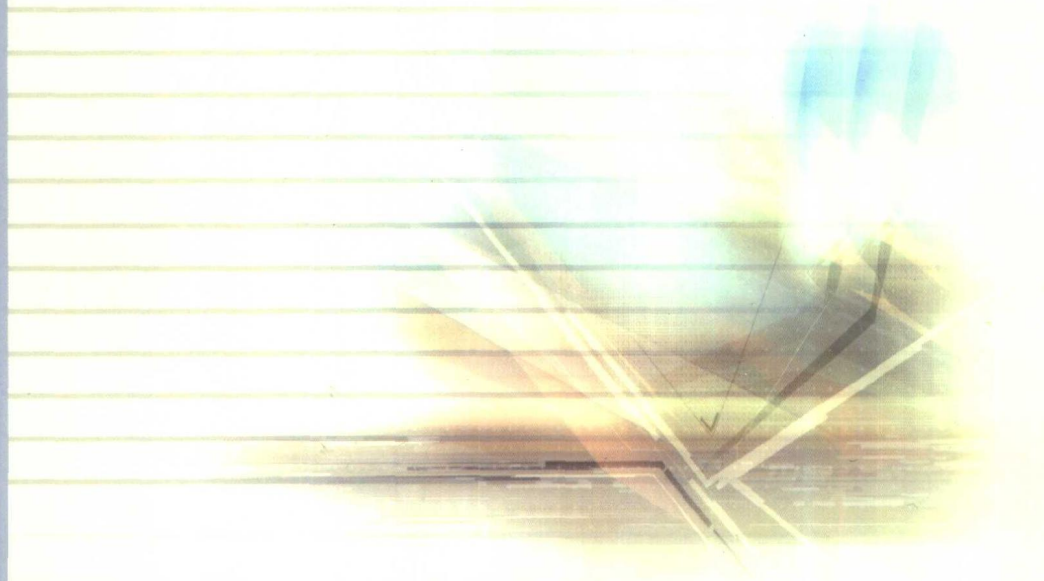
软件工程与方法丛书

影印版

有效需求分析

Effective Requirements Practices



(美) Ralph R. Young 编著

Pearson Education

科学出版社

www.sciencep.com

软件工程与方法丛书

有效需求分析

（影印版）

Effective Requirements Practices

（美） Ralph R. Young 编著

科学出版社

北京

图字：01-2003-6691 号

内 容 简 介

本书从管理和技术两个角度，以案例方式阐述了软件项目中与需求分析相关的各种问题，力图让读者能够对需求分析的框架体系和过程形成较为清晰的认识，在实践中准确了解客户的业务需求，正确调配各种资源，更加准确地把握项目的方向，保证整个项目的成功。

本书内容丰富翔实，实用性强，适合作为高等学校本科生和研究生的软件工程类教材，同时也可供软件企业对开发和项目管理人员进行培训使用。本书原版配有光盘，包括书中出现的部分插图和模板文件，读者如有需要，请与我社联系（电话：010-62622941）。

图书在版编目(CIP)数据

有效需求分析= Effective Requirements Practices /（美）拉尔夫·R·扬（Ralph R. Young）编著.

—影印版. —北京：科学出版社，2004

（软件工程与方法丛书）

ISBN 7-03-012468-5

Ⅰ.有… Ⅱ.①扬…②R… Ⅲ.软件开发—系统分析—英文 Ⅳ.TP311.52

中国版本图书馆 CIP 数据核字（2003）第 099958 号

策划编辑：李佩乾/责任编辑：袁永康

责任印制：吕春珉/封面设计：飞天创意

科学出版社 出版

北京东黄城根北街16号

邮政编码：100717

http://www.sciencep.com

双青印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2004 年 1 月第 一 版 开本：787×960 1/16

