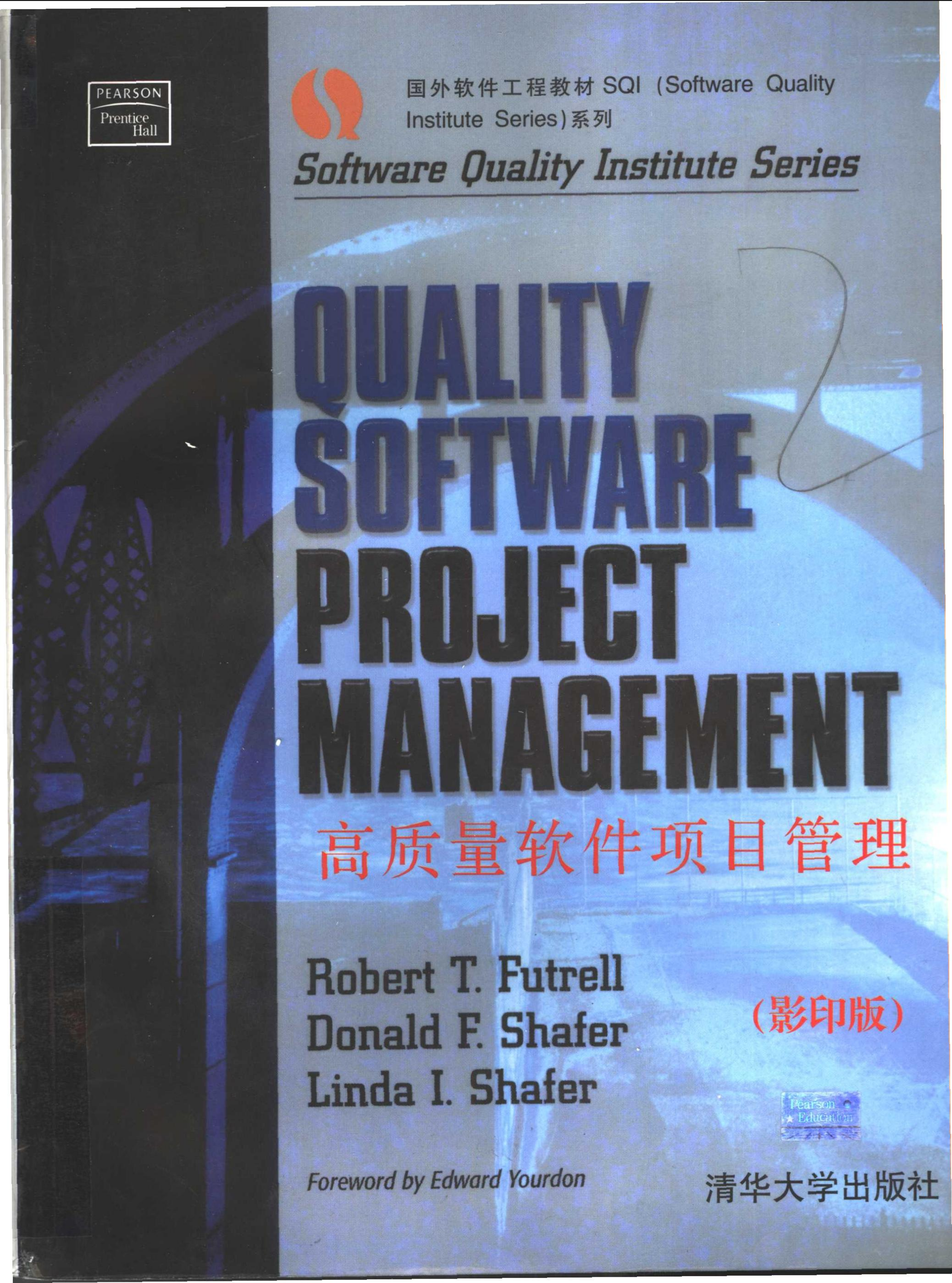
PEARSON
Prentice Hall

国外软件工程教材 SQI (Software Quality Institute Series)系列

Software Quality Institute Series



2

QUALITY SOFTWARE PROJECT MANAGEMENT

高质量软件项目管理

Robert T. Futrell
Donald F. Shafer
Linda I. Shafer

(影印版)

