

IEEE Standard for Developing Software Life Cycle Processes

Sponsor

Software Engineering Standards Subcommittee of the
Technical Committee on Software Engineering of the
IEEE Computer Society

Approved September 26, 1991

IEEE Standards Board

Abstract: The set of activities that constitute the processes that are mandatory for the development and maintenance of software, whether stand-alone or part of a system, is set forth. The management and support processes that continue throughout the entire life cycle, as well as all aspects of the software life cycle from concept exploration through retirement, are covered. Associated input and output information is also provided. Utilization of the processes and their component activities maximizes the benefits to the user when the use of this standard is initiated early in the software life cycle. This standard requires definition of a user's software life cycle and shows its mapping into typical software life cycles; it is not intended to define or imply a software life cycle of its own.

Keywords: project management process, project monitoring and control process, software development process, software implementation process, software installation process, software life cycle, software life cycle model process, software life cycle process, software maintenance process, software operation and support process, software post-development process, software pre-development process, software quality management process, software requirements process, software retirement process, software system allocation process

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

Copyright ©1992 by the
Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1992
Printed in the United States of America

Library of Congress Cataloging in Publication
information will be found on the following page.

*No part of this publication may be reproduced in any form,
in an electronic retrieval system or otherwise,
without the prior written permission of the publisher.*

IEEE Standards documents are developed within the Technical Committees of the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE which have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old, and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason IEEE and the members of its technical committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08555-1331
USA

IEEE Standards documents are adopted by the Institute of Electrical and Electronics Engineers without regard to whether their adoption may involve patents on articles, materials, or processes. Such adoption does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the standards documents.

Library of Congress Cataloging in Publication Data

Institute of Electrical and Electronics Engineers, Inc., the.

IEEE standard for developing software life cycle processes / sponsor, Software Engineering Standards Subcommittee of the Technical Committee on Software Engineering of the IEEE Computer Society.

p. cm.
"Approved September 26, 1991, IEEE Standards Board."
"IEEE Std 1074-1991."

Includes index.

ISBN 1-55937-170-6

1. Computer software—Development—Standards—United States. 2. Software maintenance—Standards—United States. I. IEEE Computer Society. Software Engineering Standards Subcommittee. II. IEEE Standards Board. III. Title
QA76.76.D47I545 1992

005.1'021873—dc20

91-41510
CIP

Foreword

(This Foreword is not a part of IEEE Std 1074-1991, IEEE Standard for Developing Software Life Cycle Processes.)

This Foreword is intended to provide the reader with some background into the rationale used to develop this standard. This information is being provided to aid in the understanding and usage of the standard. The Foreword is nonbinding.

Purpose

This is a standard for the Processes of software development and maintenance. This standard requires definition of a user's software life cycle and shows mapping into typical software life cycles, but it is not intended to define or imply a software life cycle of its own. This standard applies to the management and support Processes that continue throughout the entire life cycle, as well as all aspects of the software life cycle from concept exploration through retirement. Utilization of these Processes, and their component Activities, maximizes the benefits to the user when the use of this standard is initiated early in the software life cycle.

Software that has proceeded past the initialization phase when this standard is invoked should gradually comply with the standard.

This standard was written for any organization responsible for managing and developing software. It will be useful to project managers, software developers, quality assurance organizations, purchasers, users, and maintainers. Since it was written to consider both software and its operating environment, it can be used where software is the total system or where software is embedded in a larger system.

Terminology

The words *shall*, *must*, and the imperative form identify the mandatory (essential) material within this standard. The words *should* and *may* identify optional (conditional) material. As with other IEEE Software Engineering Standards, the terminology in this document is based on IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology (ANSI). To avoid inconsistency when the Glossary is revised, the definitions are not repeated in this document. New terms and modified definitions are included, however.

History

Work on this standard began in August 1984. A total of 15 meetings produced the draft submitted for ballot in January 1990. Two additional meetings were held to resolve comments and negative ballots, and the document was resubmitted for recirculation in February 1991.

