Preface to the Second Edition

Richard H. Thayer
Merlin Dorfman

Much has changed in the field of requirements engineering since the first edition of this tutorial was published 6 years ago. Some of the more important changes since 1990 are as follows:

- The increased recognition of the importance of requirements engineering and the risks of doing it incorrectly or insufficiently (the inability to produce complete, correct, and unambiguous software requirements) is still considered the major cause of software failure today
- An increase in the number of conferences and workshops devoted exclusively to requirements engineering, for example, the International Symposium on Requirements Engineering, held in odd-numbered years, and the International Conference on Requirements Engineering, held in even-numbered years
- An increased number of books and journals devoted to requirements engineering, for example, Requirements Engineering Journal (Springer-Verlag London, Ltd.), Davis, A.M., Software Requirements: Analysis and Specification (Prentice-Hall International, Englewood Cliffs, N.J.)
- The inclusion, in the Capability Maturity Model for Software (published in 1993 by the Software Engineering Institute), of requirements engineering as a “Key Process Area” in defining a mature software development process
- The increased development of commercially available tools that support requirements engineering functions
- The emergence of subspecialties of requirements engineering, for example, requirements elicitation, requirements verification, and requirements traceability
- More extensive use of the concept of operations (ConOps) document, which was first invented and described in the literature more than 15 years ago
- More extensive practice of the validation and verification of requirements. As with the ConOps document, this is not a new idea, but is now much more accepted

This tutorial describes the current state of the practice of requirements engineering, primarily for software systems but also for systems that may contain other elements such as hardware and people. Every attempt has been made to select good, current tutorial papers for this tutorial. A paper published in the 1980s or even the 1970s is not by definition “old” or out of date. Some very fine papers, published 10–15 years ago, are still current and are the best available description of the subject under consideration. Papers selected for inclusion in this tutorial were carefully checked against the following criteria, to determine if they:

- Accurately described the state of the practice for the given topic; if a small portion of a paper was perhaps outdated, this was explained in the chapter introduction
- Covered the state of the practice for the given topic thoroughly and evenly
- Defined the basic terms
- Avoided presenting new, unproved concepts, with no reasonable prediction as to their future
- Did not try to sell one tool or concept over all others
- Are easy to read, for example, contained no gratuitous (unnecessary) mathematics
- Are organized in a hierarchical manner (top-level concepts discussed first, second-level concepts discussed next, and so forth)
- Provided a list of additional references
- Were written by an expert in the area (to assure all of the above)

Our criteria are, of course, idealized. Even these “rules” were violated if there was a good reason. On the basis of these criteria, we have selected papers in the following six chapters:

Chapter 1: Introductions, Issues, and Terminology
Where there were good existing papers, either from the first edition of this tutorial or elsewhere, we have used them. A large number of the papers from the first edition are still current and topical. We have also sought out new papers and revisions of papers from the first edition from noted authors in the field.

This tutorial is one of a set of tutorials on “software system engineering” published or to be published by the IEEE Computer Society Press:


Duplication of papers across tutorials has been kept to a minimum. In a few cases, particularly important papers are duplicated in order that each tutorial can stand alone.

We would like to acknowledge the support provided by the following people:

- Dr. William (Bill) Sanders, Managing Editor, Computer Society Press
- Ms. Cheryl Smith, Acquisitions Assistant, Computer Society Press
- Ms. Lisa O’Conner, Tutorial Production Editor, Computer Society Press

Merlin Dorfman, PhD
Lockheed Martin Missiles & Space Company
Sunnyvale, California

Richard Thayer, PhD
California State University, Sacramento
Sacramento, California
Preface

The purpose of this tutorial is to assemble under one cover a comprehensive body of knowledge about systems and software requirements engineering. Emphasis is on software requirements analysis and specifications, system engineering and its interface with software engineering, as well as information on subjects that are affected by system and software requirements, such as verification and validation, management, and configuration management. An additional volume, Standards, Guidelines, and Examples: System and Software Requirements Engineering, edited by Merlin Dorfman and Richard H. Thayer and published by the IEEE Computer Society Press, contains material that may be considered as an appendix to this tutorial.

This tutorial contains 44 original papers and reprints on system and software requirements engineering, a glossary of over 1400 system/software engineering terms applicable to requirements engineering, an "Annotated Bibliography" of books and reports with over 50 entries, and a list of "Selected References and Other Readings" with over 300 entries.

This tutorial is intended for:

- **Managers:** The tutorial provides an overview of system and software requirements issues and describes many of the "popular" software requirement methodologies and tools. Included are descriptions of software requirements specifications and methodologies, descriptions of several "new" software process models and their relationships to requirements engineering, discussions on software requirements reviews and verification, and approaches to managing the requirements activities. Several case studies are provided.