2004 年 1 月第 一次印刷 印张：24 1/4

印数：1—3 000 字数：364 000

定价：40.00 元

（如有印装质量问题，我社负责调换〈环伟〉）

影印前言

“软件工程”是自20世纪60年代起针对所谓“软件危机”而发展起来的概念。它是指将工程化的方法应用到软件开发中，以求优质高效地生产软件产品。其中综合应用了计算机科学、数学、工程学和管理科学的原理和方法。自从这一概念提出以来，软件开发方法从60年代毫无工程性可言的手工作坊式开发，过渡到70年代的结构化分析设计方法、80年代初的实体关系方法，直到当今所流行的面向对象方法，经历了根本性的变革。随着时代的发展，软件项目管理人员越来越需要从系统和战略的角度来把握项目的方向，引导开发向更高层次发展。在这方面，国外的知名企业和研究机构已经总结了相对成熟的一套知识和方法体系，并在实践中获得了相当大的成功。这里我们就从著名的培生教育出版集团（Pearson Education Group）选取了一些软件工程与方法类的有代表性的教材影印出版，以期让国内的读者尽快了解国外此领域的发展动态，分享国外专家的知识和经验。以下对每本书的内容作一些简要的介绍，以便读者选择。

在Internet广泛普及的今天，软件承载着越来越重要的使命，工程化的开发必须能够提供健壮性和普适性更好的产品。Scott E. Donaldson和Stanley G. Siegel合作编著的《成功的软件开发》（*Successful Software Development*）一书从“软件系统开发无定式”这一事实出发，引入了一个灵活而成熟的开发过程模型——系统工程环境（Systems Engineering Environment，SEE）。该模型包含两个互相联系的基本要素：确定软件开发策略和规程，以及可用于实现目标的技术。围绕这一模型，书中对开发过程中关系项目成败的各种关键问题进行了透彻的论述，可作为软件开发团队的全程培训教材。

软件工程经过几十年的发展，其关键环节——需求分析和管理终于得到了真正的重视。伴随认识的深化和案例的丰富，一系列实用的工程化需求分析方法应运而生。Ralph R. Young的《有效需求分析》（*Effective Requirements Practices*）一书从管理和技术两个角度阐述了关系项目成败的各种问题，书中的案例分析让项目管理者能够对需求分析的框架体系和过程形成较为清晰的认识，在实践中准确了解客户的业务需求，正确地调配各种资源，从而更加准确地把握项目的方向，保证在整个项目周期中各种需求都能够得到应有的考虑和满足。

软件开发人员在系统组织方面通常会采用特定的模式，但就大多数而言，他们对体系结构的分析和判断在很大程度上都是出于自发而且并不规范。Mary Shaw和David Garlan的《软件体系结构》（*Software Architecture: Perspectives on an Emerging Discipline*）一书就对当前业界所公认的成功的系统设计思想进行了生动而全面的阐述。两位作者对软件体系结构的发展状况及其对设计的影响作用有独到的见解，相信各类读者都能从书

中获得有用的信息：专业开发人员可能对书中所介绍的设计模式比较熟悉，但从对这些模式的讨论和评价中也能够获得新的启发；学生们能够从书中学到一些有用的技巧，从较高的视角来了解系统的组织结构，而不是仅仅追逐业界的潮流或是过时的方法；对于教师而言，本书可以作为“软件体系结构”课程的教材，或者是“软件工程”或“软件设计”课程的补充教材。

对于现在已经或者将要从事软件项目管理的读者来说，Joel Henry 的《软件项目管理》（*Software Project Management: A Real-World Guide to Success*）应该说是一本难得的好教材。书中论述了软件项目的四个基本构成要素：人、过程、工具和评价体系，并向读者提供每一领域中可以选用的合适方法，使项目实施取得最满意的效果。作者本人在软件行业工作多年，积累了丰富的第一手材料，他在书中用案例对技术、管理和领导等多方面问题进行了透彻的讲解。尽管他对这些问题的结论对业内人士而言已经是耳熟能详，但不论是何种类型和水平的读者，相信都能从本书中得到指导和启发。