Pearson Education

Foreword by Edward Yourdon

清华大学出版社

Pearson Education 培生教育出版集团

国外软件工程教材 SQI（Software Quality Institute Series）系列

Quality Software Project Management
高质量软件项目管理

（影印版）

Robert T.Futrell
[美] Donald F. Shafer 著
Linda I. Shafer

清华大学出版社
北京

ISBN 0-13-091297-2
Quality Software Project Management
Robert T. Futrell
Donald F. Shafer
Linda I. Shafer

图书在版编目（CIP）数据

高质量软件项目管理=Quality Software Project Management/（美）富特雷尔等著. —影印本. —北京：清华大学出版社，2003.6
国外软件工程教材
ISBN 7-302-06523-3

Ⅰ. 高…　Ⅱ. 富…　Ⅲ. 软件开发—项目管理—教材—英文　Ⅳ. TP311.52

中国版本图书馆 CIP 数据核字（2003）第 026190 号

出 版 者：清华大学出版社（北京清华大学学研大厦，邮编：100084）
http://www.tup.tsinghua.edu.cn
http://www.tup.com.cn
责任编辑：丁岭
印 刷 者：清华大学印刷厂
发 行 者：新华书店总店北京发行所
开　　本：787×1092 1/16　　**印张**：104.5
版　　次：2003 年 6 月第 1 版　2003 年 6 月第 1 次印刷
书　　号：ISBN 7-302-06523-3/TP・4893
印　　数：0001～3000
定　　价：128.00 元

Software Quality Institute Series

The Software Quality Institute Series is a partnership between the Software Quality Institute (SQI) at The University of Texas at Austin and Prentice Hall Professional Technical Reference (PHPTR). The books discuss real-life problems and offer strategies for improving software quality and software business practices.

Each publication is written by highly skilled, experienced practitioners who understand and can help solve the problems facing software professionals. SQI series topic areas include software development practices and technologies, management of software organizations, integration of high-quality software into other industries, business issues with reference to software quality, and related areas of interest

Foreword

A few years ago, a colleague at a management consulting firm invited me to give a presentation at a monthly meeting of the local chapter of a professional software organization. It turned out that there was an ulterior motive for the invitation: My colleague explained that his firm was involved in a huge project for a major client, and that several of the client's managers would be attending my presentation.

"Here's the problem," my colleague said to me. "We've got dozens of our programmers, analysts, network architects, database designers, and other technical people working on this project—and the client is perfectly happy to pay for them. But when we told them that we need to have a project manager and some support staff to help carry out the project management tasks, they balked. They don't understand why they should have to pay for project management—and the way they described it to us, it sounds like they don't believe that project management has any value." My task for the presentation, as it turned out, was to provide an eloquent explanation of why project management was important, with the indirect implication that it was worth paying for.

If such an event had taken place in the mid-1960s, perhaps it would not have been surprising. After all, as Futrell, Shafer, and Shafer point out in the first chapter of their book, it was not until 1968 that a famous NATO software engineering conference provided some public recognition of the importance of project management in what came to be known as the "software crisis." Even in 1975 or 1985, we could have forgiven a typical business person for not appreciating that successful IT projects require more than an army of clever technical people. But my experience took place in the 1990s, and I suspect that it is being repeated today, in various parts of the world. If nothing else, it demonstrates why there is such a desperate need for a thorough, detailed book like *Quality Software Project Management.*

The illusion that no project management resources are necessary to succeed with an IT project is only slightly more dangerous than the common misconception that project management is simple, intuitive, and easily learned by skimming through a "project management for dummies" book. A quick scan of the Amazon.com Web site indicates that there are roughly half a dozen books with some variation on that title; and while the books probably do serve a constructive purpose, I'm concerned about the common perception that a 22-year-old Java programmer, with a mere two years of experience in a technical discipline, can be promoted to the position of project manager with any reasonable hope of succeeding on a non-trivial project.