Participants

This standard was developed by a working group consisting of the following members who attended two or more meetings, provided text, or submitted comments on more than two drafts of the standard:

David J. Schultz
Chair

John W. Horch
Vice Chair

Dennis E. Nickle
Secretary

Jean A. Gilmore
Group Leader

Art Godin
Group Leader

Lynn D. Ihlenfeldt
Group Leader

Robert W. Shillato
Group Leader

Joseph J. Guidos,* David W. Burnett
Configuration Managers

Michael Buckley
Editor

Joe Albano
Tom Antczak
Susan Burgess
David Burrows
Dan Chang
Paul Christensen
Raymond Day
Marvin Doran
Mike Ellwood
Arden Forrey
John Graham
Daniel Gray
Rob Harker

Carolyn Harrison
Peter Harvey
Eric Hensel
Denise Holmes
George Jackelen
Linell Jones
Laurel Kaleda
Phil Keys
Tom Kurihara
Bill Mar
Darrell Marsh
Leroy May

Denis Meredith
Manijeh Moghis
Richard Morton
Gerry Neidhart
Brian Nejme
Hans Schaefer
Isaac Shadman
Kelley Stalder
David Taylor
Leonard Tripp
George Tucker
Odo Wang
Richard Werling

*Deceased

Contributors

The following individuals also contributed to the development of the standard by attending one meeting or providing comments on one or two drafts:

Scott Allen
Kathleen Alley
Robert Baris
Dwight Bellinger
H. Ronald Berlack
William Blum
Robert Both
Fletcher Buckley
John Chihorek
Geoff Crellin
M. A. Daniels
Geoffrey Darnton
Leonard DeBaets
Ingrid deBuda
Kristin Dittmann
Gary Driver
Susan East
Leo Egan
Violet Foldes
Roger Fujii
Michael Garrard
Yair Gershkovitch
Ole Golubjatnikov
Jim Harkins
James Heil
Cheng Hu

Jim Hughes
Suzana Hutz
Ron Hysom
Phyllis Illyefalvi
Corbin Ingram
Ramon Izbinsky
Tom Jepson
Jia Yaoliang
Allen Jin
David Johnson
Richard Karcich
E. Klamm
Rick Kuhn
Stephan Lacasse
F. C. Lim
Ben Livson
Theresa Mack
Karen Mackey
Stan Magee
John Marciniak
Richard McClellan
Neal Mengel
Rocco Novak
George O'Connell
John Patchen

Jeff Pattee
Virgil Polinski
P. A. Rhodes
Bill Romstadt
Benson Scheff
Richard Schmidt
David Schwartz
Carl Seddio
Paul Sevcik
Randy Shipley
Kimberly Steele
Karen Steelman
Jim Stoner
Wayne Sue
Ann Sullivan
Daniel Teichroew
Russell Theisen
Donna Thomas
George Tice
R. Van Tilburg
Graham Tritt
Dolores Wallace
Valerie Winkler
Grady Wright
Fred Yonda
Lin Zucconi

Balloting Committee

The following persons were on the balloting committee that approved this document for submission to the IEEE Standards Board:

Joseph Albano, Jr. Eleanor Antreassian Au Man Kit Emanuel R. Baker David Barber H. Jack Barnard Dwight Bellinger M. Ben-Menachem H. Ronald Berlack Michael Blackledge William J. Boll Ronald Braun Bruce Brocka Winsor A. Brown Fletcher Buckley Susan Burgess David Burnett David Burrows Kay Bydalek Clyde Camp James L. Clartwright John Chihorek S. V. Chiyarath Won L. Chung François Coallier Christopher Cooke Michael A. Daniels Geoffrey Darnton Taz Daughtrey Bostjan K. Derganc Audrey Dorofee Carl E. Dragstedt Mary L. Eads Vera D. Edelstein L. G. Egan Caroline L. Evans John W. Fendrich A. L. Fleetwood Roger G. Fordham Joel J. Forman Julian Forster	Kirby Fortenberry Thomas J. Fouser Clairmont Fraser Roger Fujii Casey K. Fung Yair Gershkovitch Jean Gilmore Shirley A. Gloss-Soler Arthur Godin Richard Goldsmith Ole Golubjatnikov Steven Grady J. Kaye Grau Thomas A. Grobicki Victor M. Guarnera Cynthia Hardenbrook Lina Hardenburg R. T. Harley George B. Hawthorne Clark Hay William E. Hefley Manfred Hein John W. Horch Richard Horner Pei Hsia James R. Hughes Lynn Ihlenfeldt S. C. Jain Debra Jasinski Shang-Sheng Jeng J. O. Jenkins Jia Yaoliang Todd Jochim David Johnson Donna C. Johnson Laurel V. Kaleda Eiichi Kaneko Myron S. Karasik Peter Klopfenstein Shaye Koenig Richard Kuhn	Joan Kundig Tom Kurihara Renee Lamb John B. Lane Robert A. C. Lane J. Dennis Lawrence Cheryl Leventhal F. C. Lim Bertil Lindberg Carlo Lipizzi William M. Lively Ben Livson Dieter Look Harold Mains Kartik C. Majumdar John H. Manley Philip C. Marriott Roger J. Martin Tomoo Matsubara Ann Miller Edward Miller Manijeh Moghis Richard Morton Gene T. Morun Hironobu Nagano Saied Najafi Gerry Neidhart Dennis E. Nickle James O'Day Tuncer I. Oren Martin F. Owens Thomas Parrish Betty Paul William Perry Donald J. Pfeiffer John G. Phippen John Pope Salim Ramji Jean-Claude Rault Meir Razy Larry K. Reed	Donald J. Reifer James R. Roberts R. Waldo Roth Rafael San Roman Julio Sanz Stephen R. Schach Hans Schaefer Benson H. Scheff Max J. Schindler Norman Schneidewind Wolf A. Schnoege David J. Schultz Gregory D. Schumacher Leonard Seagren Carl S. Seddio Tony Sgarlatti Isaac Shadman Lawrence H. Shafer Robert Shillato David M. Siefert Harry M. Sneed A. R. Sorkowitz Kelley Stalder David Szombatfaluy William Theiford Russell E. Theisen Glendon R. Trebble Leonard Tripp Mark-Rene Uchida Margaret Updike Spyrpos Villios Thomas E. Vollman Dolores R. Wallace John W. Walz Andrew H. Weigel Richard Werling Paul Wolfgang Paul R. Work Natalie C. Yopconka Peter F. Zoll
---	--	---	---

When the IEEE Standards Board approved this standard on September 26, 1991, it had the following membership:

Marco W. Migliaro, *Chairman*

Donald C. Loughry, *Vice Chairman*

Andrew G. Salem, *Secretary*

Dennis Bodson
Paul L. Borrill
Clyde Camp
James M. Daly
Donald C. Fleckenstein
Jay Forster*
David F. Franklin
Ingrid Fromm

Thomas L. Hannan
Donald N. Heirman
Kenneth D. Hendrix
John W. Horch
Ben C. Johnson
Ivor N. Knight
Joseph L. Koepfinger*
Irving Kolodny
Michael A. Lawler

John E. May, Jr.
Lawrence V. McCall
Donald T. Michael*
Stig L. Nilsson
John L. Rankine
Ronald H. Reimer
Gary S. Robinson
Terrance R. Whittemore

*Member Emeritus

Kristin M. Dittmann
IEEE Standards Project Editor

Acknowledgments

Participants in the working group were individually supported by their employers with travel expenses and working days. This support does not constitute or imply approval or endorsement of this standard. These organizations were:

Abbott Critical Care	Lockheed Aircraft Service Co.
ARINC, Inc.	Martin Marietta
Apollo Computer Inc.	MUMPS Development Committee
AT&T Bell Laboratories	NCR Corp.
AT&T Technologies	Naval Air Development Center
Bell Canada	Northern Telecom Canada Ltd.
Bellcore	Northrop Electronics Div.
Bell Northern Research, Inc.	Northrop Aircraft Div.
Boeing Computer Services	Perkin-Elmer
Burnett Associates	Quality Assurance Institute
Computer Sciences Corp.	Singer Link
Digital Switch Corporation	Tandem Telecommunications Systems, Inc.
E. I. Dupont de Nemours & Co.	Tektronix
E-Systems, Inc.	Teledyne Brown Engineering
Eastman Kodak Co.	Teledyne Controls
Hewlett Packard	Texas Instruments
Honeywell, Inc.	The Horch Company
IBM	U. S. Air Force
Institute for Defense Analyses	U. S. Dept. of Transportation
Jet Propulsion Laboratory	Unisys
Litton Aero Products	3M

The following organizations hosted working group meetings in their respective cities:

AT&T Bell Laboratories, Columbus, OH	Martin Marietta, Orlando, FL
Bellcore, Piscataway, NJ	NCR Corp, San Diego, CA
Boeing Computer Services, Seattle, WA	Northrop Electronics, Hawthorne, CA
Computer Sciences Corp., Silver Spring, MD	Northern Telecom, Ottawa, Ont.
E-Systems, Inc., Salt Lake City, UT	Tektronix, Beaverton, OR
Eastman Kodak Company, Rochester, NY	Teledyne Brown Engineering, Huntsville, AL
Hewlett Packard, San Jose, CA	Travelers Companies, Hartford, CT
Honeywell, Inc., Phoenix, AZ	Unisys, Eagan, MN
IEEE Computer Society, Dallas, TX	

AT&T Bell Laboratories provided word processing support. Bellcore and Computer Sciences Corporation assisted with mailings. The X3K1 Logical Flow Project provided technical coordination.

Suggestions for the improvement of this standard will be welcome. They should be sent to the Secretary, IEEE Standards Board, Institute of Electrical and Electronics Engineers, P. O. Box 1331, Piscataway, NJ 08855-1331.

Contents

SECTION	PAGE
1. Introduction.....	11
1.1 Scope.....	11
1.2 References.....	11
1.3 Definitions and Acronyms.....	11
1.3.1 Definitions.....	11
1.3.2 Acronyms.....	13
1.4 Organization of This Document	13
1.5 Use of This Standard	14
1.5.1 Applicability.....	14
1.5.2 Compliance.....	14
1.5.3 Intended Audience.....	15
1.5.4 Process and Activity Relationships	15
1.5.5 Getting Started	18
1.5.6 Additional Considerations	18
2. Software Life Cycle Model Process.....	19
2.1 Overview.....	19
2.2 Activities List	19
2.3 Identify Candidate Software Life Cycle Models.....	21
2.3.1 Input Information.....	21
2.3.2 Description.....	21
2.3.3 Output Information	21
2.4 Select Project Model	21
2.4.1 Input Information.....	21
2.4.2 Description.....	21
2.4.3 Output Information	21
3. Project Management Processes	22
3.1 Project Initiation Process	22
3.1.1 Overview.....	22
3.1.2 Activities List	22
3.1.3 Map Activities to Software Life Cycle Model.....	22
3.1.4 Allocate Project Resources.....	23
3.1.5 Establish Project Environment	24
3.1.6 Plan Project Management.....	25
3.2 Project Monitoring and Control Process.....	27
3.2.1 Overview.....	27
3.2.2 Activities List	27
3.2.3 Analyze Risks	27
3.2.4 Perform Contingency Planning.....	28
3.2.5 Manage the Project	29
3.2.6 Retain Records.....	30
3.2.7 Implement Problem Reporting Method.....	31
3.3 Software Quality Management Process.....	33
3.3.1 Overview.....	33
3.3.2 Activities List	33
3.3.3 Plan Software Quality Management.....	33
3.3.4 Define Metrics	34
3.3.5 Manage Software Quality.....	35
3.3.6 Identify Quality Improvement Needs	35

