidelines by the design and development team.

As an aid to specifying maintainability requirements, let's turn our attention to several maintainability metrics, which are commonly stated in time parameters (usually in hours). Maintenance time criteria must be carefully defined and agreed upon by the stakeholders. Commonly used maintainability parameters are listed in Table 6-1.

Table 6-1 Maintainability Parameters

6.2 MNT

<i>Parameter</i>	<i>Description</i>	<i>Meaning</i>
MTTR	Mean time to repair	How long it takes to repair the item on average
MTRR	Mean time to remove and replace	Item removal and installation time
MTBM	Mean time between maintenance	Time between maintenance actions
MCT	Mean corrective maintenance time	Time to repair a failure on average (such as MTTR)
MPT	Mean preventative maintenance time	Time required for preventive action



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

When eliciting maintainability requirements, consider the following aspects:

- ◆ **MAINTENANCE PERFORMANCE METRICS** (refer to Table 6-1). These deal with how long it takes to perform maintenance activities.
- ◆ **MAINTENANCE SUPPORT FEATURES**. Support features usually specify things to happen during system maintenance without regard to how much time it takes to do them. For example, “The software vendor shall assign a dedicated resource to perform onsite maintenance.”
- ◆ **SYSTEM MAINTENANCE FEATURES**. Software vendors might build features into the product itself or the support tools to help satisfy maintenance performance requirements with less effort. For example, “The system shall log all user actions,” or “The system shall provide remote diagnostic functions.”
- ◆ **SYSTEM COMPLEXITY**. Some developers contend that maintainability can be measured by the source code. For example, a requirement might state, “The cyclomatic complexity of the source code shall not exceed 6. No method in any object shall exceed 200 lines of code.” Assuming that a source program with 12 nested loops is more difficult to maintain than one with 3 or 4, the cyclomatic complexity measure of code essentially counts the levels of nesting. However, experienced developers find these counts of little value toward predicting maintainability.
- ◆ **DEVELOPMENT PROCESS**. What assessment methods or standards can be used to measure the maintainability of source code? For example, International Organization for Standardization (ISO) programming standards might be applied.
- ◆ **MAINTENANCE PROCESS CYCLE**.
 - ◊ **Reporting**. Notification that a defect or fault has been encountered. Who gets notified? What information is needed to report a problem?
 - ◊ **Analysis**. The problem report is reviewed, and an investigation of the cause of the problem is conducted. Where is the defect that caused the problem? What would it cost to repair the problem? What would happen if nothing was done about the problem?

6.2 MNT



- ◊ **Decision.** Based upon the analysis of the problem, what are possible solutions? Is there a work-around? What actions should be taken to keep users informed? Is additional user training needed as part of the solution? Should the problem be dismissed (it isn't worth fixing)? Is there enough known about the problem and the possible solutions to make a decision? If it is decided to fix the problem, when should it be fixed? Immediately? Fix it in the next release?
 - ◊ **Reply.** Report the outcome of the analysis and decision to affected users, business areas, and other stakeholders.
 - ◊ **Test the solution.** If it is decided to apply a work-around or a system repair, test that it works. Look for related defects that might also be corrected (preventative maintenance).
 - ◊ **Follow through on the decision.** Implement approved changes, repairs, or fixes. Regardless of the decision (for example, repair or don't repair), communicate follow-through actions to affected users and stakeholders.
- ♦ **POSSIBLE PROBLEMS.**
- ◊ **Programming error.** The system doesn't work as intended by the developer.
 - ◊ **Requirement violation.** The system doesn't work as described by the requirements.
 - ◊ **Change request.** The issue was not an expectation of the system (excluded from the scope) at the start of the development or installation project.
 - ◊ **Usability error.** The system can perform what the user wants, but the user cannot operate the system. This might be a user training issue, or the user interface needs to be improved.
 - ◊ **Unstated user expectation.** In practice, it is likely that requirements are missed. If the user has a reasonable expectation regarding the system functions, then such missing requirements are generally viewed as defects.

6.2 MNT



6.2.3 Maintainability Requirement Examples

- a) The customer service call center shall analyze 95% of the problem reports within 2 hours. Items classified as “urgent” shall be repaired within 3 business days in 98% of the reported cases.
- b) The application development process must have a regression test procedure that allows complete re-testing within two business days.
- c) A maintenance developer shall be able to modify existing statements to conform to revised regulations from the federal government with 24 labor hours or less of development and testing effort.
- d) A new consumer type code must be able to be added to the product within 12 business hours.
- e) The system must maintain a service log and, on system start-up, must check if system service is due. If a scheduled service session has not been carried out within 5 calendar days of the scheduled date, the system should discontinue operation.
- f) A development programmer who has at least one year of experience supporting this software application shall be able to add a new product feature, including source code modifications and testing, with no more than one week of labor.
- g) The system shall not be shut down for maintenance more than once in a 24-hour period.

6.2 MNT



6.2.4 Maintainability Suggested Questions

DATA (WHAT?)

ID	Maintainability Suggested Questions	Ask this to:
MNT1S	What concerns exist about vendor-supplied documentation?	<i>identify concerns.</i>
MNT2S	What data elements must have unique values?	<i>identify data needs.</i>
MNT3S	What are special handling requirements for certain data types?	<i>identify exceptions.</i>
MNT4S	What control parameters will be stored and maintained?	<i>identify various controls.</i>
MNT5S	What standards for managing common data elements like name and address fields apply?	<i>maintain compliance with standards.</i>
MNT6S	What date, time, date and time, attributes must be enforced consistently?	<i>maintain compliance with standards.</i>
MNT7S	What is the expected level of precision on calculations?	<i>maintain compliance with standards.</i>
MNT8R	What check digit routines will be used by the system?	<i>identify variety of routines.</i>
MNT9R	What data access methods must be used?	<i>identify various methodologies.</i>
MNT10R	What are the data type attributes that must be enforced consistently?	<i>maintain compliance with standards.</i>
MNT11R	What are the numeric data attributes that must be enforced consistently?	<i>maintain compliance with standards.</i>
MNT12R	What numeric standards apply for: signed, unsigned, units, and rounding?	<i>maintain compliance with standards.</i>
MNT13B	What maintenance records should be kept?	<i>identify maintenance data.</i>
MNT14B	What messaging formats must be used?	<i>identify various formats.</i>
MNT15B	What maintenance schedule information do users need?	<i>identify various methodologies.</i>
MNT16B	What else should I be asking about maintainability "data"?	<i>uncover additional requirements.</i>

6.2 MNT



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

ROLES (WHO?)

ID	Maintainability Suggested Questions	Ask this to:
MNT17S	Who should be notified if scheduled maintenance will be delayed or not performed?	<i>identify role responsibilities.</i>
MNT18S	Who supports the users when maintenance is performed?	<i>identify role responsibilities.</i>
MNT19S	Who needs access security to perform system maintenance?	<i>identify security levels.</i>
MNT20S	What effect do maintenance activities have on customers?	<i>identify users.</i>
MNT21S	What effect do maintenance activities have on users?	<i>identify users.</i>
MNT22R	Who should be notified when time-triggered maintenance tasks fail?	<i>identify role responsibilities.</i>
MNY23R	Who performs routine maintenance?	<i>identify role responsibilities.</i>
MNT24R	Who performs system upgrades?	<i>identify role responsibilities.</i>
MNT25R	Who migrates enhancement releases?	<i>identify role responsibilities.</i>
MNT26R	Who owns the points of interface?	<i>identify role responsibilities.</i>
MNT27R	Who is responsible when an interface with a device fails?	<i>identify role responsibilities.</i>
MNT28R	Who is responsible for the interface?	<i>identify role responsibilities.</i>
MNT29R	Who (what role) will be performing the maintenance?	<i>identify role responsibilities.</i>
MNT30R	Who (role) will be authorized to manage control parameters?	<i>identify role responsibilities.</i>
MNT31R	Who sets the rules for the interface?	<i>identify role responsibilities.</i>
MNT32R	What is the expected skill level of the maintenance team?	<i>identify skills.</i>
MNT33R	How much experience does the current staff have with the devices?	<i>identify skills.</i>
MNT34R	What points of interface exist with vendor components?	<i>identify software interfaces.</i>
MNT35R	What level of access is expected of the maintenance staff?	<i>identify various levels.</i>
MNT36B	Who should be notified of maintenance tasks and schedules?	<i>identify role responsibilities.</i>
MNT37B	Who needs to monitor maintenance records?	<i>identify role responsibilities.</i>
MNT38B	What else should I be asking about maintainability "roles"?	<i>uncover additional requirements.</i>

6.2 MNT



PURPOSE (WHY?)

ID	Maintainability Suggested Questions	Ask this to:
MNT39S	How quickly must enhancements be added?	<i>establish expectations related to flexibility.</i>
MNT40S	What response time is expected for adding simple, moderate, or complex enhancements?	<i>establish expectations related to flexibility.</i>
MNT41S	What is the expected skill level of the enhancement development team?	<i>establish expectations.</i>
MNT42S	How easy should it be to change devices?	<i>establish expectations.</i>
MNT43S	What preventative maintenance should be applied?	<i>identify anticipated actions.</i>
MNT44S	What external mandates for upgrades or changes impact the project?	<i>identify external factors.</i>
MNT45S	What features are most likely to change?	<i>identify potential changes.</i>
MNT46S	What proposed standards or changes to existing standards exist?	<i>identify potential changes.</i>
MNT47S	What anticipated new releases expand functionality?	<i>identify scope of change.</i>
MNT48S	How are early adopters in this line of business served?	<i>identify service expectations.</i>
MNT49S	What are the drivers of business change?	<i>identify sources for change.</i>
MNT50S	What changes to the business organization could affect the system?	<i>identify sources of impact.</i>
MNT51S	How mature is the line of business this application is being built to support?	<i>identify sources of information.</i>
MNT52S	How mature is the industry in this line of business?	<i>identify sources of knowledge.</i>
MNT53S	What types of mandatory changes can be expected?	<i>identify types of changes and reduce response.</i>
MNT54S	What business rules could change that would affect the system?	<i>identify volatile business rules to ease maintenance for change.</i>
MNT55S	What date management and terminology will be used consistently by the system?	<i>maintain compliance with standards.</i>
MNT56R	What degree of complexity is anticipated for add-on enhancements?	<i>identify complexities.</i>
MNT57R	What is the importance of staying on current releases?	<i>identify configurations.</i>
MNT58R	What hardware devices or software features are expected to require the most frequent maintenance?	<i>identify points of failure.</i>
MNT59R	What audits are performed on maintenance service levels?	<i>identify standards.</i>

6.2 MNT



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?) (continued)

ID	Maintainability Suggested Questions	Ask this to:
MNT60R	How stable is the configuration environment?	<i>identify various configurations and possible impacts.</i>
MNT61R	What user interface and presentation design standards should be in place?	<i>maintain compliance with standards.</i>
MNT62R	What are standard display formats for all data attribute types?	<i>maintain compliance with standards.</i>
MNT63R	What file format and database interface standards must be in place?	<i>maintain compliance with standards.</i>
MNT64R	What are the industry standard messaging formats?	<i>maintain compliance with standards.</i>
MNT65R	What protocols must be used?	<i>maintain compliance with standards.</i>
MNT66R	What are the industry standards for the protocols?	<i>maintain compliance with standards.</i>
MNT67B	What will be needed to keep business running while software or hardware is being replaced or upgraded?	<i>identify business impacts.</i>
MNT68B	What concerns exist about vendor supplied software?	<i>identify concerns.</i>
MNT69B	What are the cost expectations for maintenance enhancements?	<i>identify constraints.</i>
MNT70B	What are the known industry standards?	<i>identify controlling standards.</i>
MNT71B	What trade-offs are there between maintenance and availability?	<i>identify possible trade-offs.</i>
MNT72B	What trade-offs are there between maintenance and reliability?	<i>identify possible trade-offs.</i>
MNT73B	What trade-offs are there between maintenance and flexibility?	<i>identify possible trade-offs.</i>
MNT74B	What elements must be defined to develop an acceptable service-level agreement?	<i>identify priorities.</i>
MNT75B	What changes in the regulations could affect the system?	<i>identify regulations.</i>
MNT76B	What service level agreements are in place with the customers?	<i>identify various agreements.</i>
MNT77B	What vendor agreements are in place on the software?	<i>identify various agreements.</i>
MNT78B	What regulations govern this application?	<i>identify various regulations.</i>
MNT79B	What else should I be asking about maintainability "purpose"?	<i>uncover additional requirements.</i>

6.2 MNT



TIMING (WHEN?)

ID	Maintainability Suggested Questions	Ask this to:
MNT80s	During what time frames is routine maintenance unacceptable?	<i>identify effects on availability.</i>
MNT81s	When should routine maintenance be performed?	<i>identify effects on availability.</i>
MNT82s	How frequently will enhancements be added to the system?	<i>identify rates of change related to flexibility.</i>
MNT83s	How fast-paced are the competitors' responses?	<i>identify response rates.</i>
MNT84s	What lead-time notice is there for mandatory changes?	<i>identify response to demand.</i>
MNT85s	What are the seasons in this line of business?	<i>identify seasonality and understand the impacts.</i>
MNT86s	What time standard has been established?	<i>identify timing expectations.</i>
MNT87s	When could future installations happen?	<i>identify timing.</i>
MNT88s	What are the responses necessary due to seasonality?	<i>identify timing.</i>
MNT89s	What industry changes are anticipated?	<i>identify trends that could impact implementation.</i>
MNT90s	What changes are anticipated in the industry or organization?	<i>identify trends.</i>
MNT91s	What elements or features change frequently due to external events, decisions, or regulations?	<i>identify volatility and reduce impact.</i>
MNT92R	When will it be necessary to perform maintenance on hardware devices?	<i>clarify timing.</i>
MNT93R	How frequently do the devices require repair?	<i>identify frequency of failure.</i>
MNT94R	How often are maintenance releases expected?	<i>identify frequency.</i>
MNT95R	What specific time frames during the day should maintenance activities be avoided?	<i>identify maintenance schedule concerns.</i>
MNT96R	What peak periods in processing cause maintenance concerns?	<i>identify maintenance schedule concerns.</i>
MNT97R	What maintenance is triggered by time of day?	<i>identify maintenance schedule concerns.</i>
MNT98R	What maintenance is triggered by events?	<i>identify maintenance schedule concerns.</i>

6.2 MNT



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

TIMING (WHEN?) (continued)

ID	Maintainability Suggested Questions	Ask this to:
MNT99R	What maintenance is performed automatically by the system?	<i>identify maintenance schedule concerns.</i>
MNT100R	How often do upgrades to interfaces get released?	<i>identify pace of change.</i>
MNT101R	How often do upgrades to components get released?	<i>identify rates of change.</i>
MNT102B	How often do regulations change?	<i>identify frequency.</i>
MNT103B	What else should I be asking about maintainability "timing"?	<i>uncover additional requirements.</i>

LOGISTICS (WHERE?)

ID	Maintainability Suggested Questions	Ask this to:
MNT104S	What time zone is the standard for the system?	<i>identify locations.</i>
MNT105S	Where is the most maintenance needed?	<i>identify locations.</i>
MNT106S	What maintenance support is expected at each location?	<i>identify locations.</i>
MNT107S	What maintenance is performed at each location?	<i>identify locations.</i>
MNT108R	Where is the source code escrowed to prevent emergency situations from vendor-supplied software?	<i>identify emergency recovery.</i>
MNT109R	What changes in the configuration environment could affect the system?	<i>identify environmental impacts.</i>
MNT110R	What will be the environment of use?	<i>identify environments.</i>
MNT111R	What will be the development environment used?	<i>identify environments.</i>
MNT112R	What level of maintenance is required at each business location?	<i>identify locations.</i>
MNT113R	What routine maintenance is performed at each business location?	<i>identify locations.</i>
MNT114R	What maintenance is performed offsite?	<i>identify locations.</i>
MNT115R	What is the schedule for upgrades or releases at each location?	<i>identify locations.</i>

6.2 MNT



LOGISTICS (WHERE?) (continued)

ID	Maintainability Suggested Questions	Ask this to:
MNT116R	What time-triggered maintenance is performed at each location?	<i>identify maintenance activities by location.</i>
MNT117R	What event-triggered maintenance is performed at each location?	<i>identify maintenance activities by location.</i>
MNT118R	By location, what maintenance is done manually?	<i>identify maintenance activities by location.</i>
MNT119R	By location, what maintenance is automated?	<i>identify maintenance activities by location.</i>
MNT120R	On what types of operating environments/hardware configurations will this system be deployed?	<i>identify various configurations.</i>
MNT121B	What else should I be asking about maintainability “logistics”?	<i>uncover additional requirements.</i>

PROCESS (How?)

ID	Maintainability Suggested Questions	Ask this to:
MNT122s	Under what events or conditions should this system deviate from any standard?	<i>identify potential challenges.</i>
MNT123s	Under what conditions is the industry standard messaging format not adequate?	<i>identify potential for noncompliance.</i>
MNT124s	Under what conditions is the industry standard protocol not adequate?	<i>identify potential for noncompliance.</i>
MNT125s	What features are strongest from the competitors?	<i>identify pressured features.</i>
MNT126s	What features are most appealing to the customers?	<i>identify priority features.</i>
MNT127s	What workaround processes are performed while the system is being maintained?	<i>identify process impact.</i>
MNT128s	What functions are limited or unavailable during maintenance?	<i>identify process impact.</i>
MNT129R	What types of support is needed from the vendors?	<i>identify dependencies.</i>
MNT130R	What vendor devices will be used in this system?	<i>identify device interfaces.</i>
MNT131R	What types of harsh conditions are hardware components subject to that could cause frequent failure?	<i>identify exposure.</i>

6.2 MNT



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PROCESS (HOW?) (continued)

ID	Maintainability Suggested Questions	Ask this to:
MNT132R	What provisions exist for modification of the industry standard messaging format?	<i>identify potential changes.</i>
MNT133R	What provisions exist for modification of the industry standard protocol?	<i>identify potential changes.</i>
MNT134R	What are current procedures and processes for system maintenance?	<i>identify processes.</i>
MNT135R	What maintenance activities are performed automatically?	<i>identify processes.</i>
MNT136R	What maintenance activities are performed manually?	<i>identify processes.</i>
MNT137R	What is the expected interruption when adding, changing, or repairing a hardware device?	<i>identify response expectations.</i>
MNT138R	What types and varieties of hardware devices will be used on the system?	<i>identify various devices.</i>
MNT139B	What is the expectation for repairing related, non-repaired defects?	<i>identify various defects and level expectations.</i>
MNT140B	What else should I be asking about maintainability “processes”?	<i>uncover additional requirements.</i>

6.2 MNT



6.3 SCALABILITY (SCL)

USER CONCERN: How easy is it to expand or upgrade the system's capabilities?

RELATED CATEGORIES: Augmentability, Expandability, Flexibility

6.3.1 Scalability Definition

Scalability is the degree to which the software system is able to expand its processing capabilities upward and outward to support business growth.

6.3.2 Scalability Discussion

Scalability requirements specify a way in which a system must be able to expand without significant impact, usually to accommodate growth in business volume. The system must scale to handle increased usage without a proportional increase in all the supporting elements.

Scalability is the degree to which the system is capable of taking on more when the load increases, and having ways to react without turning to drastic means for change. Scalability means that the development of a system (software and hardware) must not restrict the business to any particular level of volume. That is, development of the system should take into account any possibility of future changes, and make adequate provisions for that eventuality. The users should identify the most likely areas for future growth.