Becoming a *bona fide* project manager is not a quick or easy process—and if I can accomplish only one thing in this brief foreword, let me also emphasize that it's not equivalent to achieving competence with a software product like Microsoft Project. That particular program, as well as a dozen others like it, are enormously useful tools for carrying out some of the scheduling activities associated with a project. But as the authors describe in enormous detail in this book, there's more to project management than just drawing PERT charts and Gantt charts. Indeed, there are some 34 key competencies, as the authors point out; perhaps we can get away with mediocrity or minimal competence in one or two of those competencies, if the circumstances of the project allow it, but there are literally dozens of things we need to be good at if we're going to call ourselves "project managers" in the highly complex field of IT systems development.

The authors have been involved with a software project management certification program at the University of Texas at Austin's Software Quality Institute; in the best of worlds, IT organizations would send their fledgling project managers to such a program for a total immersion course—as well as sending their veteran project managers (most of whom have acquired only a haphazard understanding of the 34 key competencies through on-the-job training) for a refresher course. But for those of us who don't have the time, or whose employers don't have the budget or the foresight to send us to such a program, the next best thing is a book like *Quality Software Project Management.*

Chances are that you won't be able to read this book in a single sitting. It's not an "airplane book" that you can read on a quick flight from New York to Chicago; it's not even a "weekend book" that you can take to the beach for some summertime reading. You'll need to set aside an hour or two each evening over a period of several weeks or months to be able to absorb all of the guidelines, checklists, procedures, and advice from these eminently qualified practitioners of software project management. You should also take advantage of the Web site and additional resources provided by the authors and realize that mastery of project management is an ongoing process.

When I first started working in the computer field in the mid-1960s, my goal was to be the best assembly-language programmer on the planet. Of course, the computer I was working on at the time has long since disappeared, but there is still a great deal of honor and virtue to be associated with mastery of such technical skills as programming, testing, or database design. For many of us, though, a fascination with technical skills is eventually replaced by a dedication to project management skills because it doesn't take more than two or three projects to realize that success or failure is far more likely to be determined by management issues than technical issues. In my case, it took nearly a decade to make that shift in preferences and priorities. Simply wanting to become a project manager is not enough. I only wish that I had had a book like *Quality Software Project Management* to provide the foundation for skills and practices that I had to learn on my own, piece by piece. As for you, dear reader, rejoice: You do have such a book, and if you read it and study it carefully, it will not only speed up the learning process, but it may also help you avoid some unpleasant project disasters along the way!

Ed Yourdon

September, 2001

Preface

Quality Software Project Management was written by and for software practitioners who need a hands-on guide to the non-deterministic but leading-edge task of managing software development projects. The book takes its overall outline from the successful Software Project Management (SWPM) certification program at The University of Texas at Austin's Software Quality Institute, a division of the College of Engineering's Center for Lifelong Engineering Education (CLEE).

Software project managers and their development teams play a critical role in the success of modern businesses, be they high-tech or otherwise. These professionals and their knowledge of sound management practices and thorough software development, enhancement, and maintenance processes, can determine organizational success or failure.

The trend toward increased software quality is responsible for the promulgation of new standards to certify that development processes meet certain benchmarks. Certifications to standards are becoming more common as buyers demand tighter quality controls. Software project managers must be keenly aware of standards such as those published by the Institute of Electrical and Electronics Engineers (IEEE), as well as continually evolving practices, guided in part by the Software Engineering Institute's (SEI) Capability Maturity Model (CMM), and by a new emphasis on the management of small projects.

It is in recognition of these trends that UT's College of Engineering and its Software Quality Institute (SQI) created the SWPM certificate program in 1993. Since then, hundreds of software project managers have graduated from the program. Those managers are currently applying "best practices" to overcome the limitations of a tight labor force and to meet the rapidly changing needs of their customers and organizations in today's highly competitive marketplace. This book is a consolidation of teachings from that certification program as it has evolved over the years.