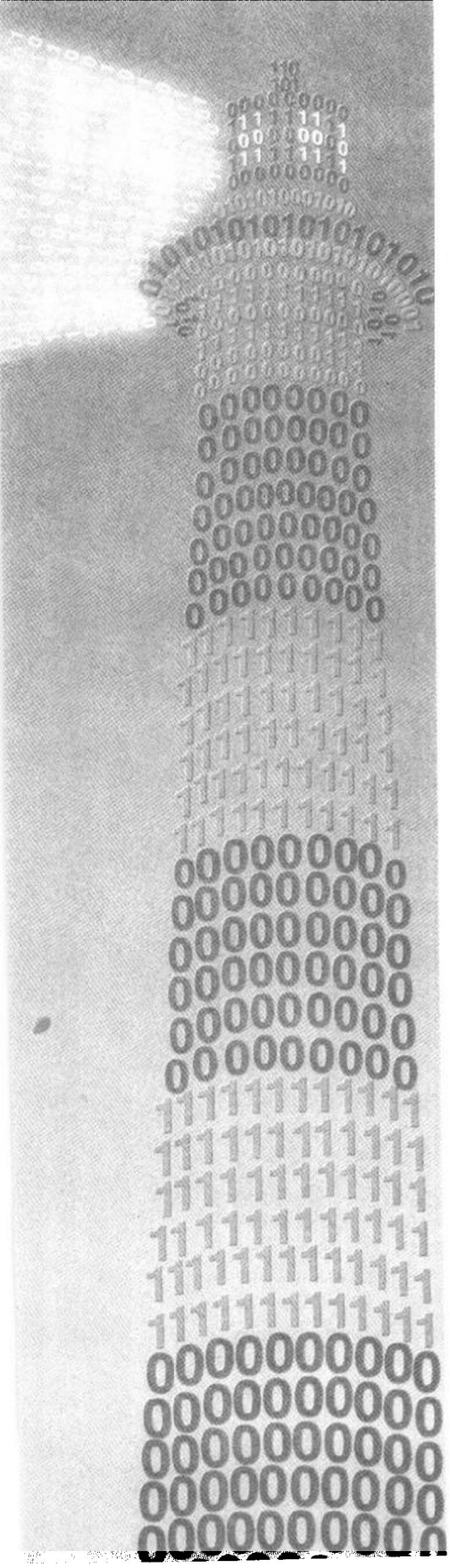
软件开发中出现错误在所难免，关键是要及早发现，而不要让其扩散，增加纠正的成本。软件同级评审制度是任何高质量软件开发过程的重要环节，然而受过这方面必要培训的专业人员却还是凤毛麟角。Karl E. Wiegers 的《软件同级评审》（*Peer Reviews in Software*）就是针对这方面需求而编写的。本书介绍了软件同级评审的全过程，内容涉及各种正规和非正规的评审方法和质量保证技巧。书中对大型项目及开发团队地域分散等情况下的同级评审进行了专门探讨。对于项目管理者而言，本书能够帮助他们以实用的资源启动同级评审计划，增进开发人员间的沟通，最终按期提交高质量的软件产品。

面向对象技术的应用已经越来越普遍，而与此同时越来越多的管理者在项目进行中却要面对许多原本隐藏着的成本和意外情况。对整个项目而言，管理者在规划阶段是否有远见、在进行过程中对各种情况反应是否得当，这些都影响着项目的成败。尽管市场上介绍对象技术的书很多，但对于项目实施中所需进行的规划和预测却还缺乏系统的知识归纳。Alistair Cockburn 的《对象软件项目求生法则》（*Surviving Object-Oriented Projects*）一书就以大量专家的知识和经验向读者提供成功管理对象软件项目的实用指导和建议，帮助读者应对项目中的意外挑战，使项目正常进行并最终获得成功。书中指出了对象软件项目所面临的潜在风险，并对时间安排、预算、人员安排以及成本合理化等重要问题进行了探讨，提供了可操作的解决方案。对于从事对象软件项目管理的读者而言，本书是一本相当合适的参考书，可以作为相关培训的教材。

以上就是目前这套影印版丛书的大致内容。需要指出的是，这套丛书并非一个封闭的体系。随着软件工程的深入发展，必将涌现出更多新的开发理念和方法，我们也将尽己所能将更多新的优秀图书吸纳进来。读者对于这套丛书目前的内容或未来的发展有何意见或建议，望不吝赐之。

编　者

2003 年 11 月



Foreword

It's your worst nightmare. A customer walks into your office, sits down, looks you straight in the eye, and says, "I know you think you understand what I said, but what you don't understand is what I said is not what I mean." Invariably, this happens late in the project, after deadline commitments have been made, reputations are on the line, and serious money is at stake.