6.3 SCL

When eliciting scalability requirements, consider the following aspects:

- ◆ **ABILITY TO COPE WITH INCREASING PROCESSING LOAD.**
 - ◊ Effect on data inquiries and report generation.
 - ◊ Maintenance and administration tasks.
 - ◊ Frequency, timing, and resources needed for installations and upgrades.



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

- ◊ Spreading the processing load across multiple systems or platforms (expand outward).
- ◊ Boosting the processing power of a specific system (expand upward).
- ◆ EXPANDING BUSINESS LOCATIONS.
 - ◊ Minimize redundant processing and administration.
 - ◊ Coordination of resources (data, people, software, and hardware).
- ◆ POSSIBLE CAUSE FOR DEGRADATION.
 - ◊ Support resources.
 - ◊ Downtime for routine maintenance.
 - ◊ Duration of processing (how long does it take?).
 - ◊ Contention for resources.
- ◆ RECYCLE HARDWARE TO MINIMIZE WASTE.
 - ◊ Ability to add hardware instead of replacing hardware.

6.3 SCL



6.3.3 Scalability Requirement Examples

- a) The elapsed duration of time required to produce any statement or report showing information about transactions shall be based upon how much data are presented rather than the total quantity of stored data.
- b) The effort needed to administer the payroll system (as measured in hours per month of system administrators' time) shall not increase with an increase in the number of employees. If there is a significant increase in system operation work, it shall be proportionately less than an increase in the number of employees.
- c) The payroll system shall be scalable to support unlimited growth in the number of employees.
- d) The business rules repository shall be scalable to manage an unrestricted number of additional rules.
- e) The travel reservation system shall be scalable to accommodate its use by an unlimited number of agency offices world wide.
- f) The claims system shall support all assigned adjustors following any catastrophic event.
- g) The account management system shall support unlimited customer, account, and transaction relationships.
- h) The transaction authorization system shall scale to potential hourly spikes of 1,000% in authorization requests during peak holiday shopping.

6.3 SCL



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

6.3.4 Scalability Suggested Questions

DATA (WHAT?)

ID	Scalability Suggested Questions	Ask this to:
SCL1s	What scalability data history must be stored?	<i>identify historical data.</i>
SCL2s	What analysis reporting of history data is needed?	<i>identify historical data.</i>
SCL3s	What scalability data are input to the system?	<i>identify inputs/outputs.</i>
SCL4s	What scalability data are output from the system?	<i>identify inputs/outputs.</i>
SCL5s	What scalability data do management request?	<i>identify reporting needs.</i>
SCL6s	What scalability data are necessary for audit purposes?	<i>identify reporting needs.</i>
SCL7s	What scalability data are necessary for regulatory and compliance purposes?	<i>identify reporting needs.</i>
SCL8s	What demographic reporting is needed?	<i>identify reporting needs.</i>
SCL9R	What sources could provide analysis data?	<i>identify historical data.</i>
SCL10R	What historical data are stored currently?	<i>identify historical data.</i>
SCL11R	What volume data are tracked currently?	<i>identify reporting needs.</i>
SCL12B	What data would help to identify growth trends or patterns?	<i>identify historical data.</i>
SCL13B	What else should I be asking about scalability "data"?	<i>uncover additional requirements.</i>

6.3 SCL



ROLES (WHO?)

ID	Scalability Suggested Questions	Ask this to:
SCL14s	Who audits system growth and expansion?	<i>identify additional stakeholders.</i>
SCL15s	What regulatory or government agencies must be notified of expansion?	<i>identify additional stakeholders.</i>
SCL16s	Who should receive scalability reports?	<i>identify role responsibilities.</i>
SCL17s	Who is responsible for the business architecture?	<i>identify role responsibilities.</i>
SCL18R	Who is responsible for monitoring software and hardware expansion?	<i>identify additional stakeholders.</i>
SCL19R	What sources could provide analysis data?	<i>identify additional stakeholders.</i>
SCL20R	Who interfaces with the system that is being upgraded?	<i>identify additional stakeholders.</i>
SCL21R	Who is responsible for the system architecture?	<i>identify role responsibilities.</i>
SCL22B	Who should participate in a system inspection/review?	<i>identify additional stakeholders.</i>
SCL23B	Who monitors growth trends and patterns?	<i>identify role responsibilities.</i>
SCL24B	What else should I be asking about scalability “roles”?	<i>uncover additional requirements.</i>

6.3 SCL



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?)

ID	Scalability Suggested Questions	Ask this to:
SCL25s	How elastic is the anticipated scalability?	<i>identify combinations of metrics or indicators.</i>
SCL26s	How easy must it be to expand the system?	<i>identify concerns.</i>
SCL27s	What product or feature is most likely to stimulate growth?	<i>identify functions.</i>
SCL28s	What is the extreme scale of system components?	<i>identify functions.</i>
SCL29s	What is the extreme scale of system features?	<i>identify functions.</i>
SCL30s	What is the extreme scale of system functions?	<i>identify functions.</i>
SCL31s	What magnitude of increase in customers is expected?	<i>identify growth potential.</i>
SCL32s	What magnitude of increase in users is expected?	<i>identify growth potential.</i>
SCL33s	What magnitude of increase in products is expected?	<i>identify growth potential.</i>
SCL34s	What magnitude of increase in sales is expected?	<i>identify growth potential.</i>
SCL35s	What magnitude of increase in transactions is expected?	<i>identify growth potential.</i>
SCL36s	What magnitude of increase in inputs is expected?	<i>identify growth potential.</i>
SCL37s	What magnitude of increase in outputs is expected?	<i>identify growth potential.</i>
SCL38s	How rapidly might business volume increase?	<i>identify growth potential.</i>
SCL39s	Why is this level of scalability needed?	<i>identify influencing activity.</i>
SCL40s	What is the current maximum number of: users, customers, accounts, and transactions?	<i>identify metrics.</i>
SCL41s	What is the basis for scalability projections?	<i>identify metrics.</i>
SCL42s	What business loss would result from not being scalable?	<i>identify risks.</i>
SCL43s	What business could be lost due to expansion?	<i>identify risks.</i>
SCL44s	What feature of the business is most likely to grow the fastest?	<i>identify trends or patterns.</i>
SCL45s	What is the most probable source for growth: users, customers, transactions?	<i>identify trends or patterns.</i>

6.3 SCL



PURPOSE (WHY?) (continued)

ID	Scalability Suggested Questions	Ask this to:
SCL46s	How might business spurts be one-time events or sustainable?	<i>identify trends or patterns.</i>
SCL47s	How might business spurts be sporadic or on-going occurrences?	<i>identify trends or patterns.</i>
SCL48s	What feature of the business is most likely to grow the most?	<i>identify trends or patterns.</i>
SCL49R	What will enable an expansion of the software without a disproportionate increase in hardware?	<i>identify configurations.</i>
SCL50R	What will enable an expansion of the software without a disproportionate increase in support personnel?	<i>identify constraints.</i>
SCL51R	What is the affect of vertical (software) scaling?	<i>identify growth potential.</i>
SCL52R	What is the affect of horizontal (hardware) scaling?	<i>identify growth potential.</i>
SCL53B	What indicators of growth should be monitored?	<i>identify cycles.</i>
SCL54B	How can the system scale and continue to meet performance and reliability requirements?	<i>identify possible trade-offs.</i>
SCL55B	What is the trade-off between scalability and maintainability?	<i>identify possible trade-offs.</i>
SCL56B	What is the trade-off between scalability and flexibility?	<i>identify possible trade-offs.</i>
SCL57B	What is the trade-off between scalability and efficiency?	<i>identify possible trade-offs.</i>
SCL58B	What else should I be asking about scalability "purpose"?	<i>uncover additional requirements.</i>

6.3 SCL



TIMING (WHEN?)

ID	Scalability Suggested Questions	Ask this to:
SCL59s	How soon after implementation might business volume increase?	<i>identify cycles.</i>
SCL60s	How long does the peak volume last?	<i>identify cycles.</i>
SCL61s	What response time expectations are there for peak periods?	<i>identify metrics.</i>
SCL62s	How are peak usage volumes measured?	<i>identify metrics.</i>
SCL63s	What noticeable difference between peak periods and normal activity is acceptable?	<i>identify metrics.</i>
SCL64s	What event could cause that surprise development?	<i>identify metrics.</i>
SCL65s	How fast could a surprise happen?	<i>identify metrics.</i>
SCL66s	When do peak processing volumes occur?	<i>identify processing volumes.</i>
SCL67s	What throughput volumes are expected during peak periods?	<i>identify processing volumes.</i>
SCL68s	How fast does the volume change from peak to normal levels?	<i>identify rates of change.</i>
SCL69R	When are additions or upgrades typically implemented?	<i>identify processing cycles.</i>
SCL70R	When should system expansions be avoided if possible?	<i>identify processing cycles.</i>
SCL71R	When have upgrades been scheduled in the past?	<i>identify processing cycles.</i>
SCL72R	What is the average duration of an upgrade install?	<i>identify processing time.</i>
SCL73B	When are specific periods when additions or upgrades should be avoided?	<i>identify processing cycles.</i>
SCL74B	What else should I be asking about scalability "timing"?	<i>uncover additional requirements.</i>

6.3 SCL



LOGISTICS (WHERE?)

ID	Scalability Suggested Questions	Ask this to:
SCL75s	What locations could be problematic with sudden increase in business volume?	<i>identify location issues.</i>
SCL76s	What types of problems could arise due to variances in locations?	<i>identify location issues.</i>
SCL77s	What types of organizational structures are needed to be aware of as the system scales?	<i>identify organizational units.</i>
SCL78s	What area has been the most constrained so far but is building momentum?	<i>identify organizational units.</i>
SCL79s	What area has been dismissed for growth potential but could be a surprise development?	<i>identify organizational units.</i>
SCL80s	What could cause volume projections to change in a particular location?	<i>identify rates of change.</i>
SCL81s	What locations could increase in volume faster than other locations?	<i>identify volumes and locations.</i>
SCL82s	What locations appear to have static volume?	<i>identify volumes and locations.</i>
SCL83R	Where will additional software/hardware be needed to expand the business?	<i>identify location resources.</i>
SCL84R	Where will additional maintenance and administering be needed to expand the business?	<i>identify location resources.</i>
SCL85R	Where can software be installed on an existing machine?	<i>identify location resources.</i>
SCL86R	Where can software be upgraded on an existing machine?	<i>identify location resources.</i>
SCL87B	Where might corporate restructuring take place to accommodate growth?	<i>identify organizational units.</i>
SCL88B	What else should I be asking about scalability "logistics"?	<i>uncover additional requirements.</i>

6.3 SCL



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PROCESS (How?)

ID	Scalability Suggested Questions	Ask this to:
SCL89s	What system component is the most critical to growth of the business?	<i>identify critical components.</i>
SCL90s	In what ways must the system keep performing just as well as it grows?	<i>identify critical components.</i>
SCL91s	What component will be the greatest challenge to scale?	<i>identify functions.</i>
SCL92s	What specific features are needed so that the system will continue to be as easy to use as volumes grow?	<i>identify functions.</i>
SCL93s	What specific features cannot degrade as the system scales?	<i>identify functions.</i>
SCL94s	What aspect of the system needs to be scalable?	<i>identify functions.</i>
SCL95s	What business activity could lead to rapid increase in business volume?	<i>identify influencing activity.</i>
SCL96s	What processes in which locations are most susceptible to peak volume pressure?	<i>identify process and location relationships.</i>
SCL97s	What business processes are vulnerable to volume pressure?	<i>identify process and volume relationships.</i>
SCL98s	What events could cause sudden increase in business volume?	<i>identify rates of change.</i>
SCL99R	What will enable an expansion of the software without a disproportionate increase in ancillary system support? (backup, security, restore, recovery)	<i>identify configurations.</i>
SCL100R	How dynamically can the system be expanded?	<i>identify configurations.</i>
SCL101R	What third-party components are in use/planned for use that will have scalability issues?	<i>identify functions.</i>
SCL102R	What architectural feature could be the most difficult to scale?	<i>identify functions.</i>
SCL103R	Where are the weak points in the architecture?	<i>identify functions.</i>
SCL104R	What is the difference between peak maximum and “normal” volumes in a process cycle?	<i>identify processing volumes.</i>
SCL105B	What specific features are needed by the system to help it be scalable?	<i>identify functions.</i>
SCL106B	What else should I be asking about scalability “processes”?	<i>uncover additional requirements.</i>

6.3 SCL



6.4 VERIFIABILITY (VER)

USER CONCERN:	How easy is it to show that the system performs its functions?
RELATED CATEGORIES:	Implementability, Installability, Testability

6.4.1 Verifiability Definition

Verifiability is the extent to which tests, analysis, and demonstrations are needed to prove that the software system will function as intended.

6.4.2 Verifiability Discussion

Whether buying or building, the software system must be checked to see if it meets the requirements and delivers the functionality expected by the users. **Verification and Validation** (V & V) is a name given to the process of checking the software. **Validation** is checking that you're building the right product, while **verification** is checking that you're building the product right. That is, validation is concerned with making sure the software meets the user expectations, and verification is concerned with making sure the software meets its specified functional and nonfunctional requirements.

There are two complementary approaches used with the verification and validation process to check the software system:

6.4 VER

- (1) **Software inspections, also known as peer reviews, are performed to check the system specifications.** Several development deliverables are reviewed such as the requirements document, design models and diagrams, and program source code. Inspections are considered to be static techniques as you don't need to run the software.



- (2) **Software testing involves running a version of the software with test data or components.** Testing is a dynamic technique of verification and validation. The outputs of the testing are examined to check if the software is performing as required.

Software inspections are a valuable technique, one that should not be overlooked or minimized during the implementation process. Several studies have demonstrated that inspections are far more effective than testing for error or defect discovery. Many reports show findings of 60 percent error detection through inspections and some as much as 90 percent. There are three major advantages of software inspections over software testing:

- ◆ **INCOMPLETE VERSIONS OF THE SYSTEM CAN BE INSPECTED WITH LITTLE OR NO ADDITIONAL COST.** On the other hand, testing an incomplete system requires special processes in order to test the part that is available. These processes and specialized test data add to the system development costs.
- ◆ **AS A STATIC TECHNIQUE, INSPECTIONS CAN DISCOVER MANY ERRORS IN A SYSTEM** and don't have to be concerned with interactions between errors. Conversely, during testing, errors can camouflage or hide other errors. Once an error is discovered it is difficult to know if other output anomalies are due to new error or a side effect of the previous discovered error.
- ◆ **INSPECTIONS CAN BE USED FOR PURPOSES BROADER THAN ERROR DETECTION.** For instance, the software representation can be reviewed for compliance to standards. Program source code can be reviewed for inappropriate algorithms and poor programming structure that could make the system difficult to maintain, update, and reuse.

6.4 VER

Wallace and Ippolito describe a number of software verification and validation techniques, which are summarized in Table 6-2. The table does not represent an all-inclusive list of techniques.



Table 6-2 Software Verification and Validation Techniques [Wallace, 1996]

TECHNIQUE	Requirements Verification	System Test (Requirements Validation)	Technique Description
Algorithm analysis	◆		Examines the logic and accuracy of the software requirements by translating algorithms into some language or structured format. The analysis involves re-deriving equations or evaluating the suitability of specific numerical techniques.
Back-to-back testing		◆	Detects test failures by comparing the output of two or more programs implemented to the same specification. The same input data are applied to two or more program versions and their outputs are compared to detect anomalies.
Boundary value analysis		◆	Detects and removes errors occurring at parameter limits or boundaries. The input domain of the program is divided into a number of input classes. The tests should cover the boundaries and extremes of the classes.
Consistency analysis	◆		Compares the requirements of any existing software with the new software requirements to ensure consistency.
Control flow analysis	◆		Transforms text describing software requirements into graphic flows where they can be examined for correctness. Control flow analysis is used to show the hierarchy of main routines and their sub-functions and checks that the proposed control flow is free of problems.
Coverage analysis		◆	Measures how much of the structure of a unit or system has been exercised by a given set of tests.
Database analysis	◆		Ensures that the database structure and access methods are compatible with the logical design. It is performed on programs with significant data storage to ensure that common data and variable regions are used consistently between all calling routines, that data integrity is enforced and that no data or variable can be accidentally overwritten by overflowing data tables, and that data typing and use are consistent throughout the program.
Dataflow analysis	◆		Dataflow analysis is important for designing the high-level architecture of applications. It can check for variables that are read before they are written, written more than once without being read, and written but never used.

6.4 VER



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Table 6-2 Software Verification and Validation Techniques [Wallace, 1996] (continued)

TECHNIQUE	Requirements Verification	System Test (Requirements Validation)	Technique Description
Decision tables	◆		Provide a clear and coherent analysis of complex logical combinations and relationships. This method uses two-dimensional tables to concisely describe logical relationships between Boolean program variables.
Error seeding		◆	Determines whether a set of test cases is adequate by inserting ("seeding") known error types into the program and executing it with the test cases. If only some of the seeded errors are found, the test case set is not adequate.
Event tree analysis	◆		Event tree analysis uses a bottom-up approach to model the effects of an event that may have serious repercussions. The initiation event is the root of the event tree. Two lines are drawn from the root, depicting the positive and negative consequences of the event. This is done for each subsequent consequence until all consequences are considered.
Finite state machines	◆		Check for incomplete and inconsistent software requirements by modeling the software in terms of its states, inputs, and actions.
Functional testing		◆	Executes part or all of the system to validate that the user requirement is satisfied.
Inspections	◆		Inspections are evaluation techniques whereby the software requirements, software design, or code is examined by a person or group other than the author to detect faults, violations of development standards, and other problems.
Interface analysis	◆	◆	Interface analysis is a static analysis technique. It is used to demonstrate that the interfaces of sub-programs do not contain any errors that lead to failures in a particular application of the software.
Interface testing		◆	Interface testing is a dynamic analysis technique. It is similar to interface analysis, except test cases are built with data that test all interfaces.
Mutation analysis		◆	Determines the thoroughness with which a program has been tested, and in the process detects errors. This procedure involves producing a large set of versions or "mutations" of the original program, each derived by altering a single element of the program. Each mutant is then tested with a given collection of test data sets.



CHAPTER 6: REVISION REQUIREMENTS

Table 6-2 Software Verification and Validation Techniques [Wallace, 1996] (continued)

TECHNIQUE	Requirements Verification	System Test (Requirements Validation)	Technique Description
Performance testing		◆	Measures how well the software system executes according to its required response times, CPU usage, and other quantified features in operations.
Prototyping		◆	Prototyping helps to examine the probable results of implementing software requirements. Examination of a prototype may help to identify incomplete or incorrect software requirements and may also reveal if any software requirements will not result in desired system behavior.
Regression analysis and testing	◆	◆	Regression analysis and testing is used to re-evaluate software requirements and software design issues whenever any significant code change is made. It involves retesting to verify that the modified software still meets its specified requirements.
Requirements parsing	◆		Involves examination to ensure that each software requirement is defined unambiguously by a complete set of attributes (for example, initiator of an action, source of the action, the action, the object of the action, constraints).
Reviews	◆	◆	Reviews are meetings at which the software requirements, software design, code, or other products are presented to the user, sponsor, or other interested parties for comment and approval, often as a prerequisite for concluding a given activity of the software development process.
Simulation	◆		Simulation is used to evaluate the interactions of large, complex systems with many hardware, user, and other interfacing software units. Simulation uses an executable model to examine the behavior of the software.
Software failures mode, effects and critical analysis (SFMECA)	◆		Reveals weak or missing software requirements by using inductive reasoning to determine the effect on the system of a unit (includes software instructions) failing in a particular failure mode. A matrix is developed for each unit depicting the effect on the system of each unit's failure in each failure mode.
Software fault tree analysis	◆		Identifies and analyzes software safety requirements. It is used to determine possible causes of known hazards. Its purpose is to demonstrate that the software will not cause a system to reach an unsafe state, and to discover what environmental conditions would allow the system to reach an unsafe state.