In addition to knowledge of the principles of software engineering, software project managers must incorporate skills for managing people, products, and process into their daily routine. For this reason, *Quality Software Project Management* is grounded in two interlaced bodies of knowledge developed by internationally recognized organizations: the Project Management Institute (PMI®) and the American Society for Quality (ASQ). SQI instructors, many of whom are certified software (CSQE) and project management professionals (PMP®), refine knowledge identified by those two organizations and contribute decades of their own industry experience with the most up-to-date practices. Quality, applicability, timeliness, portability, and profitability are all main areas of focus, both for the SWPM certificate program and for this book, on which it is based.

Software engineering principles and quality goals are necessary but not sufficient for the needs of today's marketplace. Shorter cycle times, completed with fewer resources, are also in demand. Products must be carefully targeted toward the specific functional requirements of increasingly sophisticated customers. Software developers and managers dealing with these challenging and often conflicting goals, must be highly skilled in planning, coordinating, and managing software projects. They must know how to tailor best practices to their current projects and to take advantage of their organization's past experience when constructing project plans. Establishing the proper metrics to monitor project performance is essential, as is having necessary multi-disciplinary team leadership skills. Furthermore, software project management must view the project "big picture" as it relates to their profession and to their career advancement.

Quality Software Project Management has evolved from the strong belief of the authors, and based on their experience, that with a defined process, quality software can be developed in a repeatable fashion. Figure 1 shows that methods, tools, and technology interrelate in complex and constant ways and require the process in order to achieve balance. These three entities are at the heart of quality, software, and project management, and will therefore be used throughout the text. A *method* is defined as a manner, means, or process for accomplishing something. A *tool* is defined as an implement or machine used to do work or perform a task. *Technology* is defined as the application of scientific knowledge in industry or business.

The experience of the authors is that the knowledge in this guide, applied by practitioners, along with the effective use of methods, tools, and techniques encapsulated in 34 competencies, will result in quality software. "Quality" incorporates the necessary functionality as well as other factors such as reliability, usability, etc. Figure 2 represents how ideas are turned into products through iterations of such use.