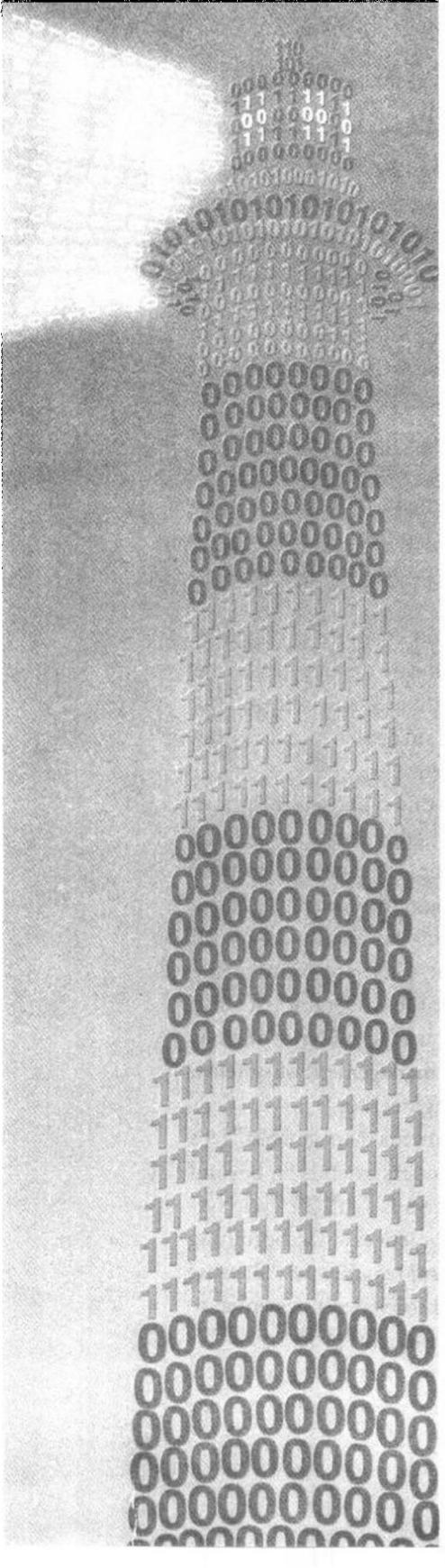
All of us who have worked in the systems and software business for more than a few years have lived this nightmare, and yet, few of us have learned to make it go away. We struggle when we try to elicit requirements from our customers. We have trouble understanding the information that we do acquire. We often record requirements in a disorganized manner, and we spend far too little time verifying what we do record. We allow change to control us, rather than establishing mechanisms to control change. In short, we fail to establish a solid foundation for the system or software that we intend to build. Each of these problems is challenging. When they are combined, the outlook is daunting for even the most experienced managers and practitioners. But solutions do exist.

In this book, Ralph Young presents a comprehensive requirements process that identifies the key steps necessary to gather, understand, record, and verify the things that customers require when they request systems and software solutions. His treatment of the subject addresses the three critical elements that make a process work: (1) the tasks required to

implement the process, (2) practical guidance required to implement the tasks, and (3) a strategy that helps to improve the process continually.

An experienced industry practitioner and manager, Dr. Young recognizes that effective requirements practices demand effective people practices. He provides pragmatic guidelines for creating requirements teams that meld the skills and knowledge of both customers and contractors. He recommends practical methods for eliciting requirements and at the same time understanding real customer needs. He proposes step-by-step suggestions for maintaining communication between the customer and all engineering groups as the project proceeds. He suggests indispensable techniques for verifying that the requirements we do record accurately reflect the customer's needs. He discusses the impact of change on requirements and offers practical steps for managing change. In short, Ralph Young has written an excellent guide for those who must understand and manage their customers' requirements. And this means just about everyone in the systems and software world.

Roger S. Pressman, Ph.D.
Consultant and Author
President, R.S. Pressman & Associates, Inc.



Preface

Dealing effectively with requirements tops the list of the challenges to managers and practitioners developing systems and software. Improving the effectiveness of requirements practices has been a focus for me throughout my career. My vision of this book is to help you in your life's work by providing practical, useful, effective requirements practices.