6.4 VER



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Table 6-2 Software Verification and Validation Techniques [Wallace, 1996] (continued)

TECHNIQUE	Requirements Verification	System Test (Requirements Validation)	Technique Description
Stress testing		◆	Tests the response of the system to extreme conditions to identify vulnerable points within the software, and to show that the system can withstand normal workloads.
Test certification		◆	Test certification ensures that reported test results are the actual finding of the tests. Test-related tools, media, and documentation are certified to ensure maintainability and repeatability of tests. This technique is also used to show that the delivered software product is identical to the software product that was subjected to V & V.
Walkthroughs	◆	◆	Walkthroughs are similar to inspections, but less formal. A walkthrough is an evaluation technique in which a designer or programmer leads one or more other members of the development team through a segment of software design or code, while the other members ask questions and make comments about technique and style, and identify possible errors, violations of development standards, and other problems.

Furthermore, there are a number of words that might be flagged as unverifiable when trying to write requirements. Several examples of these potentially unverifiable words, along with suggested substitutes, are listed in Table 6-3. Additional problematic words are found in the examples of re-written requirement statements provided in Table 6-4 on page 272. In the manner used in these examples, the term **verifiable** is synonymous with **testable**. Here the term is used to describe an attribute of the requirement. A requirement is *verifiable* (testable) if a test can be devised to demonstrate correct implementation.

6.4 VER



Table 6-3 Certain Words Flag Unverifiable Requirements [Hooks, 2001]

<i>Unverifiable Words</i>	<i>Possible Substitutes</i>
Flexible	<ul style="list-style-type: none"> • Bending threshold or spring constant • Features that will cover anticipated changes from operational concepts
Easy or user-friendly	<ul style="list-style-type: none"> • A maximum number of steps to perform an operation • An educational standard reference • A list of features found on similar popular products • Menus or prompts to guide user
Accommodate	<ul style="list-style-type: none"> • Precise definition of accommodation from operational concepts
Ad hoc	<ul style="list-style-type: none"> • List of features that support all uses anticipated in operational concepts
Safe	<ul style="list-style-type: none"> • List of features that prevent harm from operator error anticipated in operational concepts • References to specific safety standards
Sufficient or adequate	<ul style="list-style-type: none"> • Quantities or other dimensions
Useable	<ul style="list-style-type: none"> • Exact features needed
When required or if required	<ul style="list-style-type: none"> • Exact circumstances • Triggering events from operational concepts
Fast or quickly	<ul style="list-style-type: none"> • Minimum acceptable speed
Portable	<ul style="list-style-type: none"> • Dimensions and weight • Description of desired carrying means • Operating systems that the software must run on
Light-weight	<ul style="list-style-type: none"> • Maximum acceptable weight
Small	<ul style="list-style-type: none"> • Maximum acceptable dimensions
Large	<ul style="list-style-type: none"> • Minimum acceptable dimensions
Easily, clearly, or other “-ly” words	<ul style="list-style-type: none"> • Quantities appropriate for the verb that the “-ly” word modifies (for example, replace “fit easily” with “fit in X by Y by Z space”)
Maximize, minimize, optimize, or other “-ize” words	<ul style="list-style-type: none"> • Limits, greater than or equal to, less than or equal to

6.4 VER



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

Table 6-4 Rewrites for Unverifiable Requirements [Hooks, 2001]

<i>Unverifiable</i>	<i>Verifiable</i>
ABC shall support ad hoc queries.	<ul style="list-style-type: none">ABC shall retrieve up to five user-specified data items per user query.ABC shall retrieve those records meeting the criteria in any legal Standard Query Language user query.
The ZZ database shall be flexible.	<ul style="list-style-type: none">The ZZ database shall have eight user-definable fields per record.
The sorting arm shall be flexible.	<ul style="list-style-type: none">The sorting arm shall elastically deform under loads of 0-75 pounds.
PQR shall clearly display safety warnings.	<ul style="list-style-type: none">PQR shall display safety warnings in yellow letters [1" +/- 0.05" high] and [0.5" +/- 0.05" wide].
The power supply shall be portable.	<ul style="list-style-type: none">The power supply shall weigh 25 pounds or less.The power supply shall be less than or equal to 20 inches in each dimension.The power supply shall have a carrying handle of the dimensions in drawing 12 of reference 4 (Human Factors Standards).
The case shall accommodate contingency maintenance tools.	<ul style="list-style-type: none">The case shall have maintenance tool storage to hold all tools in drawing A.
The TMS shall handle deposits quickly.	<ul style="list-style-type: none">The TMS shall scan and record customer account number and amount from a single deposit slip in 2 seconds or less.
XYZ shall be user-friendly.	<ul style="list-style-type: none">XYZ shall have controls labeled with their purpose in letters 0.3" +/- 0.03".XYZ shall have controls positioned in the order (from left to right) of their use.XYZ shall display menus of control options.XYZ shall display prompts to remind the user of the next step.XYZ shall use the display convention of product PQR.XYZ shall have emergency stop controls colored red.
MNOP shall be safe.	<ul style="list-style-type: none">MNOP shall stop operation if a person comes within 10 feet of any moving component.MNOP shall stop heaters if the vat temperature exceeds 100 degrees Celsius.MNOP shall meet UL 544 Section 3.4 standards for temperatures on external surfaces.

6.4 VER



When eliciting verifiability requirements, consider the following aspects:

- ◆ **VERIFICATION AND VALIDATION (V & V)** techniques that might be used during development and/or implementation of the software system (refer to Table 6-2).
- ◆ **POSSIBLE INSPECTION CHECKS BY FAULT CLASS** [Sommerville, 2007].
 - ◊ **Data faults.**
 - Are all program variables initialized before their values are used?
 - Have all constants been named?
 - Should the upper bound of arrays be equal to the size of the array or size -1?
 - If character strings are used, is a delimiter explicitly assigned?
 - Is there any possibility of buffer overflow?
 - ◊ **Control faults.**
 - For each conditional statement, is the condition correct?
 - Is each loop certain to terminate?
 - Are compound statements correctly bracketed?
 - In case statements, are all possible cases accounted for?
 - If a break is required after each case in case statements, has it been included?
 - ◊ **Interface faults.**
 - Do all function and method calls have the correct number of parameters?
 - Do formal and actual parameter types match?
 - Are the parameters in the right order?
 - If components access shared memory, do they have the same model of the shared memory structure?

6.4 VER



- ◊ **Storage management faults.**
 - If a linked structure is modified, have all links been correctly reassigned?
 - If dynamic storage is used, has space been allocated correctly?
 - Is space explicitly de-allocated after it is no longer required?
- ◊ **Exception management faults.**
 - Have all possible error conditions been taken into account?
- ◊ **Input/Output faults.**
 - Are all input variables used?
 - Are all output variables assigned a value before they are output?
 - Can unexpected inputs cause corruption?
- ♦ **INSTALLABILITY OF THE SYSTEM COMPONENT.**
 - ◊ **Installation instructions.** What documentation is needed to tell the installer what to do? Who is responsible for documenting these instructions?
 - ◊ **Authorization to install.** Who has authority to migrate the software into production? What special elements are needed to appropriate access during installation?
 - ◊ **Upgrading.** This is specifically concerned with installing an upgrade when a previous version of the software is already in production.
 - ◊ **Troubleshooting.** Provide help resources to the installer if problems arise. The resources should help the installer identify the problem and fix it in at least 80% of the instances encountered.
 - ◊ **Security.** What might need to be done to minimize the risk of security attacks or breaches? For example, might the installer need to have access restrictions related to sensitive data?
 - ◊ **Training.** What special training must be provided to the installation personnel?



- ◊ **Uninstallation procedures and instructions.** The degree to which software is uninstalled depends largely on what the software is used for. Is it necessary to remove all versions? Does converted data need to be purged?

6.4.3 Verifiability Requirement Examples

- a) No member of a test panel of 500 children (aged 8) shall incur an injury while playing with the product. The product must comply with product safety regulations as defined by (specify).
- b) The maximum number of test cases to cover testing of any particular source code module shall be 20.
- c) The Customer Information System shall be implemented using Release 5 of Library 4B.
- d) The system and supporting infrastructure for the automatic shut-down sequence must be validated to the highest reasonable commercial reliability standards.
- e) The design of the Payroll System shall include software that tests the operating system and the communication links, memory devices, and peripheral devices.
- f) It shall be possible for the All-in-One Printer software to be installed by a typical customer who has no special expertise. The installation process shall be convenient and involve the entry of little information by the user.
- g) When a new version of the payroll system is released, it shall be possible to upgrade to it from any previous version.
- h) Software testing will require the use of a test database with data extracted from the production database. This test database will be deleted after successful implementation of the software system.
- i) At the time of conversion the customer demographic information shall have a 99.5% match rate after data scrubbing.
- j) All developers on the project shall have identical development environment configurations, and all testers shall have identical quality assurance environment configurations.

6.4 VER



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

6.4.4 Verifiability Suggested Questions

DATA (WHAT?)

ID	Verifiability Suggested Questions	Ask this to:
VER1S	What defined validation exists for each data element?	<i>clarify validations.</i>
VER2S	How are the calculations validated?	<i>clarify validations.</i>
VER3S	What test data must be recorded and how fast must the system output this data?	<i>identify activities.</i>
VER4S	What test beds of data exist?	<i>identify available resources.</i>
VER5S	What data elements are unique?	<i>identify data attributes.</i>
VER6S	What bounds, limits, and ranges of data elements are known?	<i>identify data attributes.</i>
VER7S	What data elements have a one-time use?	<i>identify data attributes.</i>
VER8S	What data elements can be reused?	<i>identify data characteristics.</i>
VER9S	What data or special reports are required by the regulatory agencies?	<i>identify sources of compliance.</i>
VER10R	What test data are needed to simulate all user classes?	<i>identify access security needs.</i>
VER11R	What classifications of defects are suitable for this system?	<i>identify classifications.</i>
VER12R	What simulated data streams will be needed to test the system?	<i>identify combinations of actions.</i>
VER13R	What is the source of the calculations used in the system?	<i>identify definitive sources for validation.</i>
VER14R	How are the factors used in the calculation traced to their origins?	<i>identify factors.</i>
VER15R	What purchased hardware/software can be obtained to start the population of information?	<i>identify needed resources.</i>
VER16R	What licenses are required with the purchase of hardware/software?	<i>identify needed resources.</i>
VER17R	What defects are documented from the design and development team?	<i>identify resources.</i>
VER18R	What information does each team member need to build and test the system?	<i>identify test information requirements.</i>
VER19R	What defined data patterns, formats, and presentation styles are there?	<i>identify types of data.</i>
VER20R	How valid is the test bed of data?	<i>identify verification data.</i>
VER21B	What else should I be asking about verifiability “data”?	<i>uncover additional requirements.</i>

6.4 VER



ROLES (WHO?)

ID	Verifiability Suggested Questions	Ask this to:
VER22s	Who are the best testers?	<i>identify additional stakeholders.</i>
VER23s	Who loves to beat the system?	<i>identify additional stakeholders.</i>
VER24s	Who can supply validation models?	<i>identify additional stakeholders.</i>
VER25s	Who will validate help information?	<i>identify additional stakeholders.</i>
VER26s	What customer involvement will be needed to test the system?	<i>identify additional stakeholders.</i>
VER27s	What user involvement will be needed to test the system?	<i>identify additional stakeholders.</i>
VER28s	Who would like to see the project/system fail?	<i>identify role perspectives.</i>
VER29s	Who will do user acceptance testing?	<i>identify role responsibilities.</i>
VER30s	What agency has jurisdiction over this product line?	<i>identify role responsibilities.</i>
VER31s	Who will be signing off the system for promotion into production?	<i>identify role responsibilities.</i>
VER32s	Who must perform the test, perform the analysis or demonstration, or inspect the system?	<i>identify role responsibilities.</i>
VER33s	What extra reviews from outside experts or inspectors and liability insurance will be required before the product meets standards?	<i>identify sources of validation.</i>
VER34s	What liability concerns does the legal department have?	<i>identify sources of validation.</i>
VER35s	What liability concerns does the insurance provider have?	<i>identify sources of validation.</i>
VER36s	What customer acceptance testing will be necessary?	<i>identify sources of validation.</i>
VER37R	What is the known skill level of the development team and their ability to unit test?	<i>assess lessons learned.</i>
VER38R	Who gets what hardware?	<i>determine verification resources.</i>
VER39R	Who gets what software?	<i>determine verification resources.</i>
VER40R	What access is required at start-up?	<i>identify access security needs.</i>
VER41R	Who will validate that necessary training occurred?	<i>identify additional stakeholders.</i>

6.4 VER



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

ROLES (WHO?) (continued)

ID	Verifiability Suggested Questions	Ask this to:
VER42R	Whom will users contact for system support?	<i>identify additional stakeholders.</i>
VER43R	Who will validate that the system works at an acceptable level to be implemented?	<i>identify additional stakeholders.</i>
VER44R	Who will create a testing traceability model?	<i>identify role responsibilities.</i>
VER45R	Who is responsible for the Quality Assurance (QA) test plan?	<i>identify role responsibilities.</i>
VER46B	What stakeholders and team members need special access during development?	<i>identify access security needs.</i>
VER47B	What stakeholders and team members need special access during testing?	<i>identify access security needs.</i>
VER48B	What stakeholders and team members need special access to promote to production?	<i>identify access security needs.</i>
VER49B	Who should participate in reviews?	<i>identify additional stakeholders.</i>
VER50B	What else should I be asking about verifiability “roles”?	<i>uncover additional requirements.</i>

6.4 VER



PURPOSE (WHY?)

ID	Verifiability Suggested Questions	Ask this to:
VER51s	What are currently common points of failure with the systems and the users?	<i>assess conditions.</i>
VER52s	How important is "cost/time to repair" as an evaluation criteria for going to production with known defects?	<i>identify criterion.</i>
VER53s	How important is probability as an evaluation criteria for going to production with known defects?	<i>identify criterion.</i>
VER54s	How important is the degree of impact as an evaluation criteria for going to production with known defects?	<i>identify criterion.</i>
VER55s	How important is risk as an evaluation criteria for going to production with known defects?	<i>identify criterion.</i>
VER56s	With regard to requirements quality criteria, how are the requirements going to be validated?	<i>identify criterion.</i>
VER57s	With regard to requirements quality criteria, how are the requirements going to be verified?	<i>identify criterion.</i>
VER58s	With regard to requirements quality criteria, how are the design components going to be tested?	<i>identify criterion.</i>
VER59s	Where are the particularly complex elements in this system?	<i>identify criterion.</i>
VER60s	What is the acceptance criterion for decision regarding promotion to production?	<i>identify criterion.</i>
VER61s	What requirements are there for the origination of every factor?	<i>identify definitive sources.</i>
VER62s	What requirements will have to be developed in parallel for testing?	<i>identify dependencies.</i>
VER63s	What industry standards and any other standards mandated by regulatory agencies apply?	<i>identify governing/regulatory controls.</i>
VER64s	What levels of defect are acceptable?	<i>identify metrics.</i>
VER65s	What level of confidence in the system is required to promote into production status?	<i>identify scales of measure.</i>
VER66s	What verification requirements will the government, regulatory agencies, and/or insurance carriers place on the product?	<i>identify sources of validation.</i>

6.4 VER



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?) (continued)

ID	Verifiability Suggested Questions	Ask this to:
VER67s	What industry product standards apply to product certification?	<i>identify sources of validation.</i>
VER68s	What tests must be done to satisfy marketing claims?	<i>identify sources of validation.</i>
VER69s	What tests will validate that the product meets contractual agreements?	<i>identify sources of validation.</i>
VER70s	What requirements may be difficult to verify and increase risk?	<i>identify types of events.</i>
VER71s	What additional verification costs must be considered?	<i>identify verifications.</i>
VER72R	What is the history of defects from the quality assurance (QA) team?	<i>assess lessons learned.</i>
VER73R	What is the history of defects from the development team?	<i>assess lessons learned.</i>
VER74R	What business policies apply to the developers and testers of the system?	<i>clarify business rules with internal and external development staff.</i>
VER75R	What specific design constraints on size and complexity of components should be considered to facilitate testing?	<i>identify specific constraints.</i>
VER76B	What is the cost of testing versus the cost of defect?	<i>identify possible trade-offs.</i>
VER77B	What else should I be asking about verifiability "purpose"?	<i>uncover additional requirements.</i>

6.4 VER



CHAPTER 6: REVISION REQUIREMENTS

TIMING (WHEN?)

ID	Verifiability Suggested Questions	Ask this to:
VER78s	What customers/users need to be available during design?	<i>identify needed resources.</i>
VER79s	What customers/users need to be available during coding and development?	<i>identify needed resources.</i>
VER80s	What customers/users need to be available for testing in the quality assurance (QA) environment?	<i>identify needed resources.</i>
VER81s	What customers/users need to be available during promotion to production?	<i>identify needed resources.</i>
VER82s	What customers/users need to be available during system roll-out?	<i>identify needed resources.</i>
VER83s	What additional time is required to verify the system?	<i>identify time constraints.</i>
VER84R	What equipment needs to be purchased for roll-out?	<i>identify needed resources.</i>
VER85R	What data will need to be populated at start-up?	<i>identify roll-out information requirements.</i>
VER86R	How will user manuals be available and distributed?	<i>identify roll-out information requirements.</i>
VER87R	When is equipment for roll-out needed?	<i>identify timing of resource delivery.</i>
VER88R	When will development begin?	<i>identify timing of resource delivery.</i>
VER89R	What functions of the system must be verified during design?	<i>identify types and timing of verification.</i>
VER90R	What functions of the system must be verified during coding and development?	<i>identify types and timing of verification.</i>
VER91R	What functions of the system must be verified prior to production implementation?	<i>identify types and timing of verification.</i>
VER92R	What functions of the system must be verified after roll-out to production?	<i>identify types and timing of verification.</i>
VER93R	What types of combinations of activity are expected to happen concurrently?	<i>identify types of events.</i>
VER94R	What various times must the system be tested?	<i>identify types of events.</i>
VER95R	When will system testing occur?	<i>identify types of events.</i>
VER96B	What else should I be asking about verifiability “timing”?	<i>uncover additional requirements.</i>

6.4 VER



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

LOGISTICS (WHERE?)

ID	Verifiability Suggested Questions	Ask this to:
VER97s	What extreme environment simulations should be considered?	<i>identify environment conditions.</i>
VER98s	What regulatory agencies and standards apply to each business location?	<i>identify localized controls.</i>
VER99s	Where will system development be done?	<i>identify locations.</i>
VER100s	Where will quality assurance (QA) testing be done?	<i>identify locations.</i>
VER101s	In which business locations will the system be implemented?	<i>identify locations.</i>
VER102R	What different power, cooling or heating, software or hardware support will the system require during development than it will require in operation?	<i>identify environmental conditions.</i>
VER103R	Where is equipment for roll-out needed?	<i>identify location of resource delivery.</i>
VER104R	What prototypes must be developed?	<i>identify prototypes.</i>
VER105R	What various places must be included during system testing?	<i>identify types of events.</i>
VER106R	Where will system testing occur?	<i>identify types of events.</i>
VER107R	What features must be testable in combination?	<i>identify types of feature relationships.</i>
VER108R	What features must be testable in isolation from the rest of the system?	<i>identify types of features.</i>
VER109R	What calculations are used in multiple locations?	<i>identify usage.</i>
VER110B	What special facilities or equipment will be needed to test the system?	<i>identify resources.</i>
VER111B	What additional testing resources (software, hardware, and/or people) are needed?	<i>identify resources.</i>
VER112B	What else should I be asking about verifiability "logistics"?	<i>uncover additional requirements.</i>

6.4 VER



PROCESS (How?)