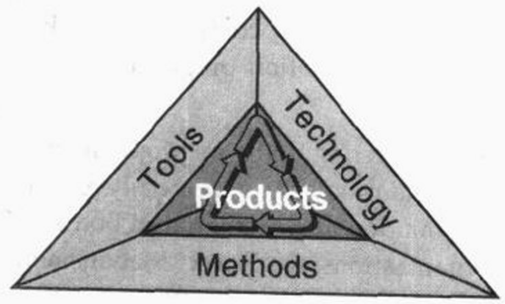


FIGURE 1
Methods, Tools, and Technology Relationships

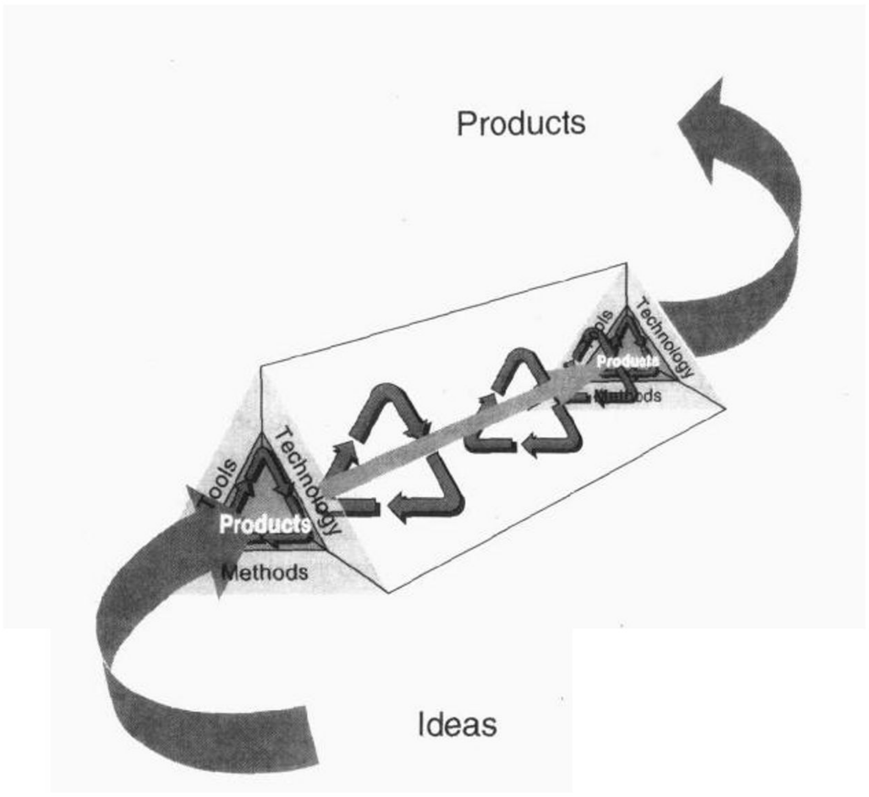


FIGURE 2
Transforming Ideas into Products

While based on the SWPM course materials, this book is not simply a recitation of them. The specific combined experience of these authors (almost 100 years worth) permeates the work and attempts to blend the thoughts of about 30 instructors into "one voice." A composite case study has been developed containing most of the common types of problems encountered in software projects. The project scenario reflects today's increasingly common need for rapid "Internet time" software development.

Using the Guide as a Course Text

If you are participating in either the online or the classroom presentation of The University of Texas at Austin Software Quality Institute's Software Project Management certification program, this will be your main text. If you are a professor or instructor of software engineering, this text will suffice for a semester-long course in software engineering plus project management. The bodies of knowledge for project management, software engineering, and software quality, recognized by several professional societies (IEEE, SEI, PMI, ASQ) are presented. If you are a student of project management and software engineering, please feel confident that real industry veterans have authored this text.

Acknowledgements

The authors of this text are "masters" not "philosophers," meaning that each has a Masters Degree, but none has a PhD at this time. In the original academic world of associates, bachelors, and masters, the masters of the trade knew how to apply their knowledge to "real" tasks. They were the practitioners. Philosophers were considered to be in a different category, focusing on theory and more ethereal concepts. Our current academic society assumes that philosophers are also masters of application. While there is no question that these philosophers deservedly receive the highest recognition, it also seems to be the case that, with the fields of software engineering and project management, the masters of application are often found in industry rather than in academia. Computer science is to software engineering as chemistry is to chemical engineering. The former is about the theory, and the latter is about practical application of the theory. A mathematician who wrestles with the theoretical question of whether an answer exists, has a different job than the engineer who needs to know the answer in order to use it. While paying homage to all of the theorists who have developed computer science, we hope to be some of the masters, who, in some small way, add to its application.

We wish to personally thank The University of Texas at Austin, Center for Lifelong Engineering Education, Software Quality Institute's staff for their unwavering, cheerful, consistent (and constant) help. They made the last eleven software project management certification program materials available to us, and professionally, efficiently, and effectively helped us get everything we needed. Candy Walser-Berry, Marilyn Robertson, Theresa Lestingi, Heather Wagner, Jayne Tune, Carolyn Stark—thanks! The Chinese railway case study became a real but fair student challenge due to the original work of Jack Odom and the acting skills of John McNeill. The employee owners of Athens Group provided material for Appendix B, "Real World Projects," and a wealth of metrics data. The instructors who shaped the SWPM lessons deserve special credit—many of them are cited in the reference sections of the individual chapters. We appreciate the SQI Board of Advisors who have volunteered their time, since 1993, to make a program of high quality. Thank you Paul Petralia and Jennifer Blackwell at Prentice Hall, and especially to Barry Busler of IBM. And, of course, we also appreciate and thank our children for cheering us on—a collection of four fabulous young women and one incredible young man.

Contents