This book describes ten requirements practices that provide a framework for overcoming current industry problems. Although systems and software development efforts have been going on for five decades, the industry has major difficulty worldwide in delivering products that meet customer needs. By applying effective requirements practices, one can remove causes of project failure. The reasons for failure are well documented (see Chapter 1). The needed improvement activities can be financed via the one third of total project costs now wasted. This book is full of suggestions concerning how to transform this waste into productive use.

The theme of this book is that practitioners should insist on using effective requirements practices. The use of effective requirements practices will reduce costs, improve the quality of work products, and increase customer satisfaction. The practices, ideas, suggestions, and recommendations provided in this book can be used individually or collectively, and not all have to be implemented to achieve

progress. One can gradually implement some good practices quickly, with good payback, and then continue to work toward a more sophisticated, high-performance set of requirements practices.

This book provides a baseline for managers and project leaders to use to ensure that they are doing what is necessary to make a project successful. The practices, methods, techniques, and the requirements process itself have been filtered through experience, so the ideas are practical, cost-effective, and proven.

This book deals with the practical difficulties of requirements elicitation and management from a pragmatic, organizational, and project perspective. Attention is given to the pitfalls, costs, and risks as well as to the benefits of these practices. Unfortunately, many good practices never get implemented because the benefits are oversold, and the costs and risks aren't recognized. Political realities of organizations and projects must also be considered.

Application of the practices in this book will result in more productive, healthier, and happier organizations for systems and software development. The analysis extends beyond the technical issues to human issues and values. This book emphasizes the need for a shared vision of project success and advises how to obtain the required customer and supplier commitment.

The effective requirements practices described in this book will help you whether you are in a small organization or a large one, whether you build systems or software. Advice is provided for the information technology executive, consultant, manager, architect, systems or software engineer, systems integrator, developer, tester, process improvement engineer, member of the quality assurance group, or one responsible for configuration management. This book is invaluable for systems and software engineering courses, at both the undergraduate and graduate levels, and also for venues relating practical, useful guidance such as industry association and corporate courses concerning management of systems, requirements, as well as systems and software process engineering.

Although one frequently sees references to "requirements management," let's be clear that the challenge to system and software developers extends far beyond simply managing requirements. The requirements process is a full life cycle systems engineering process. It requires special effort and practices at the beginning of a project or system to identify what I refer to as *the real requirements.* Because the world changes while we are developing systems and software, it's essential to address new and changed requirements within the requirements process.

The requirements process requires mechanisms, for example, to achieve a shared vision, to ensure joint customer and supplier responsibility for the

requirements, and to enable effective project coordination. The requirements process impacts every other activity performed in developing systems and software. One needs an automated requirements tool that provides for attributes such as the priority of each requirement, how it is linked to the design, where it is met in the code, how it is verified and tested, and so forth. It should be apparent already that we as an industry do not spend enough time and effort on the requirements process, and that this itself is a root cause of our problems.

The requirements comprise the basis for all the development work that follows. If you don't get the requirements right, you are in for a long, hard, and expensive pull. Your chances of "finishing" are small, and the probability of satisfying the users of the planned system is nil. We know from industry experience that customers don't know their "real requirements" (even though they may have spent a lot of time defining them and believe they know them). Suppliers and system developers don't know them either. Identifying the real requirements requires an *interactive* requirements process, supported by effective mechanisms, methods, techniques, and tools. The requirements process need not be complicated or expensive. However, a requirements process is *required* for a project of any size. It's more important that a project or organization *have* a requirements process than the nature of its specific components.

This leads to another fundamental premise of this book: Continuous improvement and a quality ethic within a project or organization lead to repeatable processes and reuse that save time and money. My commitment to these values comes from my work experiences and also from my study under Dr. W. Edwards Deming. Dr. Deming clarified for me that many of the root causes of problems are not technical issues. Rather, the root causes concern our responsibility as organizational and project executives, managers, and leaders to provide the environment in which "the workers" can be effective, productive, and fulfilled. Management must empower its work force to unleash its incredible capabilities.