ID	Verifiability Suggested Questions	Ask this to:
VER113s	What test cases will be written at the requirements level?	<i>identify approaches.</i>
VER114s	What features of the system can expect the most frequent changes or enhancements?	<i>identify features.</i>
VER115s	What proof of quality, beyond functional requirements testing, is required to make promotion decisions?	<i>identify sources of validation.</i>
VER116s	What conditions outside the normal usage must be tested?	<i>identify types of conditions.</i>
VER117s	What are the most critical functions to test?	<i>prioritize functions.</i>
VER118R	In what processes have similar or predecessor systems run into error conditions?	<i>assess lessons learned.</i>
VER119R	Due to complexity or criticality, what elements of the system need to be isolated?	<i>classify features.</i>
VER120R	What are the highest risk areas to test?	<i>identify exposure.</i>
VER121R	How will the system be rolled out?	<i>identify implementation processes.</i>
VER122R	What internal and external interfaces must be tested in addition to the system?	<i>identify integration.</i>
VER123R	How will users be trained?	<i>identify needed resources.</i>
VER124R	What purchased hardware/software can be obtained to perform specific events?	<i>identify needed resources.</i>
VER125R	What test support equipment or software will be needed?	<i>identify resources.</i>
VER126R	What need is there for a defect tracking tool?	<i>identify resources.</i>
VER127R	What test scripts have been used in the past?	<i>identify sources of reference.</i>
VER128R	How will defects/faults be classified?	<i>identify tactics.</i>
VER129R	What are the classification levels of defects/faults?	<i>identify tactics.</i>
VER130R	What simulators need to be developed for testing?	<i>identify testing processes.</i>
VER131R	What special testing tools will the Quality Assurance (QA) team need?	<i>identify tools.</i>
VER132R	How will the requirements for this system be verified?	<i>identify verifications.</i>
VER133R	What could be used from the development and testing environments for verifying the production system?	<i>leverage efforts.</i>
VER134B	What else should I be asking about verifiability “processes”?	<i>uncover additional requirements.</i>

6.4 VER



6.5 SUGGESTED READING

Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage, Tom Gilb, [Gilb, 2005]. Chapter 5 presents definitions for quality attributes, including **flexibility**, which decomposes into connectability and tailorability.

Customer-Centered Products, by Ivy Hooks and Kristin Farry, [Hooks, 2001]. Chapter 10 presents an explanation of **verification**.

Software Engineering, 8th Edition, by Ian Sommerville, [Sommerville, 2007]. Part 5 focuses on techniques for **software verification and validation**.

Software Requirement Patterns, by Stephen Withall, [Withall, 2007]. Chapter 10 of this book presents **flexibility** as its own domain category consisting of six requirement patterns: un-parochialness, multiness, multi-lingual, **scalability**, **extendability**, and installability.

Software Requirements: Styles and Techniques, by Soren Lauesen, [Lauesen, 2002]. Section 6.10 discusses **maintenance** activities such as repairing defects, extending the product, informing and training users, while Section 6.11 presents **maintainability** requirement examples.

System Requirements Analysis, Jeffrey O. Grady, [Grady, 2006]. Although Grady's book focuses on system requirements instead of business requirements, Section 3.7, "Specialty Engineering Requirements Analysis," includes a discussion on reliability and **maintainability** engineering. Grady defines the activities that make up the **Requirements Verification Management** process in Section 6.6.





7

TRANSITION REQUIREMENTS

IN THIS CHAPTER:

7.1 Interoperability (IOP)	287
7.2 Portability (POR)	298
7.3 Reusability (REU)	307
7.4 Suggested Reading.....	316

HOW EASY IS IT TO ADAPT TO CHANGES IN THE TECHNICAL ENVIRONMENT?

Transition requirements describe the ability of the software system to adapt to its surrounding environment. Users who come in contact with the software system by managing the upkeep of the system are generally most concerned with transition requirements. That is, users view *quality* as a system that:

- ◆ Can be moved efficiently to other operating environments.
- ◆ Connects easily with other systems.
- ◆ Lends itself to reuse.



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

The transition group of nonfunctional categories encompasses the concerns of users that are managing a production system. Transition requirements answer user concerns for the following software qualities:

- ◆ **INTEROPERABILITY.** How easy is it to interface with another system?
- ◆ **PORTABILITY.** How easy is it to transport?
- ◆ **REUSABILITY.** How easy is it to convert for use in another system?

In the same sequence as listed above, this chapter presents the following for each transition category: definition, brief discussion, examples of requirements, and suggested elicitation questions. The introduction to Part Two explains the anatomy of a category and provides guidance for how to use the suggested questions.



7.1 INTEROPERABILITY (IOP)

USER CONCERN: How easy is it to interface with another system?

RELATED CATEGORIES: Compatibility, Connectivity, Operational

7.1.1 Interoperability Definition

Interoperability is the extent to which the software system is able to couple or facilitate the interface with other systems.

7.1.2 Interoperability Discussion

In a broad definition, interoperability is a property referring to the ability of diverse systems and organizations to work together (inter-operate). In software engineering terminology, “interoperability” refers to the capability of a system to work with other systems without special effort from the user. It involves the capability of various technical units to transfer data in a manner such that the user needs little or no knowledge of the specific workings of the technical units.

Interoperability is the ability for multiple software components to interact regardless of their implementation programming language or hardware platform. Interoperability is a term used to describe the ability of different source programs to exchange data via common exchange formats, to read and write the same file formats (a particular way to encode information for storage by a computer—for example, JPEG, GIF), and to use the same protocols (a standard that controls or enables the connection, communication, and data transfer between two computing endpoints, for example, TCP/IP).

Interoperability requirements describe the ease with which the system collaborates with partner applications and external operations. Interoperability requirements also identify the ability to add or remove interfaces without disrupting the core system.

7.1 IOP



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

When eliciting interoperability requirements, consider the following aspects:

- ◆ **SOFTWARE TESTING.** Software produced to a common standard depends on clarity of the standards. However, there may be discrepancies in their implementation that system or unit testing may not uncover. This requires that systems formally be tested in a production scenario to ensure they actually will inter-operate as expected. Interoperable system testing is different from conformance-based testing as conformance to a standard does not necessarily cause interoperability with another system, which is also tested for conformance.
- ◆ **PRODUCT ENGINEERING.** This means implementing a common standard as defined by the industry with the specific intention of achieving interoperability with other software implementations that also follow the same standard.
- ◆ **INDUSTRY PARTNERSHIP.** Industry partnerships, either domestic or international, sponsor standard workgroups with the purpose to define a common standard that may be used to allow software systems to intercommunicate for a defined purpose. At times an industry will sub-profile an existing standard produced by another organization to reduce options and thus making interoperability more achievable for implementations.
- ◆ **COMMON TECHNOLOGY.** The use of a common technology, such as Internet Protocol (IP), may speed up and reduce complexity of interoperability by reducing variability between components from different sets of separately developed software products, and thus allowing them to intercommunicate more readily.
- ◆ **STANDARD IMPLEMENTATION.** Software interoperability requires a common agreement that is normally arrived at via an industrial, national, or international standard.



7.1.3 Interoperability Requirement Examples

- a) The system must be able to interface with any HTML (HyperText Markup Language) browser.
- b) The baselined version 2 of the spreadsheet must be able to access information from the previous baselined version 1.
- c) The Automated Teller Machine (ATM) must interface with the Automated Clearing House (ACH) in order to complete withdrawal transactions.
- d) The product shall not use picture icons that could be considered offensive in any country where the product is marketed.
- e) The common language used in the incoming mail department shall be English to increase communication effectiveness and reduce processing errors.
- f) Any change or upgrade to the interface between the Order Entry System and the Inventory Warehouse system shall be installed simultaneously by both systems.
- g) Local communications for each SmartMeter system can be used within the premises to link to local devices through utilization of standard protocols. There will be no reliance or operational mandate upon a Supplier to organize a site visit in order to install and/or maintain operation of any local device out of the SmartMeter system.
- h) Physical communications will be guaranteed as interoperable through the use of an agreed minimum national service standard (to be defined) for prioritized exchanges of data. Any such service standards would be expressed in the time taken to complete the data exchange, for example, 15 minutes, 2 hours, 1 day. It will not be prescriptive about the number of retries required or delivery performance—all exchanges of data will complete.
- i) All SmartMeter systems will provide a standard interface that can be used by meter operators for installation and maintenance purposes without disturbing any meter seals and reinstating any tamper detection covers.
- j) To be interoperable with the SmartMeter energy system, any local device must support the chosen local communications, data interface, protocol and security solutions.

7.1 IOP



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

7.1.4 Interoperability Suggested Questions

DATA (WHAT?)

ID	Interoperability Suggested Questions	Ask this to:
IOP1S	Why are the data important to the business?	<i>clarify business needs.</i>
IOP2S	What data are expected to be customizable by the customer?	<i>clarify business needs.</i>
IOP3S	Describe the purpose of the data used.	<i>clarify business needs.</i>
IOP4S	How would other departments or people describe the data?	<i>clarify business needs.</i>
IOP5S	What information is missing from reports currently produced?	<i>identify additional data needed.</i>
IOP6S	What need is there to record what was sent and received on the interface?	<i>identify data types.</i>
IOP7S	Where can documentation for industry standards for this interface be found?	<i>identify external factors.</i>
IOP8S	What kinds of data exchange are envisioned (query, update, create)?	<i>identify usage types.</i>
IOP9R	What need is there to record acknowledgment of receipt from the other system?	<i>identify communication types.</i>
IOP10R	What documentation exists for the interface?	<i>identify documentation.</i>
IOP11R	What need is there to detect missing or duplicated traffic?	<i>identify monitoring techniques.</i>
IOP12B	What information must be exchanged with other systems?	<i>identify information sources and targets.</i>
IOP13B	What else should I be asking about interoperability “data”?	<i>uncover additional requirements.</i>

7.1 IOP



ROLES (WHO?)

ID	Interoperability Suggested Questions	Ask this to:
IOP14s	Who should not be using this system?	<i>clarify users.</i>
IOP15s	How does one user role type differ from another?	<i>clarify users.</i>
IOP16s	Does each person, department, or system that interfaces with the system under development receive and/or provide information?	<i>clarify why the system is used.</i>
IOP17s	How many of each role type are expected to interact with the system in the first month of operation? First year?	<i>estimate size of user community.</i>
IOP18s	What are the standards and abilities of the external supplier?	<i>evaluate suppliers.</i>
IOP19s	How does this external supplier rate on the vendor scorecard?	<i>evaluate vendors.</i>
IOP20s	Who should be involved in supplying requirements regarding business policies, rules, and procedures?	<i>identify additional stakeholders.</i>
IOP21s	How will customers find the information they want?	<i>identify customer needs.</i>
IOP22s	What customers expect the latest device to be available?	<i>identify customer types.</i>
IOP23s	What customers are early adapters of new technology?	<i>identify customer types.</i>
IOP24s	What are the roles and responsibilities of each "who" identified?	<i>identify responsibilities.</i>
IOP25s	What role will the business partners play in the development of this system?	<i>identify responsibilities.</i>
IOP26s	What external suppliers use the interface?	<i>identify types of interfaces.</i>
IOP27s	What kind of customers will want to use this system?	<i>identify users.</i>
IOP28s	What internal departments will interface with the system under development or installation?	<i>identify users.</i>
IOP29s	What external departments will interface with the system under development or installation?	<i>identify users.</i>
IOP30s	What new roles will be needed?	<i>identify users.</i>
IOP31s	How important is each "who" that has been identified?	<i>prioritize responsibilities.</i>
IOP32R	Who owns the interface?	<i>identify accountability.</i>
IOP33R	What other interfaces exist with this external supplier?	<i>identify relationships.</i>

7.1 IOP



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

ROLES (WHO?) (continued)

ID	Interoperability Suggested Questions	Ask this to:
IOP34R	Who has expertise in the technology?	<i>identify resources.</i>
IOP35R	Who will build the interface adapter?	<i>identify responsibilities.</i>
IOP36R	Who must be alerted when the interface fails?	<i>identify responsibilities.</i>
IOP37R	Who is responsible for the interface?	<i>identify responsibilities.</i>
IOP38R	Who maintains the interface documentation?	<i>identify responsibilities.</i>
IOP39R	Who (role) coordinates the interfaces?	<i>identify responsibilities.</i>
IOP40R	Who (role) approves the addition/removal of interfaces?	<i>identify responsibilities.</i>
IOP41R	Who will be responsible for the integration of the interfaces?	<i>identify responsibilities.</i>
IOP42R	Who has access to the required technology?	<i>identify sources of knowledge.</i>
IOP43B	What else should I be asking about interoperability "roles"?	<i>uncover additional requirements.</i>

PURPOSE (WHY?)

ID	Interoperability Suggested Questions	Ask this to:
IOP44S	Why is the system needed?	<i>clarify business purpose.</i>
IOP45S	What is the impact of not having the interface?	<i>evaluate usage strategies.</i>
IOP46S	What agreements (contracts, obligations) have been put in place?	<i>identify agreements.</i>
IOP47S	What agreements (contracts, obligations) must be secured?	<i>identify agreements.</i>
IOP48S	What state, national, and international laws affect the processes performed by the system under development?	<i>identify business rules.</i>
IOP49S	What corporate policies will affect the system?	<i>identify business rules.</i>
IOP50S	What additional limits exist that should be imposed on customers?	<i>identify business rules.</i>
IOP51S	What additional limits exist that should be imposed on business partners?	<i>identify business rules.</i>

7.1 IOP



PURPOSE (WHY?) (continued)

ID	Interoperability Suggested Questions	Ask this to:
IOP52s	What policies must change to meet new business objectives?	<i>identify business rules.</i>
IOP53s	How quickly do new devices reach the market?	<i>identify market rates.</i>
IOP54s	What other applications would interface for exchange of data?	<i>identify partner applications.</i>
IOP55s	What corporate policies of business partners conflict with the system?	<i>identify policy risks.</i>
IOP56s	What is the effect on business if an imposed policy was not followed?	<i>identify policy risks.</i>
IOP57s	What legal policies must be enforced to protect the company?	<i>identify policy risks.</i>
IOP58s	For each policy, what would be the impact on the business if the rule is not enforced?	<i>identify policy risks.</i>
IOP59s	What data are important to the business?	<i>identify scope of the effort.</i>
IOP60s	What data are important to the business partners?	<i>identify scope of the effort.</i>
IOP61s	What information is the customer interested in?	<i>identify scope of the effort.</i>
IOP62s	What types of future interfaces are anticipated?	<i>identify types of changes.</i>
IOP63s	What is the purpose for the interface?	<i>identify usages.</i>
IOP64s	What protections are needed from the external vendor? (For example, warranty, bonding)	<i>identify vendor agreements.</i>
IOP65s	What would be the effect on the business if this event were not handled by the system?	<i>prioritize events.</i>
IOP66s	How important is each data group to the success of this system?	<i>prioritize requirements.</i>
IOP67s	How important is each policy to the success of the system?	<i>prioritize requirements.</i>
IOP68s	What rules must be enforced to maintain profitability?	<i>prioritize requirements.</i>
IOP69s	How important is each event to the system under development?	<i>prioritize requirements.</i>
IOP70s	How important is each location to the system?	<i>prioritize requirements.</i>
IOP71s	How are conflicts with the standards reconciled?	<i>prioritize standards.</i>
IOP72R	What are restrictions on the format or medium that must be used for input or output?	<i>identify formats and media.</i>

7.1 IOP



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?) (continued)

ID	Interoperability Suggested Questions	Ask this to:
IOP73R	What specifications are needed for the interface adapter?	<i>identify implementation constraints.</i>
IOP74R	What controls are needed over change?	<i>identify limitations.</i>
IOP75R	What interfaces require special access control?	<i>identify limitations.</i>
IOP76R	What specific technology must be used?	<i>identify technical constraints.</i>
IOP77R	What specific technology should not be used?	<i>identify technical constraints.</i>
IOP78B	What industry standard protocols must be applied to this application?	<i>identify constraints.</i>
IOP79B	What version(s) of industry standards must be followed?	<i>identify constraints.</i>
IOP80B	What need is there to switch the interface on and off?	<i>identify controls.</i>
IOP81B	What additional usages exist for this interface?	<i>identify options.</i>
IOP82B	What is the trade-off with efficiency, reliability, and integrity?	<i>identify possible trade-offs.</i>
IOP83B	If there are multiple interface alternatives, which alternative is the preferred priority for use?	<i>identify priorities.</i>
IOP84B	What standards for interfaces does the company require?	<i>identify standards.</i>
IOP85B	What industry standards exist for the interface?	<i>identify standards.</i>
IOP86B	What is a widely-accepted standard for this interface?	<i>identify standards.</i>
IOP87B	Describe the purpose of the interface.	<i>identify strategies.</i>
IOP88B	What inputs are coming from systems outside the proposed system?	<i>identify various inputs.</i>
IOP89B	What outputs are sent to systems outside the proposed system?	<i>identify various outputs.</i>
IOP90B	What else should I be asking about interoperability “purpose”?	<i>uncover additional requirements.</i>

7.1 IOP



TIMING (WHEN?)

ID	Interoperability Suggested Questions	Ask this to:
IOP91s	When are periods of time that information will be accessed more than others?	<i>clarify usage.</i>
IOP92s	During what peak times will the system be used for any of the role types?	<i>clarify usage.</i>
IOP93s	What business conditions would trigger an action by the system?	<i>identify events.</i>
IOP94s	What happens at the start of a normal work day?	<i>identify time-triggered events.</i>
IOP95s	What happens at the end of a normal work day?	<i>identify time-triggered events.</i>
IOP96s	What events occur weekly?	<i>identify time-triggered events.</i>
IOP97s	What events occur monthly?	<i>identify time-triggered events.</i>
IOP98s	What events occur quarterly?	<i>identify time-triggered events.</i>
IOP99s	What events occur annually?	<i>identify time-triggered events.</i>
IOP100s	What other standard periods drive activities?	<i>identify time-triggered events.</i>
IOP101s	When are the peak times that each event occurs, if any?	<i>identify time-triggered events.</i>
IOP102s	What business events are cyclical?	<i>identify time-triggered events.</i>
IOP103s	How much new information is expected the first month through the first year that the new system is operating?	<i>identify volume metrics.</i>
IOP104s	What is the minimum number of occurrences of each event?	<i>identify volume metrics.</i>
IOP105s	What is the maximum number of occurrences of each event?	<i>identify volume metrics.</i>
IOP106s	What is the average number of occurrences of each event?	<i>identify volume metrics.</i>
IOP107R	How often is the interface upgraded?	<i>identify change controls.</i>
IOP108R	What timing expectations exist when using the interface?	<i>identify implementation constraints.</i>
IOP109R	What timing constraints must be considered while using interfaces?	<i>identify implementation constraints.</i>
IOP110R	How well and how quickly must the interface recover from a failure?	<i>identify implementation tactics.</i>
IOP111R	When is the interface upgraded?	<i>identify timing.</i>
IOP112R	When is the information exchanged?	<i>identify timings.</i>
IOP113B	What else should I be asking about interoperability “timing”?	<i>uncover additional requirements.</i>

7.1 IOP



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

LOGISTICS (WHERE?)