SECTION	PAGE
4. Pre-Development Processes	37
4.1 Concept Exploration Process	37
4.1.1 Overview	37
4.1.2 Activities List	37
4.1.3 Identify Ideas or Needs	37
4.1.4 Formulate Potential Approaches	38
4.1.5 Conduct Feasibility Studies	39
4.1.6 Plan System Transition (If Applicable)	39
4.1.7 Refine and Finalize the Idea or Need	40
4.2 System Allocation Process	42
4.2.1 Overview	42
4.2.2 Activities List	42
4.2.3 Analyze Functions	42
4.2.4 Develop System Architecture	43
4.2.5 Decompose System Requirements	43
5. Development Processes	45
5.1 Requirements Process	45
5.1.1 Overview	45
5.1.2 Activities List	45
5.1.3 Define and Develop Software Requirements	45
5.1.4 Define Interface Requirements	46
5.1.5 Prioritize and Integrate Software Requirements	47
5.2 Design Process	48
5.2.1 Overview	48
5.2.2 Activities List	48
5.2.3 Perform Architectural Design	48
5.2.4 Design Data Base (If Applicable)	49
5.2.5 Design Interfaces	50
5.2.6 Select or Develop Algorithms	50
5.2.7 Perform Detailed Design	51
5.3 Implementation Process	52
5.3.1 Overview	52
5.3.2 Activities List	52
5.3.3 Create Test Data	52
5.3.4 Create Source	53
5.3.5 Generate Object Code	53
5.3.6 Create Operating Documentation	54
5.3.7 Plan Integration	55
5.3.8 Perform Integration	55
6. Post-Development Processes	57
6.1 Installation Process	57
6.1.1 Overview	57
6.1.2 Activities List	57
6.1.3 Plan Installation	57
6.1.4 Distribute Software	58
6.1.5 Install Software	59
6.1.6 Accept Software in Operational Environment	59

SECTION	PAGE
6.2 Operation and Support Process	60
6.2.1 Overview.....	60
6.2.2 Activities List	60
6.2.3 Operate the System	60
6.2.4 Provide Technical Assistance and Consulting.....	61
6.2.5 Maintain Support Request Log	61
6.3 Maintenance Process	62
6.3.1 Overview.....	62
6.3.2 Activities List	62
6.3.3 Reapply Software Life Cycle.....	62
6.4 Retirement Process	63
6.4.1 Overview.....	63
6.4.2 Activities List	63
6.4.3 Notify User	63
6.4.4 Conduct Parallel Operations (If Applicable)	64
6.4.5 Retire System.....	64
7. Integral Processes	66
7.1 Verification and Validation Process.....	66
7.1.1 Overview.....	66
7.1.2 Activities List	66
7.1.3 Plan Verification and Validation.....	66
7.1.4 Execute Verification and Validation Tasks	67
7.1.5 Collect and Analyze Metric Data	68
7.1.6 Plan Testing	69
7.1.7 Develop Test Requirements	70
7.1.8 Execute the Tests	71
7.2 Software Configuration Management Process.....	72
7.2.1 Overview.....	72
7.2.2 Activities List	72
7.2.3 Plan Configuration Management.....	72
7.2.4 Develop Configuration Identification.....	73
7.2.5 Perform Configuration Control.....	73
7.2.6 Perform Status Accounting.....	74
7.3 Documentation Development Process.....	75
7.3.1 Overview.....	75
7.3.2 Activities List	75
7.3.3 Plan Documentation	75
7.3.4 Implement Documentation	76
7.3.5 Produce and Distribute Documentation.....	77
7.4 Training Process.....	78
7.4.1 Overview.....	78
7.4.2 Activities List	78
7.4.3 Plan the Training Program.....	79
7.4.4 Develop Training Materials	80
7.4.5 Validate the Training Program	81
7.4.6 Implement the Training Program	81
8. Bibliography.....	83