The systems and software development environment needs attention, as we all can attest, based on our experience. The effective practices advocated in this book will facilitate your creation of the needed environment and will empower and enable your development team. I have witnessed (as I hope you have too) the power and the results of effective teams in positive environments. My experience is that an empowered team can accomplish anything it sets out to do. We must work to create the needed environment for success.

To benefit from the information in this book, you need bring only your involvement in systems and software-related activities coupled with a desire to

improve. If you are a customer or client of the system or software provider industry, you will be particularly interested in Chapters 1 through 5 and 12. If you are a practitioner already familiar with the issues and problems, you may proceed directly to whichever of Chapters 2 through 11 relate most closely to your specific work activities, noting the references to additional information and sources. If you are an executive or manager, you may want to focus on Chapters 1, 7, and 12 to gain added insight into the issues, to garner a high-level understanding, and to formulate some ideas concerning candidate improvement actions. If you are a student of systems or software engineering, you'll likely find it worth your time to proceed deliberately through the book. If you are participating in a requirements-related course, you'll find the entire book insightful and provocative.

A rich collection of suggested references is provided in the footnotes, the Key References and Suggested Readings sections provided for each chapter, and the bibliography. No source is included just because it provides related information. Rather, each and every one is noted because it provides additional insights, more detailed suggestions and ideas, a recommended technique/approach, or an alternative concept you might want to consider.

Primary Features of This Book

- Provides a practical organizational and project perspective.
- Emphasizes the need for a partnership approach and explains how to obtain commitment.
- Focuses on specific improvement activities.
- Explains how to evolve the real requirements.
- Considers the human dimension.
- Emphasizes the importance of effective communication.
- Provides sample templates (for example, for a requirements process and a requirements plan).
- Discusses the need to iterate the requirements and the architecture.
- Recommends how to deal with changing requirements.
- Explains requirements verification and validation.
- Suggests several mechanisms to facilitate self-correction and to help maintain momentum.
- Stresses the need for an automated requirements tool.
- Provides advice and recommendations for executives, project managers, and leaders.

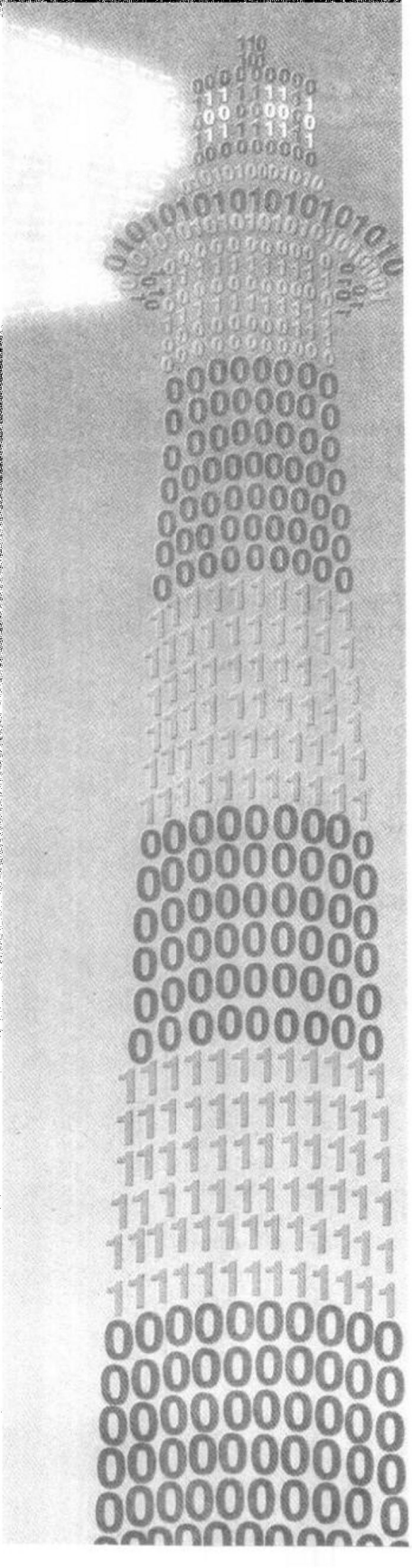
- Enables organizations and projects to utilize their resources better.
- Includes a rich set of references.