- **System and software system engineers:** The tutorial describes the front end of the software lifecycle and the difficulties interfacing with the system specifications. It also presents a basic understanding of system and software system engineering: partitioning, allocation, and flowdown of system requirements to software; software requirements analysis; and specification writing.

- **Software engineers, programmers, analysts, and other computer personnel:** The tutorial contains a general description of procedures and techniques for analyzing and specifying system and software requirements, and for writing a good requirements specifications, as well as descriptions of a number of state-of-the-art methodologies, representation methods, tools, and techniques.

- **Hardware engineers:** The tutorial describes how system specifications are partitioned and allocated to software, the techniques and tools for analyzing and describing software specifications, and how software interfaces with hardware.

- **College-level students and their teachers:** The tutorial offers sufficient background and instructional material to serve as a main or supplementary text for a course in system/software requirements analysis and specifications, in other words, requirements engineering.

This tutorial takes the view that system and software requirements engineering is the technical "center" of any system and/or software development project. Papers were selected that either describe this technical center or describe activities that interact with it (e.g., project management, configuration management, and reviews and walkthroughs). This approach provided a framework for selecting appropriate papers and assembling original material. This tutorial also takes the view that the techniques, tools, and procedures used for system engineering are similar, if not identical, to those used for software engineering.

Both original and reprinted papers are organized into chapters according to their impact on system and software requirements. Chapter 1 starts with a paper that introduces system and software engineering. Other papers in the chapter define and describe software engineering and the effect requirements have on the engineering process. Terminology and issues are also discussed.

Chapter 2 contains papers describing system engineering, operational concepts, methods for deriving software requirements from system requirements, and the application of system engineering principles to software called software system engineering.
Chapter 3 contains papers on “classic” software requirements analysis and the five types of requirements—functional, performance, external interfaces, design constraints, and quality metrics—as well as an approach to representing a software requirements specification by a users' manual. Chapter 3 also includes the most popular standard in software engineering, the IEEE Guide to Software Requirements Specifications.

Chapter 4 contains papers on ways to analyze and document software requirements. The papers deal with the classic (non-realtime) and modern (realtime) approaches to structured analysis, data-structured approaches, object-oriented methods, a method based on interface analysis, and formal (mathematically based) approaches to requirements engineering.

Chapter 5 contains papers on automated approaches to software requirements analysis (called requirements tools or computer-aided software engineering (CASE) tools). Classic tools such as SREM (Software Requirements Engineering Methodology) are described as well as innovative, experimental tools such as the knowledge-based system called Requirement Apprentice. Requirements analysis and documentation tools for personal computers are also presented, as well as some general purpose tools, such as Fourth Generation Languages (4GLs) and automated traceability tools.

Chapter 6 is concerned with managing the software requirements engineering activity and with testing and verification of system and software requirements. Papers describe a management view of requirements; the verification, validation, walkthrough/inspection, and review of requirements; and the part configuration management plays in the requirements phase.

Chapter 7 begins with three papers that provide different perspectives on the overall strategy of software development. It continues with detailed discussion of two new software development paradigms—reuse and prototyping—and their impact on software requirements.

Chapter 8 is devoted to case studies. Four case studies highlight the application of information hiding, realtime structured analysis, prototyping, and formal specification methods.

All papers, whether original or reprinted, are tutorial in nature and are intended to explain some aspects of system and/or software requirements engineering. Every attempt was made to use the latest information. However, recent papers that present the latest (as yet unproven) technology were, as a rule, excluded. The papers in this tutorial explain the best state-of-the-practice in specifying and documenting system and software requirements.

In order to fill gaps in the published literature, the authors of this tutorial contacted experienced, seasoned authors and practitioners in the areas of system and software engineering and asked them to write or revise papers for the tutorial. Each of these original papers has been independently refereed through the IEEE Computer Society peer review process. The following authors, listed alphabetically, have contributed original or revised papers to this tutorial:

- Dr. Peter P. Chen (Chen & Associates) describes the Entity-Relationship approach to data modeling that he pioneered.
- Mr. Peter Coad (Object International, Inc.) and Mr. Edward Yourdon have written about a new, but rapidly growing, approach, Object-Oriented Analysis, and its advantages over conventional methods.
- Dr. Alan M. Davis (George Mason University), chair of the IEEE working group for a “Guide for Software Requirements Specifications,” has written a paper on software requirements analysis and specifications that is based on the IEEE standard.
- Dr. Merlin Dorfman (Lockheed Missiles & Space Company, Inc.) has written an introductory paper on system and software requirements and the application of allocation, partitioning, flowdown, and traceability to system engineering.
- Mr. Richard F. Flynn (Lockheed Missiles & Space Company, Inc.) and Dr. Merlin Dorfman have updated their 1984 paper, “ARTS: An Automated Requirements Traceability System,” to reflect recent improvements in ARTS and to relate their experiences in supporting the system.
- Dr. Hassan Gomaa (George Mason University) has written a paper that emphasizes the difference between rapid and evolutionary prototyping.
- Mr. Charles P. Hollocker (Northern Telecom, Inc.), chair of the IEEE working group for a “Standard for Software Reviews and Audits,” has prepared a paper on requirements reviews, walkthroughs, and inspections.
• Dr. Norman R. Howes (Institute for Defense Analyses) has rewritten his paper on the procedure for substituting a user manual for a requirements specification.

• Mr. Steven E. Keller and Mr. Laurence G. Kahn (Dynamics Research Corporation) and Mr. Roger B. Panara (Rome Air Development Center) have written a paper on quality metrics for software requirements. This paper incorporates the R&D work done at RADC on Software Quality Attributes. There were no existing papers on software quality metrics as applied to software requirements.

• Mr. Robert J. Lano (Lano Enterprises) has written a paper on the operations concept document. The operations concept document is one of the most valuable tools in system engineering today and is required on most government contracts. There is no other existing public paper on the subject.

• Dr. John H. Manley (University of Pittsburgh) has contributed a paper on a new software development lifecycle strategy. Manley's two-track lifecycle model separates and shows the interrelationships between the management lifecycle and the technical lifecycle. The impact of system and software requirements on this model is highlighted.

• Mr. E. Dale Nelsen (Lockheed Missiles & Space Company, Inc.) has contributed an original paper on requirements allocation and flowdown, entitled "System Engineering and Requirement Allocation." One of the most important issues in system engineering today is how to properly partition, allocate, flowdown, and trace system requirements to software implementation.

• Dr. Arthur Pyster (Software Productivity Consortium) has prepared a paper on a new software development strategy called the Synthesis Process. This new process acknowledges and uses iteration, prototyping, and incremental development. It is specifically intended to promote the reuse of software.

• Mr. J. Douglas Sailor (Lockheed Missiles & Space Company, Inc.) has written a paper on system engineering patterned after the Defense Systems Management College textbook, Systems Engineering Management Guide. Sailor was the author of the first edition of the text. There were no existing papers that provided a view of system engineering to the necessary breadth and depth.

• Dr. Hasan Sayani (Advanced Systems Technology Corporation) has written a history and current status report on the Problem Statement Language/Problem Statement Analyzer (PSL/PSA).

• Dr. Cyril P. Svoboda (Advanced Systems Technology Corporation) has written on realtime and non-realtime structured analysis, consolidating the research and implementation activities in structured analysis over the past 13 years.

• Dr. Richard H. Thayer (California State University, Sacramento) and Dr. Winston W. Royce (SoftwareFirst) have written a paper on software system engineering that outlines and describes the system engineering approach to developing software.

• Dr. Joseph E. Urban (Arizona State University) has contributed a new paper on formal requirements analysis methods and executable specifications.

• Dr. Raymond T. Yeh (Syscorp International) and Dr. Peter A. Ng (New Jersey Institute of Technology) have contributed a general description of software requirements analysis and specifications from a management perspective. Dr. Yeh also wrote the foreword to the tutorial.

• Dr. Pamela Zave (AT&T Bell Laboratories) has modified her paper on "Assessments," which shows a taxonomy of the various software requirements analysis methods and techniques.

No successful endeavor has ever been carried through by one person alone. This is one of the measures of management, and this tutorial is no exception. The authors would like to thank the people and organizations that supported us in this effort, including:

• Ms. Margaret Brown, Managing Editor, IEEE Computer Society Press, for removing much of the drudgery in writing a book.

• Dr. Garry Kampen and his editors, for their review and comments.

• Ms. Barbara Dietrich of California State University, Sacramento, California, for her part in writing the glossary.

• Mr. Joseph Kramer of the library of California State University at Sacramento, for his research help.
• California State University at Sacramento, Computer Science Graduate Class in *Advanced Software Requirements Analysis* (Fall 1988), for reviewing all the papers in this tutorial and providing their frank opinions.

• Mr. Jon Digerness of Carmichael, California, for providing the illustrations that accompany the introductions to each chapter.

Richard H. Thayer  
California State University, Sacramento

Merlin Dorfman  
Lockheed Missiles & Space Company, Inc.