ID	Interoperability Suggested Questions	Ask this to:
IOP114s	What volume of information is currently provided from each location?	<i>clarify usage.</i>
IOP115s	What volume of information is currently received from each location?	<i>clarify usage.</i>
IOP116s	Where do the business partners operate?	<i>identify business locations of business partners.</i>
IOP117s	Where are customers located?	<i>identify business locations of clients.</i>
IOP118s	Where does the business operate?	<i>identify business locations.</i>
IOP119s	What are the plans for additional locations?	<i>identify business locations.</i>
IOP120R	Where will software be changed or added?	<i>identify location resources.</i>
IOP121R	Where will hardware be changed or added?	<i>identify location resources.</i>
IOP122R	Where will the system be developed?	<i>identify location resources.</i>
IOP123B	What else should I be asking about interoperability “logistics”?	<i>uncover additional requirements.</i>

PROCESS (How?)

ID	Interoperability Suggested Questions	Ask this to:
IOP124s	How quickly must the application adopt the new devices?	<i>identify adoption rates.</i>
IOP125s	What events must the business support?	<i>identify essential processing.</i>
IOP126s	What internal systems will interface with the system under development or installation?	<i>identify interfaces.</i>
IOP127s	What external systems will interface with the system under development or installation?	<i>identify interfaces.</i>
IOP128s	What are other systems this specific system must connect with?	<i>identify interfaces.</i>
IOP129s	What forms of access will the customers use?	<i>identify technology issues.</i>
IOP130s	What forms of access will the business partners require?	<i>identify technology issues.</i>



PROCESS (HOW?) (continued)

ID	Interoperability Suggested Questions	Ask this to:
IOP131R	What are alternatives when the interface fails?	<i>identify business continuation tactics.</i>
IOP132R	What happens when the interface changes?	<i>identify configuration controls.</i>
IOP133R	What configuration parameters are required?	<i>identify configurations.</i>
IOP134R	What commercial software will be part of the interface?	<i>identify configurations.</i>
IOP135R	What ways are necessary to support users adding their own devices?	<i>identify deployment strategies.</i>
IOP136R	What problems can we anticipate with upgrading users to new technology?	<i>identify deployment strategies.</i>
IOP137R	When adding new devices will the communications protocol be standard?	<i>identify devices.</i>
IOP138R	What alternate routes (multiple interfaces) exist between the systems?	<i>identify implementation options.</i>
IOP139R	What influence on interface design can we exercise?	<i>identify implementation strategies.</i>
IOP140R	How will the interface be stress-tested?	<i>identify implementation verification tactics.</i>
IOP141R	What load level has been tested on the interface?	<i>identify implementations.</i>
IOP142R	How many generations of interfaces must be supported?	<i>identify implementations.</i>
IOP143R	How is the interface upgraded?	<i>identify implementations.</i>
IOP144R	How many versions of the communications protocol are in use?	<i>identify installations.</i>
IOP145R	How widely used is the interface?	<i>identify installations.</i>
IOP146B	What options are available for the interface?	<i>identify variations.</i>
IOP147B	Where is the interface incomplete for our intended use?	<i>identify variations/deviations.</i>
IOP148B	What are possible alternatives for the interface?	<i>identify various types of interfaces.</i>
IOP149B	What else should I be asking about interoperability "processes"?	<i>uncover additional requirements.</i>

7.1 IOP



7.2 PORTABILITY (POR)

USER CONCERN: How easy is it to transport the system?

RELATED CATEGORIES: Compatibility, Connectivity, Migratability, Transferability

7.2.1 Portability Definition

Portability is the ease with which a software system can be transferred from its current hardware or software environment to another environment.

7.2.2 Portability Discussion

Portability requirements describe the ability to migrate software from one operating system (OS) to another. The most common operating system platform is the Microsoft Windows® family of products such as Vista®. True multi-user, multi-task software can be easily ported to other systems if it is designed properly. The software should run well on a multitude of platforms, including Windows®, Linux®, Unix®, and even IBM mainframes. There would not be different versions of the software written for each operating system platform, but rather one set of source and object code and one set of data files. The same software program that runs on a Windows XP® machine should function on the latest Macintosh®, with no need for the user to re-learn the software.

As technology advances and software environments get more complicated, one must take into account all associated components that combine to make the whole system. Identifying all the right components is critical to portability. Components to consider include the following:

7.2 POR

- ◆ SIZE OF WHAT IS BEING TRANSPORTED.
- ◆ ORIGINAL AND THE TARGET ENVIRONMENTS (compatibility).



- ◆ **MEANS OF TRANSPORT** (such as a manual or automated conversion).
- ◆ **RESOURCES NEEDED TO PERFORM THE TRANSPORT**, and the attributes of the new system compared to the old (side-effects).

Developing a system that is portable requires that you first identify everywhere that the system might be installed. For example, determine if the system is intended for a single site, different sites or departments, different organizations, different industries, or different regions and countries. Second, identify all variations from one installation to another.

When eliciting portability requirements, consider the following aspects:

- ◆ **DATA PORTABILITY**. Will data files be transferable and run as is on all platforms?
- ◆ **PROGRAM PORTABILITY**. Will the code run exactly the same on all platforms?
- ◆ **END-USER PORTABILITY**. Will the program have the same look and feel across platforms? Will the commands and keystrokes be identical? This flexibility must be designed in from the outset.
- ◆ **DEVELOPER/DOCUMENTATION PORTABILITY**. Will a developer or user be able to use a standard set of skills to understand the program?

7.2 POR



7.2.3 Portability Requirement Examples

- a) All timestamps recorded by the transaction processing system shall be in UTC (Universal Time Coordinated) when placed into permanent storage.
- b) The time zone shall be obvious to the user whenever a time element is displayed.
- c) The product is targeted for sale next year in the European market.
- d) The product is designed to run in business offices, but the intent is to have a version which will run in manufacturing assembly plants.
- e) The system shall be developed for Microsoft Vista® and Macintosh® operating system platforms.
- f) The HomeAccounting software may be ported to any personal computer or workstation environment supporting at least 16-bit color on a 15-inch display monitor, achieving a SPECfp95 benchmark rating of at least 5.0, and having a data storage capacity of at least 8 MB.



7.2.4 Portability Suggested Questions

DATA (WHAT?)

ID	Portability Suggested Questions	Ask this to:
POR1S	What portability requirements must be communicated to suppliers?	<i>identify data needs.</i>
POR2S	What portability requirements must be communicated to customers/users?	<i>identify data needs.</i>
POR3R	Where should portability requirements be stored for accessibility?	<i>identify data storage requirements.</i>
POR4B	What portability requirements from other systems or projects can be reused?	<i>identify existing information.</i>
POR5B	What data files must be portable?	<i>identify data needs.</i>
POR6B	What else should I be asking about portability “data”?	<i>uncover additional requirements.</i>

ROLES (WHO?)

ID	Portability Suggested Questions	Ask this to:
POR7S	How adaptable to technology changes are the customers?	<i>identify customer acceptance.</i>
POR8S	What customers are expected to be early adopters of new technology?	<i>identify customer types.</i>
POR9R	Who will be porting the system to different environments or locations?	<i>identify responsibilities.</i>
POR10R	Who is accountable for the portability requirements?	<i>identify responsibilities.</i>
POR11R	Who might identify the portability requirements?	<i>identify responsibilities.</i>
POR12R	Who is knowledgeable in the portability requirements of the technology being used?	<i>identify potential stakeholders.</i>
POR13B	What else should I be asking about portability “roles”?	<i>uncover additional requirements.</i>

7.2 POR



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PURPOSE (WHY?)

ID	Portability Suggested Questions	Ask this to:
POR14s	How important is portability?	<i>gauge importance.</i>
POR15s	What is the possibility of moving to a new platform?	<i>identify alternatives.</i>
POR16s	What will be different when moving to a new environment?	<i>identify combinations.</i>
POR17s	How do competitors release their products?	<i>identify competition strategies.</i>
POR18s	What copyright, patent, and trademark requirements need to be addressed internationally?	<i>identify corporate assets.</i>
POR19s	What intentions are there to remain an industry leader?	<i>identify industry potential.</i>
POR20s	What local governmental regulations need to be addressed?	<i>identify local controls.</i>
POR21R	What are the required operating environments?	<i>clarify environments.</i>
POR22R	What operating environment is considered the base environment?	<i>establish a base environment.</i>
POR23R	What is the transportability of the software? Hardware?	<i>identify capabilities.</i>
POR24R	When moving to a new platform which components are included?	<i>identify component severability.</i>
POR25R	What other applications will co-exist in the operating environment?	<i>identify comprehensive configuration.</i>
POR26R	How is configuration management coordinated between environments: development, test, and deployment?	<i>identify configuration combinations.</i>
POR27R	What types of database platforms are expected to be supported?	<i>identify configurations.</i>
POR28R	How mature is the industry this system supports?	<i>identify knowledge sources.</i>
POR29R	What constraints exist in the target component configuration?	<i>identify limitations.</i>
POR30R	What are the criteria for selecting new hardware configurations?	<i>identify selection criteria.</i>
POR31R	What are the criteria for selecting new software and firmware configurations?	<i>identify selection criteria.</i>
POR32R	What standard display formats for all data attribute types must be used?	<i>identify standards.</i>
POR33R	What date management and terminology will be used consistently by the system?	<i>identify universal attributes.</i>

7.2 POR



302

The Quest for Software Requirements

PURPOSE (WHY?) (continued)

ID	Portability Suggested Questions	Ask this to:
POR34R	What are the numeric data attributes that must be enforced consistently?	<i>identify universal attributes.</i>
POR35R	What data, time, date and time, attributes must be enforced consistently?	<i>identify universal attributes.</i>
POR36R	What new device types can be anticipated for future implementation?	<i>identify various devices.</i>
POR37R	What plans are being made to move to new environments?	<i>identify vision.</i>
POR38B	What is the growth potential for the industry?	<i>identify future demand.</i>
POR39B	How will this environment change with new systems?	<i>identify impacts due to change.</i>
POR40B	How will this implementation be an industry leader?	<i>identify industry positions.</i>
POR41B	What are possible trade-offs between portability and reusability?	<i>identify possible trade-offs.</i>
POR42B	What are possible trade-offs between portability and reliability?	<i>identify possible trade-offs.</i>
POR43B	What are expectations for new or emerging implementations in the industry?	<i>identify rates of change.</i>
POR44B	How soon will technology changes be available?	<i>identify rates of change.</i>
POR45B	How rapidly are new devices expected to be introduced?	<i>identify rates of change.</i>
POR46B	When an opportunity arises to move to a new environment, how quickly must the business be available?	<i>identify response rate.</i>
POR47B	What else should I be asking about portability "purpose"?	<i>uncover additional requirements.</i>

TIMING (WHEN?)

ID	Portability Suggested Questions	Ask this to:
POR48S	What is the established time zone standard?	<i>clarify standards.</i>
POR49S	How quickly do competitors move to new technology?	<i>identify competition tactics.</i>
POR50B	What else should I be asking about portability "timing"?	<i>uncover additional requirements.</i>

7.2 POR



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

LOGISTICS (WHERE?)

ID	Portability Suggested Questions	Ask this to:
POR51s	What taxes and customs must be managed when adding new countries?	<i>identify differences by country.</i>
POR52s	What languages may be added in the future?	<i>identify potential languages.</i>
POR53s	What language is the base language?	<i>identify the base.</i>
POR54s	Who can approve language translation?	<i>identify translation activities.</i>
POR55s	How will language translation be managed?	<i>identify translation activities.</i>
POR56s	What languages do customers/users speak?	<i>identify various languages.</i>
POR57s	What import/export taxes could apply in other countries?	<i>identify various taxes and fees.</i>
POR58R	What is the capability to partially disassemble the software/hardware for physically moving?	<i>identify constraints.</i>
POR59R	What equipment will need special documentation to cross the borders?	<i>identify customs constraints.</i>
POR60R	What is the likelihood that the software/hardware or product will survive shipping hazards?	<i>identify fragility.</i>
POR61R	What configuration constraints might exist in certain countries?	<i>identify political constraints.</i>
POR62R	How will international support be addressed?	<i>identify support needs.</i>
POR63R	What is the compatibility of the software/hardware with standard moving equipment and vehicles such as trucks and forklifts?	<i>identify various resources.</i>
POR64B	What else should I be asking about portability "logistics"?	<i>uncover additional requirements.</i>

7.2 POR



PROCESS (How?)

ID	Portability Suggested Questions	Ask this to:
POR65R	How stable/mature is the development environment?	<i>identify centers of knowledge.</i>
POR66R	How are enhancement releases managed through multiple configuration environments?	<i>identify configuration combinations and priorities.</i>
POR67R	What interfaces are required? (legacy systems, certain platforms)	<i>identify minimum configurations.</i>
POR68R	When upgrades are made to the systems, what is the release sequence by configuration environment?	<i>identify release schedule priorities.</i>
POR69R	When deploying devices, what configurations must be considered?	<i>identify various combinations.</i>
POR70R	What combinations of components will be used?	<i>identify various combinations.</i>
POR71R	What changes in technology are available and may be possible to take advantage of in the future?	<i>identify various components.</i>
POR72R	What various combinations of system by environment are there?	<i>identify various configurations.</i>
POR73R	What could be the expected changes to development environments? (For example, compilers, languages, versions)	<i>identify various configurations.</i>
POR74R	What component configurations are expected?	<i>identify various configurations.</i>
POR75R	What network configurations are expected?	<i>identify various configurations.</i>
POR76R	What implementation strategies should be used to develop portable software?	<i>identify various configurations.</i>
POR77R	What system/applications are dependent upon other system/applications or components?	<i>identify various dependencies.</i>
POR78R	What system/applications or components are dependent upon the organization's system/application?	<i>identify various dependencies.</i>
POR79R	With what other system/applications or components must the organization's system/application co-exist?	<i>identify various dependencies.</i>
POR80R	On what operating environments will this system be deployed?	<i>identify various environments.</i>
POR81R	What are the expected operating environments?	<i>identify various environments.</i>
POR82R	How many database platforms are anticipated to be used on installations?	<i>identify various environments.</i>

7.2 POR



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

PROCESS (How?) (continued)

ID	Portability Suggested Questions	Ask this to:
POR83R	If the product will be shipped, what considerations must be given to protecting the product during shipping? For example, packing it in cushioned containers, covering it with temporary coatings, or requiring handlers to use removable lifting attachments.	<i>identify various handling techniques.</i>
POR84R	What possible constraints exist in potential component configurations?	<i>identify various limitations.</i>
POR85R	Which components are most likely to move to a new platform?	<i>identify various types of components.</i>
POR86B	What else should I be asking about portability “processes”?	<i>uncover additional requirements.</i>

7.2 POR



7.3 REUSABILITY (REU)

USER CONCERN: How easy is it to convert for use in another system?

RELATED CATEGORIES: Commonality, Leveragability, Modularity

7.3.1 Reusability Definition

Reusability is the extent to which a portion of the software system can be converted for use in another system.

7.3.2 Reusability Discussion

There are two views that typically come to mind when using the term “reusability.” One view of reusability refers to the reuse of requirements documentation from previous projects. The other is the reusability of software functions. This second view is used as the primary intent of reusability described in this book, and we’ll come back to it after a brief discussion on reusing requirements.

Let’s begin with the merits of reusing requirements. In short, requirements reuse saves money and time. According to the experience of Sommerville and Sawyer [Sommerville, 1977], up to 80 percent of the requirements may be more or less the same for similar systems, while other authors indicate as much as 85 percent of the requirements come from existing similar application areas. Even if accurate predictions are difficult to make, significant gains are likely when reusing the requirements across several systems. One could safely expect some reuse of design, code, and test components, thus further reducing development time and cost.

Furthermore, requirements reuse reduces risks. If you are reusing requirements that have already been implemented (verified and validated) in other systems, you might reduce the risks of introducing requirements that are difficult to implement or that interact in undesirable ways with existing requirements.

7.3 REU



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

In the process of requirements elicitation, requirements may be either reused directly or indirectly. Direct reuse occurs when requirements from one system are implemented as requirements for some other system with minimal changes. Whereas, indirect reuse occurs when existing requirements are used in the elicitation process to prompt end-users for their specific requirements for a current project effort.

Let's turn our attention to the primary definition of reusability. Software reuse is the reapplication of knowledge about one system to another similar system in order to reduce the effort to develop and maintain the other system. Therefore, reusability requirements identify functions that already exist and may be partially or fully utilized in developing the system at hand. Reusing the functionality saves development time and costs as analysis and validation have already been demonstrated in the existing system. Reuse can improve reliability and reduce software management risks.

Some narrow views of software reuse include “reuse is the reapplication of source code,” or “reuse is the use of subroutines, modules, or object libraries.” By focusing narrowly on the reapplication of code components, the most highly reusable components tend to be rather small. This is because source code languages have a high degree of specificity. It takes a lot of work to piece together several small components to build a large system. Thus, the cost to build a large system is generally greater than the savings brought about by reuse.

Reusability advocates Ted Biggerstaff and Alan Perlis [Perlis, 1989] have helped to broaden the view of reusability by pointing out that domain standards are essential for the overall coordination of software components. This is similar to standards established for manufacturers of hardware components so that they can plug their components together.

For those looking to create libraries of reusable software, Biggerstaff and Perlis recommend that the library be based on a standard for the domain specific types of the data produced and consumed by the software components in that library. This is only to be accomplished by designing a library of components that have a common architectural guideline that reflects both the nature of the problem domain, as well as the computational complexities of the organization.



Since the emergence of workable standards and their acceptance by the software industry, it is now easier to develop reusable and portable application systems. Development according to standards can minimize the costs of implementing a system on different types of computers. Standards that have been accepted include:

- ◆ Programming language standards, such as COBAL and SQL.
- ◆ Operating system standards, such as Microsoft XP®, Macintosh®, and UNIX®.
- ◆ Networking standards, such as TCP/IP protocols.
- ◆ Graphical User Interface (GUI) window standards, such as Microsoft Windows®.

In summary, the reuse of software source code is a reasonable first step toward reusability. The reuse of design has a greater potential payoff, but will require significant breakthroughs to realize its magnitude potential in a broad automated way.

When eliciting reusability requirements, consider the following aspects:

- ◆ **FEASIBILITY OF SOFTWARE REUSE.**
 - ◊ **Size and complexity.** Generally, as the size and complexity of the software increases, the feasibility for reuse usually decreases. Small modules and components are more easily designed, tested, and maintained.
 - ◊ **Life-cycle phase.** As a component of the system approaches implementation, the less likely it is to be reusable. It is generally easier to reuse a requirement document, for example, than to reuse source code. There are many assumptions that go along with the code such as parameter passing conventions and operating system utilities. Also, reusable code implies that all traceable documentation preceding the code is also reusable and reliable.

7.3 REU



- ◊ **Range of applications.** The range of applications in which the software is intended to be reused can affect its feasibility for reuse. If a component is used within a narrow range of application where the terminology and assumptions are well understood, the component doesn't need rigorous definition. On the other hand, when the range of application is broad, the component must be rigorously defined since the terminology and assumptions would be more varied and less well-known.
 - ◊ **Organizational distance.** This refers to the number of organizational layers that separates the person who reuses a component from the person who initially developed the component. “Within the same department in a company,” and “between departments within the same company,” are examples of organizational distance. If the distance is short, the component is more easily reused since the information needed to reuse the component is likely to be accessible and available. If the distance is long, the information is often more difficult to obtain.
 - ◊ **Variability.** If the component has a high degree of consistent interpretation, then feasibility of reuse will be high. For example, if the component deals with basic math calculations such as addition and subtraction, there is little potential for misinterpretation. Conversely, if the component deals with many variables for calculating interest amounts in multiple currencies, then the chances for consistent interpretation are lower.
- ♦ **POSSIBLE AREAS FOR REUSE.**
- ◊ Language (reusable by people, computers).
 - ◊ Environment (reusable tools).
 - ◊ Methodology (reusable concepts).
 - ◊ Technology (education, measurement, and integration).
 - ◊ Information (restructuring existing knowledge to make it more accessible to humans).