I am very grateful to a large number of reviewers for the material presented in this book. They have been helping me for 28 years to understand what works. Some I've come to know only recently, and many of them are industry experts in requirements, systems, or software engineering. The publisher tasked some industry experts to review these materials, and their review comments were invaluable. Others provided informal reviews because they are experts in particular areas or because they are professionally interested. Addison-Wesley's publishing professionals have made an invaluable contribution to the final product.

All of the reviewers have reinforced something I already knew: These practices are urgently needed today on projects and efforts of all sizes in all systems and software efforts. My hope is that they will help you. Of course, different practices work well in different environments. This is something we all understand from our experience, no matter where that experience was acquired or what fields it concerned. So you will need to select appropriate practices, recommendations, and suggestions, and apply them with a large measure of common sense—always a great guide!

I hope that you take the time and effort to share with me your experiences in applying the practices, recommendations, and suggestions in this book. This will help me further strengthen and improve my own insights and understandings and, God willing, allow me to share them again with others. Please write to me at *ryoungrr@aol.com.*

Ralph Young
February 2001



Acknowledgments

Every author knows that writing a book is a team effort. Families sacrifice time together. Publishing professionals work miracles. Friends and associates are asked to review work, make comments, validate ideas from their experience, perhaps even write something. Industry experts are contacted and asked to confirm their views, suggest additional references, and provide insights. Librarians search diligently for needed references.

Thank you to my wife, Judy, for her incredible patience and understanding. Thank you to Addison-Wesley (A-W), particularly to my editor, Debbie Lafferty, and also to Kristin Erickson whose enthusiasm, support, and encouragement led me to write this book. Patrick Peterson of the A-W production department made this part of the effort easy, Catherine Ohala did a magnificent job as copy editor, and Steve Katigback created extremely useable indexes. Friends and associates who lent a hand (and mind) include Stephen Bachanan, Cora Carmody, Pete Carroll, Barb Dreon, Jim Faust, Jim Fowler, Bob Fox, Steven Gaffney, Tom Gilb, Sharon Guenterberg, Jack Hayes, Alice Hill-Murray, Craig Hollenbach, Ivy Hooks, Earl Hoovler, Ray Huber, Barbara Kopp, Dan Marchegiani, Charles Markert, Andy Meadow, Hal Miller, John Moore, Matt Noah, Mark Paulk, Christina Pringle, Rich Raphael, John ("Mike") Reeves, Olga Rosario, Bette Rutherford, Dora Schield, Doug Smith, John Waters, Penny Waugh, Beth Werner, Doug Whall, and Don Young.

Additional industry experts who helped include Dennis Beude, Jeff Grady, Rita Hadden, Watts Humphrey, Capers Jones, Dean Leffingwell, Steve McConnell, Fergus O'Connell, Roger Pressman, Pete Sawyer, Jerry Weinberg, Neal Whitten, Karl Wiegers, Ed Yourdon, and Richard Zultner.

Thanks are also due to many individuals, organizations, and companies that provided permission to reuse their materials, including Al Pflugrad and Litton PRC, Addison-Wesley, Rational Corporation, the Software Engineering Institute, Charles Markert, Litton Applied Technology, Compliance Automation, the Institute of Electrical and Electronics Engineers, Litton-TASC, Telelogic, John Wiley & Sons, The Open Group, Microsoft Press, McGraw-Hill, Software Productivity Research, International Thomson Computer Press, The Neal Whitten Group, AMACOM (publishing arm of The American Management Association), American Programmer, Prentice-Hall, Prentice-Hall Europe, ETP Inc., Dorset House, and the International Council on Systems Engineering.

My writings and insights have been strengthened by the reviewers, including Len Bass, Gary Chastek, Sholom Cohen, Jack Hayes, Ivy Hooks, John Moore, Joseph Morin, Mark Paulk, Daniel Rawsthorne, Rob Sabourin, Karl Weigers, Bill Wiley, and Neil Williams.

Prayer works. Thanks to Art Banks, Tom Foss, Craig Hollenbach, and Joe Matney. Family support buoys the spirit. Thanks to Kim Wallace, Jeff Young, and Matt Young.

Errors and omissions are my responsibility. God willing, I'll have opportunities to fix them.

Contents