- ◆ **DEVELOPMENT STANDARDS.**
 - ◊ Programming languages.
 - ◊ Operating systems.
 - ◊ Networks.
 - ◊ Graphical User Interfaces.

- ◆ **PORTABILITY TRADE-OFFS.**

7.3.3 Reusability Requirement Examples

- a) The payment subsystem design is based on the payment module from the ALPHA product line. The ePAYZ system should not be modified unless absolutely necessary.
- b) Development of functionality to support the Electronic Funds Transfer (EFT) payment option shall be modularized such that it can be reused by other departments of the organization.
- c) Web applications shall be developed to adhere to HyperText Markup Language (HTML) guidelines and standards.
- d) All software that runs on a client device shall be written in a prevalent programming language such that the software can be run on a personal computer without having to download a supporting environment.
- e) The hard-copy materials in the Library of Congress shall be converted to an electronic form such that the existing knowledge can be flexibly presented in different formats for use in different contexts. (Note that this involves more than just putting the materials into computer repositories and accessing the knowledge through information retrieval systems. Accessibility must also comply with copyright laws.)

7.3 REU



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

7.3.4 Reusability Suggested Questions

DATA (WHAT?)

ID	Reusability Suggested Questions	Ask this to:
REU1s	What documents from other projects could be useful?	<i>identify sources of information.</i>
REU2s	What current presentation standards exist?	<i>identify standards.</i>
REU3B	What else should I be asking about reusability "data"?	<i>uncover additional requirements.</i>

ROLES (WHO?)

ID	Reusability Suggested Questions	Ask this to:
REU4s	Who may have an issue with providing reusability?	<i>identify concerns.</i>
REU5s	Who may have knowledge of similar projects or similar requirements?	<i>identify resources.</i>
REU6s	Who else has done a similar project in the past?	<i>identify sources of information.</i>
REU7s	Who is onboard with reusability?	<i>identify supporters.</i>
REU8s	What other projects in the organization are compatible, or that cover substantially the same domains or work areas?	<i>identify various projects.</i>
REU9s	What other products involve the same users and thus have similar usability requirements?	<i>identify various user bases.</i>
REU10s	What other types of uses of the components are expected?	<i>identify various user types.</i>
REU11B	What else should I be asking about reusability "roles"?	<i>uncover additional requirements.</i>



PURPOSE (WHY?)

ID	Reusability Suggested Questions	Ask this to:
REU12s	What assumptions from other projects apply?	<i>identify alternatives.</i>
REU13s	What constraints have already been defined for another project?	<i>identify alternatives.</i>
REU14s	What are the advantages of developing software with reusable components?	<i>identify business objectives.</i>
REU15s	What expectations are there to expand the system and be able to reuse elements?	<i>identify expansion opportunities.</i>
REU16s	How much funding is available for reusability in the project?	<i>identify limits.</i>
REU17s	Which of these components is company policy?	<i>identify mandated components.</i>
REU18s	What other projects have been launched that are similar?	<i>identify synergies.</i>
REU19s	What other uses for similar applications apply?	<i>identify various alternatives.</i>
REU20s	What specific elements will be used in other applications?	<i>identify various applications.</i>
REU21s	What is the growth potential for similar applications?	<i>identify various options.</i>
REU22s	What standards have been defined and implemented in previous projects?	<i>identify various sources of information.</i>
REU23R	What are problems associated with developing software with reusable components?	<i>identify business risks.</i>
REU24R	What industry standard elements can function in isolation?	<i>identify separable elements.</i>
REU25B	What are relevant facts from recent projects?	<i>identify lessons learned.</i>
REU26B	What are possible trade-offs between reusability and portability?	<i>identify possible trade-offs.</i>
REU27B	Where has the company been successful with past reusability efforts?	<i>identify successes.</i>
REU28B	What is the scope of projects for adjacent systems?	<i>identify various extensions.</i>
REU29B	What components are so standard they have usability elsewhere?	<i>identify various options.</i>
REU30B	Which of these components are fundamental to the application domain and thus will be implemented in future projects or have been implanted in previous projects?	<i>identify various options.</i>
REU31B	What else should I be asking about reusability “purpose”?	<i>uncover additional requirements.</i>

7.3 REU



PART TWO: RIGHT APPROACH, RIGHT QUESTIONS

TIMING (WHEN?)

ID	Reusability Suggested Questions	Ask this to:
REU32s	What triggering events are the same as in other business processes?	<i>identify similar events.</i>
REU33s	What event triggers could be used in new ways as a result of this project?	<i>identify similar events.</i>
REU34s	What business processes have the same timing of use?	<i>identify similar processes.</i>
REU35s	What future business processes could have the same timing of use?	<i>identify similar processes.</i>
REU36B	What else should I be asking about reusability “timing”?	<i>uncover additional requirements.</i>

LOGISTICS (WHERE?)

ID	Reusability Suggested Questions	Ask this to:
REU37s	What processes are in use in the same locations as the processes from this project?	<i>identify similar processes.</i>
REU38s	What future processes may be deployed in the same locations?	<i>identify similar processes.</i>
REU39B	What else should I be asking about reusability “logistics”?	<i>uncover additional requirements.</i>



PROCESS (How?)

ID	Reusability Suggested Questions	Ask this to:
REU40s	What functionality is considered pilot from this project?	<i>classify features.</i>
REU41s	How will pilot functionality be monitored?	<i>identify metrics.</i>
REU42s	How will pilot functionality be expanded?	<i>identify potential implementations.</i>
REU43s	What capabilities can be adopted and modified for this project?	<i>identify similar capabilities.</i>
REU44s	What capabilities can be adapted to other processes as a result of this project?	<i>identify similar capabilities.</i>
REU45s	What similar functionality exists?	<i>identify similar functionality.</i>
REU46s	How will pilot functionality be implemented?	<i>prioritize functionality.</i>
REU47R	What implementation strategies should be used to develop reusable software?	<i>identify improvement areas.</i>
REU48R	How must software processes evolve to incorporate reuse?	<i>identify reusable processes.</i>
REU49R	How can software components be generalized so that they are usable across a range of systems?	<i>identify reuse opportunities.</i>
REU50R	How do application generators support the reuse of domain concepts?	<i>identify reuse opportunities.</i>
REU51R	How can entire application systems be reused by making them available on a range of machines?	<i>identify reuse opportunities.</i>
REU52B	What else should I be asking about reusability “processes”?	<i>uncover additional requirements.</i>

7.3 REU



7.4 SUGGESTED READING

Mastering the Requirements Process, by Suzanne Robertson and James Robertson, [Robertson, 1999].

Chapter 12 provides a lengthy discussion on **reusing** requirements.

Requirements Engineering: A Good Practice Guide, by Ian Sommerville and Pete Sawyer, [Sommerville, 1977]. Requirements elicitation guideline 4.13 discusses **reuse** requirements, including suggested steps for eliciting direct and indirect reuse requirements.

Software Engineering, 8th Edition, by Ian Sommerville, [Sommerville, 2007]. Chapter 18 describes the benefits and challenges in developing software for **reusability**.

Software Quality Engineering: A Total Technical and Management Approach, by Michael Deutsch and Ronald Willis, [Deutsch, 1988]. Chapter 3 describes the classification of nonfunctional requirements, which includes definitions for **interoperability**, **portability**, **reusability** and others.

Software Quality Metrics Enhancements, Volume 1, by James McCall and Mike Matsumoto, [McCall, 1980]. This research report presents a software quality framework consisting of quality factors, criterion, and metrics. Section 1 provides definitions for several software quality factors such as **interoperability**, **portability**, and **reusability**.

Software Requirements, 2nd Edition, by Karl Wiegers, [Wiegers, 2003]. Chapter 12 defines several software quality attributes, such as **interoperability**, **portability**, and **reusability**.

Software Requirements: Objects, Functions, & States, by Alan M. Davis, [Davis, 1993]. Davis discusses **portability**, reliability, efficiency, and human engineering (usability) in Chapter 5, “Specifying Nonbehavioral Requirements.”

System Requirements Engineering, by Pericles Loucopoulos and Vassilios Karakostas, [Loucopoulos, 1995]. Techniques for **reuse** of requirements are presented in Chapter 3, section 7.

Software Reusability, Volume 1: Concepts and Models, edited by Alan Perlis and Ted Biggerstaff, [Perlis, 1989]. This is a collection of works by various authors about software **reusability**.





REFERENCES

- [Alexander, 2002]** Alexander, Ian F. and Richard Stevens, 2002. *Writing Better Requirements*. Addison-Wesley. ISBN 0-321-13163-0.
- [Boehm, 1976]** Boehm, Barry W., J. Randall Brown, and Myron Lipow, 1976. “Quantitative Evaluation of Software Quality,” International Conference on Software Engineering ICSE, *Proceedings of the 2nd ICSE*.
- [Boehm, 1988]** Boehm, Barry W., 1988. “A Spiral Model of Software Development and Enhancement,” *IEEE Computer*, volume 21, No. 5, pp. 61–72.
- [BRG, 2000]** Business Rules Group, July 2000. *Defining Business Rules: What Are They Really? (3rd Edition)*. Available online from <http://www.BusinessRulesGroup.org>.
- [Charette, 1990]** Charette, Robert, 1990. *Applications Strategies for Risk Analysis*. McGraw-Hill. ISBN 0-07-010888-9.
- [Chrassis, 2007]** Chrassis, Mary Beth, Mike Konrad, and Sandy Shrum, 2007. *CMMI, 2nd Edition: Guidelines for Process Integration and Product Improvement*. Addison-Wesley. ISBN 0-321-27967-0.
- [Chung, 2000]** Chung, Lawrence, Brian A. Nixon, Eric Yu, and John Mylopoulos, 2000. *Non-Functional Requirements In Software Engineering*. Kluwer Academic Publishers. ISBN 0-7923-8666-3.
- [Davis, 1993]** Davis, Alan M., 1993. *Software Requirements: Objects, Functions, and States*. Prentice Hall. ISBN 0-13-805763-X.



REFERENCES

- [**Deutsch, 1988**] Deutsch, Michael S. and Ronald R. Willis, 1988. *Software Quality Engineering: A Total Technical and Management Approach*. Prentice Hall. ISBN 0-13-823204-0.
- [**Dorfman, 2000**] Dorman, Merlin and Richard H. Thayer, editors. Institute of Electrical and Electronic Engineers, Inc. (IEEE), 2000. *Software Engineering*. John Wiley & Sons. ISBN 0-8186-7609-4.
- [**Ellison, 2002**] Ellison, Robert, et al., 2002. *Foundations of Survivable Systems Engineering. Crosstalk: The Journal of Defense Software Engineering*. <http://www.stsc.hill.af.mil/crosstalk/2002/07/ellison.html>.
- [**Ferdinandi, 2002**] Ferdinandi, Patricia L., 2002. *A Requirements Pattern: Succeeding in the Internet Economy*. Addison-Wesley. ISBN 0-201-73826-0.
- [**Gause, 1989**] Gause, Donald C. and Gerald M. Weinberg, 1989. *Exploring Requirements: Quality Before Design*. Dorset House Publishing. ISBN 0-932633-13-7.
- [**Gilb, 1988**] Gilb, Tom, 1988. *Principles of Software Engineering Management*. Addison-Wesley. ISBN 0-201-19246-2.
- [**Gilb, 2005**] Gilb, Tom, 2005. *Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage*. Butterworth-Heinemann. ISBN 0-750-66507-6.
- [**Gottesdiener, 2002**] Gottesdiener, Ellen, 2002. *Requirements By Collaboration: Workshops for Defining Needs*. Addison-Wesley. ISBN 0-201-78606-0.
- [**Grady, 2006**] Grady, Jeffrey, 2006. *System Requirements Analysis*. Academic Press. ISBN 0-12-088514-5.
- [**Hay, 2003**] Hay, David C., 2003. *Requirements Analysis: From Business Views to Architecture*. Prentice Hall PTR. ISBN 0-13-028228-6.
- [**Hooks, 2001**] Hooks, Ivy F. and Kristin A. Farry, 2001. *Customer-Centered Products: Creating Successful Products Through Smart Requirements Management*. Amacom. ISBN 0-8144-0568-1.
- [**Hull, 2005**] Hull, Elizabeth, Ken Jackson, and Jeremy Dick, 2005. *Requirements Engineering, 2nd Edition*. Springer. ISBN 1-85233-879-2.



- [IBM, 2009] Rational Software Corporation, 2002. *Rational Unified Process*. Available online from <http://www.ibm.com/software/rational/>.
- [IEEE, 1990] Institute of Electrical and Electronics Engineers (IEEE). IEEE Std 610.12-1990: “IEEE Standard Glossary of Software Engineering Terminology.” IEEE Computer Society Press.
- [IEEE, 1998] Institute of Electrical and Electronics Engineers (IEEE). IEEE Std 1233-1998: “IEEE Guide for Developing System Requirements Specifications.” IEEE Computer Society Press.
- [IIBA, 2009] International Institute of Business Analysis, 2009. *A Guide to the Business Analysis Body of Knowledge (BABOK®), release 2.0*. Available online from <http://www.theiiba.org>.
- [ISO, 1991] ISO/IEC 9126, International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), 1991. *Software Engineering—Product Quality*. Available online from <http://www.iso.org>.
- [Keller, 1990] Keller, Steven E., Laurence G. Kahn, and Roger B. Panara, 1990. “Specifying Software Quality Requirements with Metrics,” reprinted in *System and Software Requirements Engineering*. IEEE Computer Society Press. ISBN 0-8186-8921-8.
- [Kotonya, 1998] Kotonya, Gerald and Ian Sommerville, 1998. *Requirements Engineering: Processes and Techniques*. John Wiley & Sons. ISBN 0-471-97208-8.
- [Kruchten, 1999] Kruchten, Philippe, 1999. *The Rational Unified Process: An Introduction*. Addison-Wesley. ISBN 0-201-60459-0.
- [Lauesen, 2002] Lauesen, Soren, 2002. *Software Requirements: Styles and Techniques*. Addison-Wesley. ISBN 0-201-74570-4.
- [Leffingwell, 2003] Leffingwell, Dean and Don Widrig, 2003. *Managing Software Requirements, 2nd Edition: A Use Case Approach*. Addison-Wesley. ISBN 0-321-12247-X.
- [Loucopoulos, 1995] Loucopoulos, Pericles and Vassilios Karakostas, 1995. *System Requirements Engineering*. McGraw-Hill Book Company. ISBN 0-07-707843-8.
- [Lyu, 1996] Lyu, Michael R., 1996. *Handbook of Software Reliability Engineering*. McGraw-Hill. ISBN 0-07-039400-8.



REFERENCES

- [**McCall, 1980**] McCall, James and Mike Matsumoto, 1980. "Software Quality Metrics Enhancements, Volume 1, Final Technical Report," Rome Air Development Center (RADC), General Electric Co.
- [**Musa, 2004**] Musa, John D., 2004. *Software Reliability Engineering: More Reliable Software Faster and Cheaper, 2nd Edition*, Arthurhouse. ISBN 1-418-49388-0.
- [**Perlis, 1989**] Perlis, Alan J. and Ted J. Biggerstaff, editors, 1989. *Software Reusability*, Volume 1: Concepts and Models. ACM Press and Addison-Wesley. ISBN 0-201-08017-6.
- [**Pfleeger, 2003**] Pfleeger, Charles P. and Shari Lawrence Pfleeger, 2003. *Security in Computing, 3rd Edition*. Prentice Hall. ISBN 0-13-035548-8.
- [**Pullum, 2001**] Pullum, Laura L., 2001. *Software Fault Tolerance Techniques and Implementation*. Artech House. ISBN 1-58053-137-7.
- [**Robertson, 1999**] Robertson, Suzanne and James Robertson, 1999. *Mastering the Requirements Process*. Addison-Wesley. ISBN 0-201-36046-2.
- [**Robertson, 2006**] Robertson, Suzanne and James Robertson, 2006. *Mastering the Requirements Process, 2nd Edition*. Addison-Wesley. ISBN 0-321-41949-9.
- [**Royce, 1970**] Royce, Winston W., 1970. "Managing the Development of Large Software Systems: Concepts and Techniques," in *WESCON Technical Papers, volume 14*. Reprinted in *Proceedings of the Ninth International Conference on Software Engineering*, 1987, pp. 328-338.
- [**Sommerville, 1977**] Sommerville, Ian and Pete Sawyer, 1977. *Requirements Engineering: A Good Practice Guide*. John Wiley & Sons. ISBN 0-471-97444-7.
- [**Sommerville, 1992**] Sommerville, Ian, 1992. *Software Engineering, 4th Edition*. Addison-Wesley. ISBN 0-201-56529-3.
- [**Sommerville, 2007**] Sommerville, Ian, 2007. *Software Engineering, 8th Edition*. Addison-Wesley. ISBN 0-321-31379-8.
- [**Standish, 1995**] The Standish Group International, Inc., 1995. *The CHAOS Report*. Standish Group International.



- [Thayer, 1990] Thayer, Richard H. and Merlin Dorfman, editors. Institute of Electrical and Electronic Engineers, Inc., 1990. *System and Software Requirements Engineering*. IEEE Computer Society Press. ISBN 0-8186-8921-8.
- [Thayer, 2000] Thayer, Richard H. and Merlin Dorfman, editors. Institute of Electrical and Electronic Engineers, Inc., 2000. *Software Requirements Engineering, 2nd Edition*. IEEE Computer Society Press. ISBN 0-8186-7738-4.
- [Tian, 2005] Tian, Jeff, 2005. *Software Quality Engineering*. IEEE Computer Society Press. ISBN 0-471-71345-7.
- [Wallace, 1996] Wallace, Dolores R. and Laura M. Ippolito, 1996. “Verifying and Validating Software Requirements Specifications,” reprinted in *Software Requirements Engineering, 2nd Edition*. IEEE Computer Society Press. ISBN 0-8186-7738-4.
- [Wiegers, 2003] Wiegers, Karl E., 2003. *Software Requirements, 2nd Edition*. Microsoft Press. ISBN 0-7356-1879-8.
- [Withall, 2007] Withall, Stephen, 2007. *Software Requirement Patterns*. Microsoft Press. ISBN 0-7356-2398-8.
- [Zachman, 1987] Zackman, John, 1987. “A Framework for Information Systems Architecture,” *IBM Systems Journal*, 26:3 (IBM Publication G321-5298). Available online from <http://www.almaden.ibm.com/research/>.
- [Zackman, 2009] Zackman, John A., 2009. *The Zachman Enterprise Framework 2™*. Available online from <http://www.zachmaninternational.com/>.





ABOUT THE AUTHOR

Roxanne E. Miller is a self-proclaimed “Requirements Super Freak.” She has been involved in the Information Technology (IT) industry since 1984. She has been consulting on requirements process improvement and business analysis practices for over 15 years. Roxanne earned a bachelor’s degree in Management Information Systems (MIS) at the University of Wisconsin-Eau Claire, Eau Claire, Wisconsin, USA. Prior to founding Requirements Quest® in 2001, Roxanne worked as a consultant, primarily in the banking and insurance industries in the roles of Programmer/Analyst, Business Analyst, and Requirements Management Process Owner and Implementation Leader.

As a result of her expertise, passion, and energizing presentation style, Roxanne is a frequent speaker at business analysis industry conferences. Furthermore, she is a “back-by-popular-demand” presenter for non-profit organizations, such as the Project Management Institute (PMI), the Software Process Improvement Network (SPIN), and Wisconsin Information Systems Quality Assurance (WISQA). Roxanne also serves on a panel of requirements experts for Search Software Quality, an online site that offers help in developing,



ABOUT THE AUTHOR

deploying, and managing software quality. For more information, visit <http://www.SearchSoftwareQuality.com>.

Roxanne is an active member and advocate of the International Institute of Business Analysis (IIBA[®]), and is a Certified Business Analysis Professional™ (CBAP[®]) recipient. She serves as President of the IIBA[®] Greater Madison Chapter, Wisconsin, USA. Additionally, Roxanne helped Wisconsin IIBA[®] chapters unite and launch an annual event, Wisconsin Business Analyst Development Day (WI BADD), which is devoted to education, development, and networking opportunities for business analysis professionals. To learn more about how IIBA[®] can benefit you and your organization, visit <http://www.theiiba.org>.

I met Roxanne as a result of glowing recommendations from several of my team members who had attended one of her requirements seminars. Roxanne is the person I confer with when I have questions pertaining to requirements development and solution design. She truly is a certified requirements guru, who provides focused, valued, and engaged training experiences. Roxanne is very responsive, helpful in many ways, and well connected in the Wisconsin BA community. She is instrumental in fostering networking opportunities for people who are genuinely interested in moving the business analysis profession forward.

—Carl Henzel

I've worked closely with Roxanne in her role as President of the IIBA Madison Chapter, and participated in one of her requirements workshops. I have been uniformly impressed with the knowledge base, communication skills and professionalism that she brings to the table. I've recommended her services to a number of colleagues without hesitation.

—John Argentiero, Senior Business Analyst

There are a lot of people who enjoy their work. There are very few who have the passion and success in their career that Roxanne has. We have collaborated on many projects. On each one, she creates a unique, enjoyable environment and "can do" motivation in the team. Roxanne is highly credible and works diligently to transfer her expertise to help her clients accomplish their goals effectively. I would recommend Roxanne for any requirements-related initiative of any size.

—Jim Dawkins





ABOUT REQUIREMENTS QUEST

Requirements Quest is an industry leader in requirements management and business analysis consultancy. Requirements Quest is renowned for its Requirements Quest Process™, a superior, repeatable approach to requirements elicitation, analysis, representation, and validation, which is deployed on client consulting engagements. Requirements Quest's core competency is Requirements Management Process Improvement, as well as Business Analysis Skills Development and Coaching. As a solution-driven consulting company, Requirements Quest is devoted to working with organizations to improve their requirements development and management processes, and committed to developing the skills of business analysis practitioners.

Requirements Quest is a corporate sponsor of the International Institute of Business Analysis (IIBA®), and founder of the Greater Madison Chapter, Wisconsin, USA. Furthermore, Requirements Quest is an active sponsor of IIBA® chapters, as well as project management and business analysis educational events. Requirements Quest is a chartered Endorsed Education Provider of IIBA®, and has enhanced the skills of thousands of practitioners in business analysis and requirements management techniques. Requirements Quest's unique training provides a



role-based approach to implementing requirements practices. Quality requirements and successful software projects are not the sole responsibility of the business analyst role; rather, quality requirements are the result of collaboration from multiple stakeholders.

The Requirements Quest logo (“the hand”) symbolizes the importance of user involvement, both internal and external, as a critical factor for project success. The goal of a successful Requirements Management Process is to get a consistent interpretation of the requirements from all the stakeholders. Requirements Quest applies a proven repeatable, five-stage Requirements Management Process that significantly elevates the involvement of the right stakeholders, and fosters a collaborative team approach with increased commitment to the requirements.

The five stages of the process are symbolized by the five fingers on “the hand.” The first four fingers represent the iterative activities of requirements development: Elicitation, Analysis, Representation, and Validation. Upon successful refinement, detailing, and approval, the requirements are baselined in the Change Control stage (represented by the thumb) where requests for change are monitored to maintain scope.



Requirements Quest's highly-credible consultants are IIBA® Certified Business Analysis Professionals™ (CBAP®). They bring years of practical experience and knowledge into your organization, and work diligently to transfer that knowledge and promote the development of your own internal requirements expertise.

Requirements Quest's goal is **bringing your business into focus®** so you can achieve effective and efficient business results. To learn more about Requirements Quest's consulting services, or to contact Roxanne Miller, please visit <http://www.RequirementsQuest.com>.



Roxanne and Requirements Quest takes an approach to business analysis that makes understanding intuitive and results efficient. She is a leading expert in coaching and developing business analysis resources and business requirements practices. Her tutelage, insight, and facilitative skills allow everyone around her to become more effective. People that implement her approach have an impact on their organizations and deliver better results.

—Mark Swiderski, President, Praxilient, Inc.

Roxanne's passion and depth of knowledge about business analysis is inspiring. Her presentations are full of energy, enthusiasm and fun! We have incorporated many of the tools and techniques delivered in Requirements Quest's training seminars to our Project Process with immediate improvement and success. Business Analysts, and other project stakeholders alike, reap a variety of methods that can be used to navigate through the challenges faced on a daily basis.

—Clare Jones, Director, Wipfli, LLP

Roxanne is a leader in the business analysis and requirements engineering industry. She has coached and mentored countless numbers of BAs and people in other software development roles, on the importance of documenting requirements before developing the solution to a business problem. Through her founding of Requirements Quest and the Greater Madison IIBA Chapter, Roxanne has been a mentor to me and a role model for many others. I, along with the entire BA community in Wisconsin, owe Roxanne a debt of gratitude for her contributions.

—David DeBruine, Owner, DeBruine & Associates, LLC



INDEX

A

A-Ask Questions strategy (STAR), 81–84
access control. *See* Access Security requirements
Access Security (ACS) requirements, 127–139
 defined, 119, 120, 127
 examples, 130
 operational requirements, 90, 120
 overview, 127–129
 suggested questions, 131–139
 terminology comparisons, 116–117
 user needs, 119
accessibility, 243
accuracy, 166
ACS. *See* Access Security requirements
action items (agendas), 59
active listening, 79
adaptability. *See* Flexibility requirements; transition requirements
adaptive maintenance, 243
agendas, 57–60, 75, 77
algorithm analysis, 267
ambiguity, 83
analysis
 algorithm analysis, 267
 boundary value analysis, 267
 business analysis, 36
 consistency analysis, 267
 control flow analysis, 267
 coverage analysis, 267
 database analysis, 267
 dataflow analysis, 267
 enterprise analysis, 32
 event tree analysis, 268
 fault tree analysis, 269
 form analysis, 32

interface analysis, 268
in maintenance process cycle, 244
mutation analysis, 268
of potential stakeholders, 48–50
regression analysis, 269
scenario analysis, 33
software failure mode, effects and critical analysis (SFMECA), 269
Survivable Systems Analysis, 194–195
task analysis, 33
in Verifiability requirements, 265–283
in verification/validation techniques, 266–270
analysis stage (requirements process), x, 9, 10, 11, 18
answers to interview questions, 67, 78, 79–81
applications. *See also* systems
 errors, 206, 245
 portability, 299
 programming language standards, 309
archiving data, 154
areas of interest, 22–23. *See also* Data area of interest;
 Logistics area of interest;
 Process area of interest; Purpose area of interest;
 Roles area of interest;
 Timing area of interest
Ask Questions strategy (STAR), 81–84
attacks on systems, 165
attention, in interviews, 78–79
attributes (nonfunctional definitions), 103, 104
authentication, 129
authenticity (data), 166
authorization, 128–129, 274
availability
 interviewees, 75
 stakeholders, 51
 systems. *See* Availability requirements
Availability (AVL) requirements, 140–151
 defined, 119, 120, 140
 examples, 142
 operational requirements, 90, 120



overview, 140–141
suggested questions, 143–151
terminology comparisons, 116–117
user needs, 119
“availability windows,” 140
AVL. *See* Availability requirements

B

back-to-back testing, 267
backups, 166
backward error recovery, 196
behaviors, 100
Biggerstaff, Ted, 308
blocking/unlocking users, 129
body language, 78, 79
Boehm, Barry, 110–111
boundary value analysis, 267
“bucket” metaphors, 99–103, 122
budgets, 26. *See also* costs
bugs, 206
business analysis, 36
business artifacts, 32
business locations, 256
business modeling, 18, 20, 21, 22
business practices, 228
business requirements
 defined, 6–7
 development of, 11
 question format, 63
business rules, 27–28

C

calendars, 229
capabilities, expanding, 255
cell phones, 56
certification (testing), 270
certification standards, 13, 28. *See also* standards

change control stage (requirements process), x, 9, 10, 11
change management, 18
change requests, 245
characteristics, requirements and, 100
check digits, 196
check-sums, 196
checklists
 interviews, 65
 requirement categories, 114–117
 stakeholders, 40
classifications (nonfunctional requirements)
 Boehm’s quality attribute tree, 110–111
 comparing categories, 114–117
 future classifications, 122
 Summerville’s classification, 112, 113
 Withall’s requirement patterns, 112–114
clients, 34
closed-ended questions, 68, 69, 77
coding (development cycles), 14, 15
collaboration, ix
communication. *See also* interviews
 agendas, 58
 building rapport, 46
 common errors in, 83
 “information hoarders,” 67
 interviews, not interrogations, 73
 “leading the witness” questions, 71
 prepared questions, 61
 visual aids, 44, 58
compatibility (systems), 298
complexity (systems), 244
compliance with standards/constraints, 266
components (requirements frameworks), 20, 21
configuration
 excluding plans from frameworks, 26
 in iterative approach, 18
confirming schedules, 58
connecting systems, 285
consistency analysis, 267
constraints, 100, 103, 104, 112
construction stage (software development), 16–18, 20, 22
content-free questions, 68, 69
contingencies for storage capacity, 155
contradictions in interviews, 83
control faults, 273
control flow analysis, 267



conversions, software, 298–299
 corrective system maintenance, 242
 corrupting system failures, 183
 costs
 nonfunctional requirements and, 106
 requirements management and, 15
 reusing requirements specifications, 307
 stakeholder availability and, 45
 system failures, 181
 countries, adapting software for, 228
 courtesy, 74
 coverage analysis, 267
 customers, 15, 34
 cyclomatic complexity code measure, 244

D

damage assessments, 195, 196
 data (information)
 archiving, 154
 authenticity, 166
 backing up, 166
 data exchange, 287
 data relationships, 166
 encryption, 167
 portability, 299
 precision, 166
 protection, 128
 restoring, 166
 in software requirements, 4
 viewpoints on, 19
 Data area of interest (what questions)
 defining in requirements, 4–6
 vs. "how" questions, 24–25
 in requirements frameworks, 20, 22, 91
 suggested questions
 Access Security questions, 131–132
 Availability questions, 143
 Efficiency questions, 156
 Flexibility questions, 231–232
 Integrity questions, 168–171

Interoperability questions, 290
 Maintainability questions, 247
 Portability questions, 301
 Reliability questions, 185
 Reusability questions, 312
 Scalability questions, 258
 Survivability questions, 198–199
 Usability questions, 209–211
 Verifiability questions, 276
 data faults, 273
 database analysis, 267
 dataflow analysis, 267
 dates, 229
 de-authentication, 129
 deadlines, 15
 decision tables, 268
 decisions (maintenance process cycle), 245
 degradations in system performance, 256
 delegating authority, 128
 demonstrations (software), 265–283
 deployment (systems), 18

design
 "how" and "why" questions *vs.*, 73
 in iterative approach, 18
 in requirements frameworks, 20, 22
 in software development cycles, 12, 15
 design-level requirements, 207
 design patterns, 112–114, 115
 detailed answers to questions, 68, 71–72
 Deutsch, Michael, 114, 116–117
 development process requirements, 207
 direct questions, 70
 direct reuse of requirements, 308
 discard guidance (software), 243
 disciplines (iteration tasks), 17–18
 distractions during interviews, 56
 documentation, 32, 299
 downtime, 141
 drafts (interview materials), 60–61



E

ease of data entry, 207
ease of handling (systems), 207
Efficiency (EFC) requirements, 152–164
 defined, 120, 121, 152
 examples, 155
 operational requirements, 90, 121
 overview, 152–155
 suggested questions, 156–164
 terminology comparisons, 116–117
 user needs, 121
efficiency trade-offs, 109
elaboration questions, 70–71
elaboration stage (software development), 16–18
elicitation questions. *See* questions
elicitation stage (requirements process), x, 9, 10, 11
Ellison, Robert, 194
encryption, 167
end-users, 34. *See also* users
engineering standards, 288
enhancement requests, 32
enterprise analysis, 32
environments (system)
 adapting to, 285
 in iterative approach, 18
 system-level requirements, 8
 transferring systems between, 298
error seeding, 268
"et cetera" answers, 80
evaluating interviews, 85
event tree analysis, 268
exception handling, 196
exception management faults, 274
exchange formats (files), 287
expanding software capabilities, 255
expectations
 in interviews, 66–67
 of users, 245
expertise
 analyzing, 48–50
 indicating levels of, 45
 potential stakeholders, 42, 43–44

extendability (systems), 227
external requirements, 112
eye contact, 78

F

facts, in interviews, 72, 83
failures, system. *See* system failures
fault-free system operation, 181
fault tolerance, 195–196. *See also* Survivability requirements
fault tree analysis, 269
faults
 classes of, 273
 detection, 195
 fault avoidance, 181
 fault tolerance, 195–196. *See also* Survivability requirements
 fault tree analysis, 269
 recovery from, 195, 196
 repairing, 196
features, defined, 8
feedback loops in development, 15
files
 destruction, 165
 formats, 287
finite state machines, 268
Flexibility (FLX) requirements, 227–241
 defined, 120, 121, 227
 examples, 230
 overview, 227–230
 revision requirements, 90, 121, 226
 suggested questions, 231–241
 terminology comparisons, 116–117
 user needs, 121
flexibility trade-offs, 109
FLX. *See* Flexibility requirements
follow-up actions
 interviews, 59
 maintenance process cycle, 245
follow-up questions, 67, 81–83, 95
"foolproof" design, 127
form analysis, 32



format of interviews, 59, 75–76
 forward error recovery, 196
 frameworks (requirements), 19–23, 26
 functional requirements
 "bucket" metaphor, 122
 data and, 22
 defined, iii, 99–106
 vs. nonfunctional, 101–103
 system requirements and, 8, 101–102
 functional testing, 268
 functions
 functional requirements and, 100
 reusing, 307
 viewpoints on, 19
 future requirements, planning for, 255–256

G

geographic regions, 229
 Gilb, Tom, 116, 117
 goals of interviews, 75
 Graphical User Interface standards, 309
 growth, scaling with, 255–256
 guidelines, adherence to, 207

H

hard-drive backups, 166
 hardware
 failures, 165
 recycling, 256
 transferring systems between, 298
 Hay, David, 127
 hidden functions, 128
 housekeeping tasks, 141
 "how" questions. *See also* Process area of interest
 vs. design questions, 73

details and, 71
 direct questions, 70
 as not defining requirements, 24–25
 human engineering, 206

ililities." *See* nonfunctional requirements
 implementation (system)
 excluding plans from frameworks, 26
 in iterative approach, 18
 in software development cycles, 14, 15
 standards and, 288
 implementation plans, 26
 inactive users, 129
 inception phase (software development), 16–18
 increasing processing loads, 255
 indirect reuse of requirements, 308
 industries
 flexibility requirements and, 228
 partnerships and standards, 288
 information. *See* data (information)
 "information hoarders," 67
 input/output faults, 274
 inspection checks for faults, 273
 inspections. *See* software inspections
 installation requirements, 274–275
 INT. *See* Integrity requirements
 integration, 108
 Integrity (INT) requirements, 165–180
 defined, 120, 121, 165
 examples, 167
 operational requirements, 90, 121
 overview, 165–167
 suggested questions, 168–180
 terminology comparisons, 116–117
 user needs, 121
 integrity trade-offs, 109
 interchangeability (system), 228, 243
 interfaces
 changing, 287
 faults, 273



interface analysis, 268
standards, 309
testing, 268
Interoperability (IOP) requirements, 287–297
defined, 120, 122, 287
examples, 289
overview, 287–288
suggested questions, 290–297
terminology comparisons, 116–117
transition requirements, 90, 122, 285–286
interoperability trade-offs, 109
interpretations, interviewees', 83
interviews
 agendas, 57–60
 asking right questions, 66–74
 checklist for, 65
 distributing materials for, 60–61
 expectations, 66–67
 facts *vs.* opinions, 72
 note takers, 59, 61–62
 notes, typing, 84
 phone interviews, 77
 prepared questions, 63–64
 preparing for, 55–65
 question types, 68–74
 scheduling, 56–57, 82
 as source for requirements, 33
 STAR interviews, 74–85
 styles of, 76
 success criteria, 64
introductions at interviews, 75
inventory requirements, 20. *See also* Data area of interest
IOP. *See* Interoperability requirements
Ippolito, Laura M., 266–270
ISO/IEC 9125, 116
iterations
 appropriate questions during, 94
 defined, 17
 iterative approach (software development), 14, 16–18
 in requirements management, 10
"ities." *See* nonfunctional requirements

K

Keller, Steven E., 114, 116–117
keystroke counts, 207
Kruchten, Philippe, 16

L

laddering questions, 71–72
languages, 229
Lausen, Soren, 206, 207
"leading the witness" questions, 71
learning curves (systems), 207
libraries of reusable software, 308
life cycles of reusable software, 309
likability (systems), 207
Listen strategy (STAR), 80, 81
listening, 78, 79
localization, 228
logging on or off, 129
Logistics area of interest (where questions)
 defining in requirements, 4–6
 in requirements frameworks, 20, 22, 23, 92
suggested questions
 Access Security questions, 137–138
 Availability questions, 149
 Efficiency questions, 162–163
 Flexibility questions, 238–239
 Integrity questions, 178–179
 Interoperability questions, 296
 Maintainability questions, 252–253
 Portability questions, 304
 Reliability questions, 190
 Reusability questions, 314
 Scalability questions, 263
 Survivability questions, 203
 Usability questions, 219–220
 Verifiability questions, 282



M

maintainability
 defined, 182
 trade-offs, 109
 Maintainability (MNT) requirements, 242–254
 defined, 120, 121, 242
 examples, 246
 maintenance process cycle, 244–245
 metrics, 243–244
 overview, 242–245
 potential issues, 245
 revision requirements, 90, 121, 226
 suggested questions, 247–254
 terminology comparisons, 116–117
 user needs, 121
 maintenance (system), 141, 244
 manuals (documentation), 32, 299
 marketing surveys, 32
 Matsumoto, Mike, 114, 116–117
 McCall, James, 114, 116–117, 118
 mean corrective maintenance time, 243
 mean preventative maintenance time, 243
 mean time between maintenance, 243
 mean time to failure, 183
 mean time to remove and replace, 243
 mean time to repair, 243
 measurements, 229
 memory, errors in, 83
 metaquestions, 73
 metrics
 maintainability, 243–244
 reliability, 183
 usability, 207
 "middle men," identifying, 51
 migrating software, 298–299
 miscommunication, 83
 MNT. *See* Maintainability requirements
 modularization, 243
 motivation, 19, 20. *See also* Purpose area of interest
 moving systems, 285
 MPT (mean preventative maintenance time), 243
 MTBM (mean time between maintenance), 243
 MTTR (mean time to remove and replace), 243

MTTF (mean time to failure), 183
 MTTR (mean time to repair), 243
 multiple site installations, 229
 multiplicability (systems), 228
 mutation analysis, 268

N

names of interviewees, 78
 needs. *See* user needs and requirements
 networks. *See also* Logistics area of interest
 networking standards, 309
 in requirements framework, 20
 viewpoints on, 19
 new system features, 225
 non-corrupting system failures, 183
 nonbehavioral requirements, 100, 107. *See also* nonfunctional requirements
 nonfunctional requirements. *See also* operation requirements; revision requirements; transition requirements; specific requirement categories
 anatomy of, 92–94
 bad definitions for, 102
 classifications, 110–117
 data and, 22
 defined, iii, 96, 99–106
 defined more simply, 103–105
 descriptions and questions, 91–93
 difficulties in identifying, 106–110
 as disliked term, 105
 vs. functional, 101–103
 rephrasing, 270–272
 system requirements and, 8, 101–102
 trade-offs, 108–110
 types of, iv
 user-focused approach, 118–122
 "normal operating times," 140
 notes
 analyzing, 82
 note takers, 59, 61–62, 76
 typing, 84
 numeric displays, 229



O

observation
 observational errors, 83
 requirements source, 32
offspring qualities, 110–111
open-ended questions, 68–70
operating systems
 migrating software, 298–299
 standards, 309
operation requirements
 "bucket" metaphor, 122
 defined, 125
 question categories, 90
 as requirements category, 102–103
 types of, 126
 Access Security requirements, 127–139
 Availability requirements, 140–151
 Efficiency requirements, 152–164
 Integrity requirements, 165–180
 Reliability requirements, 181–193
 Survivability requirements, 194–205
 Usability requirements, 206–222
 user-focused approach, 118–121
operations
 in requirements frameworks, 20, 21, 22
 in software development cycles, 14, 15
opinions/opinion polls, 72, 207
organizational distance, 310
organizational requirements, 112
organizations. *See also* Roles area of interest
 in requirements frameworks, 20
 revision requirements and, 228
outages, 140

P

parent qualities, 110–111
partial system availability, 141

participants in interviews, 59
passwords, 128, 129
peer reviews, 265–266
people. *See also* Roles area of interest; stakeholders
 as sources, 31
 viewpoints on, 19
perfective maintenance, 242
performance testing, 269
Perlis, Alan, 308
permanent storage, 154
permanent system failures, 183
permission for note takers, 62
personal space, 78
perspectives. *See* viewpoints and perspectives
phone interviews, 77
plans in software development cycles, 12
"plug-and-play" features, 227
POFOD (probability of failure on demand), 183
Portability (POR) requirements, 298–306
 defined, 120, 122, 298
 examples, 300
 overview, 298–299
 suggested questions, 301–306
 terminology comparisons, 116–117
 transition requirements, 90, 122, 285–286
 user needs, 122
portability trade-offs, 109
posture and body language, 79
potential stakeholder list, 42–43
precision, data, 166
preparations for interviews, 55–65
prepared questions, 63–64, 81–84, 95
preventative maintenance, 242
preventing system failures, 182, 196
prioritizing requirements, 15
probability of failure on demand (POFOD), 183
problem counts, 207
problem reports, 32
procedure guides, 32
Process area of interest (how questions)
 defining in requirements, 4–6
 in requirements frameworks, 20, 22, 23, 92
 suggested questions
 Access Security questions, 138–139
 Availability questions, 150–151
 Efficiency questions, 163–164



Flexibility questions, 240–241
 Integrity questions, 179–180
 Interoperability questions, 296–297
 Maintainability questions, 253–254
 Portability questions, 305–306
 Reliability questions, 191–193
 Reusability questions, 315
 Scalability questions, 264
 Survivability questions, 204–205
 Usability questions, 221–222
 Verifiability questions, 283
 processing capacity, 153, 154, 255
 producers, requirements, v, 35–38
 product engineering, 288
 product-level requirements, 112, 207
 products (nonfunctional definitions), 103, 104
 program errors, 206, 245
 program portability, 299
 programming language standards, 309
 project management, 18
 project plans, 26
 properties, nonfunctional requirements and, 100, 104
 protocols, 287
 prototypes/prototyping, 15, 269
 purpose, interviews, 58, 75
 Purpose area of interest (why questions)
 defining in requirements, 4–6
 vs. "how" questions, 24–25
 in requirements frameworks, 20, 22, 23, 91
 suggested questions
 Access Security questions, 134–136
 Availability questions, 145–146
 Efficiency questions, 158–160
 Flexibility questions, 234–236
 Integrity questions, 174–176
 Interoperability questions, 292–294
 Maintainability questions, 249–250
 Portability questions, 302–303
 Reliability questions, 187–189
 Reusability questions, 313
 Scalability questions, 260–261
 Survivability questions, 201
 Usability questions, 214–217
 Verifiability questions, 279–280

Q

quality
 Boehm's quality attribute tree, 110–111
 in nonfunctional definitions, 103, 104
 nonfunctional requirements and, 100
 quality attributes. *See* nonfunctional requirements
 requirements. *See* nonfunctional requirements
 Question strategy (STAR), 80, 81
 questionnaires, 32
 questions
 asking right questions, 66–74
 closed-ended questions, 68, 69–70, 77
 context-free questions, 68, 69
 direct questions, 70
 elaboration questions, 70–71
 follow-up questions, 81–83, 95
 "leading the witness," 71
 metaquestions, 73
 open-ended, 68–70
 preparing for interviews, 63–64
 providing for interviews, 61
 STAR interviews, 81–84
 suggested elicitation questions
 Access Security requirements, 131–139
 Availability requirements, 143–151
 Efficiency requirements, 156–164
 Flexibility requirements, 231–241
 how to use, 94–95
 Integrity requirements, 168–180
 Interoperability requirements, 290–297
 Maintainability requirements, 247–254
 Portability requirements, 301–306
 Reliability questions, 185–193
 Reusability questions, 312–315
 Scalability requirements, 258–264
 Survivability requirements, 198–205
 unique identifiers, 92
 Usability requirements, 209–222
 Verifiability requirements, 276–283
 types of, 68–74
 unanswered questions, 85



R

R-Recap & Wrap-up strategy (STAR), 84–85
ranges of application (software), 310
rate of failure occurrence (ROCOF), 183
Rational Unified Process (RUP), 18
re-authentication, users, 129
recall, errors in, 83
Recap and Wrap-up strategy (STAR), 84–85
receivers. *See* requirements receivers
recognition of system failures, 195
recording interviews, 76
recoverable system failures, 183
recovery from system failures, 183, 195, 196,
 242–245
recycling hardware, 256
registering users, 127–129
regression analysis and testing, 269
REL. *See* Reliability requirements
releases, software, 141
relevance (requirements), 108
Reliability (REL) requirements, 181–193
 defined, 120, 121, 181
 examples, 184
 metrics, 183
 operational requirements, 90, 121
 overview, 181–183
 suggested questions, 185–193
 terminology comparisons, 116–117
 user needs, 121
reliability trade-offs, 109
repair guidance (systems), 243
repairing system faults, 196
rephrasing unverifiable requirements, 270–272
replies (maintenance process cycle), 245
reporting (maintenance process cycle), 244
representation stage (requirements process), x, 9, 10, 11
requirements. *See also* functional requirements; nonfunctional
 requirements; software
 requirements; *specific requirement categories*
 defects in, 206
 defined, 4, 99–105

development, defined, 7–8
elicitation. *See* elicitation
focus areas, 4–6
frameworks, 19–23, 26
management process, x, 6, 9–18
parsing, 269
producers, v, 35–38
receivers, v, 21, 22, 35–38, 40
representation. *See* representation
requirement patterns, 112–114, 115
reusing documentation, 307–308
reusing requirements, 308
suppliers, v, 21, 22, 35–38, 40
supporters, 35–38, 40
types of, 102–103
unverifiable language, 270–272
validation. *See* validation
violations of, 245
what requirements are not, 24–28
requirements elicitation. *See* elicitation
requirements frameworks, 19–23, 26
requirements management process
 benefits of, 15
 components of, 9–12
 defined, 9
 goals, 6
 software development lifecycle, 12–18
 stages in, x
requirements producers, v, 35–38
Requirements Quest logo, x, 326
requirements receivers, v, 21, 22, 35–38, 40
requirements representation. *See* representation
requirements suppliers, v, 21, 22, 35–38, 40
requirements supporters, 35–38, 40
requirements validation. *See* validation
resistance to failure, 194
respect, 74–75
response time/respondiveness, 152, 153
responses to interviewees, 78
restoring data, 166
Reusability (REU) requirements, 307–315
 defined, 120, 122, 307
 examples, 311
 overview, 307–311
 possible areas for reuse, 310–311
 suggested questions, 312–315



terminology comparisons, 116–117
 transition requirements, 90, 122, 285–286
 types of reuse, 308
 user needs, 122
 reusability trade-offs, 109
 reusable stakeholder profile repositories, 50
 reusing documentation, 307–308
 reusing requirements, 308
 Review strategy (STAR), 80–81
 reviews, 269
 revision requirements
 "bucket" metaphor, 122
 defined, 225–226
 question categories, 90
 as requirement category, 102–103
 suggested questions
 Flexibility requirements, 227–241
 Maintainability requirements, 242–254
 Scalability requirements, 255–264
 Verifiability requirements, 265–283
 user-focused approach, 118–119, 120, 121
 risks, 26, 307
 robustness requirements. *See* Survivability requirements
 ROCOF (rate of failure occurrence), 183
 Roles area of interest (who questions)
 defining in requirements, 4–6
 developing institutional lists of, 39–41
 in requirements frameworks, 19, 20, 22, 91
 stakeholder classification, 35–38
 stakeholder profiles, 41–51
 suggested questions
 Access Security questions, 132–133
 Availability questions, 144
 Efficiency questions, 157–158
 Flexibility questions, 233
 Integrity questions, 171–173
 Interoperability questions, 291–292
 Maintainability questions, 248
 Portability questions, 301
 Reliability questions, 186–187
 Usability questions, 312
 Scalability questions, 259
 Survivability questions, 199–200
 Usability questions, 211–213
 Verifiability questions, 277–278

Royce, Winston, 15
 RUP (Rational Unified Process), 18

S

S-Start with Style (STAR), 74–75
 Scalability (SCL) requirements, 255–264
 defined, 120, 121, 255
 examples, 257
 overview, 255–256
 revision requirements, 90, 121, 226
 suggested questions, 258–264
 terminology comparisons, 116–117
 user needs, 121
 scenario analysis, 33
 schedules
 excluding from requirements frameworks, 26
 interview scheduling, 56–57, 58, 82
 requirements management and, 15
 SCL. *See* Scalability requirements
 scope, 15, 20, 21, 22
 security issues, 166, 274. *See also* Access Security requirements; Integrity requirements
 seeding, 268
 services, functional requirements and, 100
 session usage, 129
 SFMECA (software failure mode, effects and critical analysis), 269
 simulations, 269
 simultaneous users, 154
 single site installations, 229
 size factors
 migrating software, 298
 reusable software, 309
 software
 installation, 274–275
 migrating or porting, 298–299
 in nonfunctional definitions, 103, 104
 requirements. *See* functional requirements; nonfunctional requirements; software requirements; specific requirement categories
 software development



- developers' views and requirements, 5
lifecycles, 12–18, 244
maintainability and, 244
releases, 141
specifications. *See* specifications
software failure mode, effects and critical analysis
 (SFMECA), 269
software fault tree analysis, 269
software inspections, 265–266, 268, 273
software requirements. *See also* functional requirements; nonfunctional requirements; specific requirement categories
 benefits of, 15
 business requirements. *See* business requirements
 defined, 4, 5
 descriptions and questions, 91–93
 in development cycles, 12, 15
 functional/nonfunctional defined, 99–106
 hierarchy of levels, 6–8
 in iterative approach, 18
 management. *See* requirements management process
 quality of requirements, 56
 rephrasing, 270–272
 requirements frameworks, 19–23
 sources for, 31–33
 specifications. *See* specifications
 system requirements. *See* system requirements
 user-focused approach, 118–122
 user requirements. *See* user needs and requirements
software specifications. *See* specifications
Sommerville, Ian, 116, 117, 183, 195–196
source code, 244, 309
specifications
 defined, 9
 materials not included in, 26
 researching older specs, 32
 structure of, 24–25
spiral model (software development), 16
sponsors' support, 51
SRV. *See* Survivability requirements
stakeholders
 analyzing expertise, 48–50
 appropriate questions for, 94
 areas of expertise, 42, 43–44
availability, 51
building rapport, 46
centralized profile repositories, 50
checklists, 40
classification of, 35–38
defined, 5, 34
developing institutional lists of, 39–41
"information hoarders," 67
interviewing. *See* interviews
"pecking order" of, 45
potential list of roles, 42–43
profiles, 41–51
requirements process and, 10
sources for requirements, 31–33
surveying, 44–47
types of, 33–41
unavailability, 44
vs. users, 5
standardization, 243
standards
 as business rules, 28
 interoperability and, 288
 maintainability and, 244
 representatives as stakeholders, 41
 requirements management process, 13
 reusable software and, 308
 types of, 309
standards organizations, 41
STAR interviews
 A-Ask Questions strategy, 81–84
 R-Recap and Wrap-up, 84–85
 S-Start with Style, 74–75
 T-TQLR strategy, 79–81
Start with Style (STAR), 74–75
storage capacity, 153, 154–155
storage management faults, 274
stress testing, 270
subjective qualities, 108
success criteria (requirements process), 95
successful interviews, 64
summarizing interviews, 85
Summerville, Ian, 112, 113
suppliers. *See* requirements suppliers
support, securing, 51
surveys, 32, 44–47, 46
Survivability (SRV) requirements, 194–205



defined, 120, 121, 194
 examples, 197
 operational requirements, 90, 121
 overview, 194–196
 suggested questions, 198–205
 terminology comparisons, 116–117
 user needs, 121

Survivable Systems Analysis, 194–195

system environments, 8, 18, 285, 298

system failures

- causes, 182
- classes of, 183
- costs, 181
- impacts, 141
- maintainability requirements, 242–245
- metrics, 183
- operation requirements, 181–183
- outages, 140, 141
- preventing, 182, 196
- survivability, 194–196

system models, 21, 22

system outages, 140, 141

system requirements

- classifying, 122
- defined, 7–8
- development of, 11
- functional/nonfunctional requirements and, 8
- illustrated, 6
- in requirements frameworks, 20, 101–102

systems

- adapting, 285
- complexity, 244
- defined, 7–8
- environments, 8, 18, 285, 298
- failures. *See* system failures
- maintenance. *See* Maintainability requirements; maintenance (system)
- models, 21, 22
- in nonfunctional definitions, 103, 104
- outages, 140, 141
- requirements. *See* system requirements
- transferring, 298
- upgrading, 141, 274

T

T-TQLR strategy (STAR), 79–81
 tailorability, 228
 taking notes, 76, 82, 84
 tape recordings, 62
 task analysis, 33
 task lists, 26
 task times, 207
 technology

- interoperability and, 288
- in requirements frameworks, 20, 21

telephone interviews, 77

temporary storage, 154

terminology

- revision requirements and, 227–228
- unverifiable, 270–272

test certification, 270

test checkpoints, 243

"testable" terminology, 270

testing

- data restoration, 166
- interface testing, 268
- interoperability, 288
- in iterative approach, 18
- in maintenance process cycle, 245
- performance testing, 269
- regression analysis and testing, 269
- in software development cycles, 14, 15
- software testing, 266
- stress testing, 270
- unverifiable language, 270–272
- Verifiability requirements, 265–283
- verification and validation techniques, 266–270

thankning interviewees, 85

threats (system), 165

throughput, 152, 153–154

time formats, 229

time parameters (maintainability), 243

Timing area of interest (when questions), 161–162

- defining in requirements, 4–6
- in requirements frameworks, 20, 22, 23, 91

suggested questions



Access Security questions, 136–137
Availability questions, 147–148
Efficiency questions, 161–162
Flexibility questions, 237–238
Integrity questions, 176–177
Interoperability questions, 295
Maintainability questions, 251–252
Portability questions, 303
Reliability questions, 189
Reusability questions, 314
Scalability questions, 262
Survivability questions, 202
Usability questions, 218–219
Verifiability questions, 281
viewpoints on, 19
timing of interviews, 63
trade-offs, 108–110, 206
traffic, 153
training, 274
transient system failures, 183
transition requirements
 "bucket" metaphor, 122
 defined, 285–286
 question categories, 90
 as requirement category, 102–103
suggested questions
 Interoperability requirements, 287–297
 Portability requirements, 298–306
 Reusability requirements, 307–315
 user-focused approach, 118–119, 120, 121–122
transition stage (software), 16–18
transparent system availability, 141
troubleshooting systems, 274
Tune in, Question, Listen, and Review strategy,
 79–81

U

unanswered questions, 85
unavailability (stakeholders), 44
unavailability (systems), 182
"unavailability windows," 140

understanding scores, 207
uninstallation, 275
unrecoverable system failures, 183
unreliable systems, 242–245
unstated expectations, 245
unverifiable terminology, 270–272
upgrading systems, 141, 274
usability errors, 245
Usability (USE) requirements, 206–222
 defined, 120, 121, 206
 examples, 208
 metrics, 207
 operational requirements, 90, 121
 overview, 206–207
 suggested questions, 209–222
 terminology comparisons, 116–117
 user needs, 121
usability trade-offs, 109
usage, monitoring, 154
USE. *See* Usability requirements
user accounts, 127–129
user-focused approach to requirements, 118–122
user-friendliness, 206
user needs and requirements
 development of requirements, 11
 illustrated, 6
 needs, defined, 4
 requirements management and, 15
 user needs in requirements, 119, 121, 122
 user requirements, defined, 7
users
 authentication, 129
 authorization, 128–129
 defined, 7
 end-users, 34
 expectations, 245
 needs. *See* user needs and requirements
 observing, 32
 one-time, 128
 perceptions of quality, 225, 285
 ported software and, 299
 questionnaires, 32
 registering/de-registering, 127–128
 requirements, 5. *See also* user needs and requirements
 simultaneous, 154
 vs. stakeholders, 5



tasks, 33
user-focused approach to requirements, 118–122

V

V & V (Verification and Validation), 265–270, 273
validating facts, 83
validation stage (requirements process), x, 9, 10, 11
variability, reusable software and, 310
Verifiability (VER) requirements, 265–283
 defined, 120, 121, 265
 examples, 275
 overview, 265–275
 revision requirements, 90, 121, 226
 suggested questions, 276–283
 terminology comparisons, 116–117
 unverifiable terminology, 270–272
 user needs, 121
 verification and validation techniques, 266–270
"verifiable" terminology, 270
Verification and Validation (V & V), 265–270, 273
verification plans, 26
videotaping, 62
viewpoints and perspectives
 elicitation questions and, 23
 in requirements frameworks, 19, 20, 21, 92
visual aids in communication, 44, 58, 77

W

walkthroughs, 270
Wallace, Dolores R., 266–270
waterfall models, 14–15, 16
"what" questions, 69, 70. *See also* Data area of interest
"when" questions, 70. *See also* Timing area of interest
"where" questions, 70. *See also* Logistics area of interest

"who" questions, 70. *See also* Roles area of interest
"why" questions, 66, 69, 70, 73. *See also* Purpose area of interest
Willis, Ronald, 116–117
Withall, Stephen, 24, 25, 101–102, 112–114
work settings, 56
workability, 152
workshops, 33
wrapping up interviews, 84–85

Z

Zachman, John, 19–23