FIGURES	PAGE
Fig 1. Example of Invoked Processes	17
Fig 2. Information Flow	18
Fig 3. Software Life Cycle Relationships	20
TABLES	
Table 1. Standard Organization	13
Table 2. Cross Reference of IEEE Standards	84
APPENDIXES	
Appendix A. Mapping Software Life Cycle Processes to Various Examples of Software Life Cycles	86
Appendix B. Software Project Management Tailoring Template	99
Appendix C. Process Interrelationships	105
APPENDIX TABLES	
Table A1. Software Life Cycle Example Based on Eight Phases	87
Table A2. Software Life Cycle Example Based on Five Phases	90
Table A3. Software Life Cycle Example Based on Prototyping	93
Table A4. Software Life Cycle Example Based on an Operational Specification	96
Table B1. Software Project Management Tailoring Template	100
APPENDIX FIGURE	
Fig C1. Process Interrelationships	106
Index	107

IEEE Standard for Developing Software Life Cycle Processes

1. Introduction

1.1 Scope. This standard provides the set of Activities that constitute the Processes that are mandatory for the development and maintenance of software, whether stand-alone or part of a system. (Non-software Activities, such as hardware development and purchasing, are outside of the scope of this standard.) This standard also provides associated Input and Output Information.

For convenience, Activities are listed and described under specific Processes. In practice, the Activities may be performed by persons whose organizational titles or job descriptions do not clearly convey that a Process is part of their job. The Process under which an Activity is listed in this standard may be transparent in practice.

This standard does not prescribe a specific software life cycle model (SLCM). Each using organization must map the activities specified in the standard into its own software life cycle (SLC). If an organization has not yet defined an SLC, it will be necessary for them to select or define one before attempting to follow this standard. Further, this standard does not presume the use of any specific software development methodology nor the creation of specific documents.

For software already developed, it is recommended that these requirements, or a subset thereof, be applied. The existence of this standard should not be construed to prohibit the imposition of additional or more stringent requirements where the need exists, e.g., critical software.

Compliance with this standard is defined in 1.5.1.

1.2 References. No other publications are required for use of this standard. However, a list of other IEEE standards, which may be consulted for additional guidance, is given in the Bibliography in Section 8. Although this standard does not require adherence to any other IEEE standard, knowledge of principles and concepts described in the standards listed in the Bibliography would be helpful.

1.3 Definitions and Acronyms

1.3.1 Definitions. The definitions listed here establish meanings within the context of this standard. Definitions of other terms used in this document can be found in IEEE Std 610.12-1990 [1].¹

Activity. A constituent task of a Process. *See: task.*

analysis. Examination for the purpose of understanding.

anomaly. Any deviation from requirements, expected or desired behavior, or performance of the software.

contractual requirements. Customer-imposed performance, logistics, and other requirements and commitments governing the scope of software development, delivery, or support.

customer. The person, or persons, who pay for the product and usually (but not necessarily) decide the requirements. In the context of this document the customer and the supplier may be members of the same organization. [5]

data base. A collection of data fundamental to a system. [22]

¹The numbers in brackets correspond to those of the bibliographic references listed in Section 8.

evaluation. Determination of fitness for use.

external. An Input Information source or Output Information destination that is outside the control of this standard, and therefore may or may not exist.

function. A specific purpose of an entity or its characteristic action. [22]

installation. The period of time in the software life cycle during which a software product is integrated into its operational environment and tested in this environment to ensure that it performs as required. [1]

Mapping. Establishing a chronological relationship of the Activities in this standard according to a selected SLCM.

methodology. A body of methods, rules, and postulates employed by a discipline.

owner. A single point of contact, identified by organization position.

problem. The inability of a system or component to perform its required functions within specified performance requirements. [1]

Process. A function that must be performed in the software life cycle. A Process is composed of Activities.

product. Any output of the software development Activities; e.g., document, code, model.

quality management. That aspect of the overall management function that determines and implements the quality policy. (ISO 9000)

quality policy. The overall quality intentions and direction of an organization as regards quality, as formally expressed by top management. (ISO 9000)

revision. A controlled item with the same functional capabilities as the original plus changes, error resolution, or enhancements.

software life cycle (SLC). A project-specific, sequenced mapping of Activities.

software life cycle model (SLCM). The skeleton framework selected by each using organization on which to map the Activities of this standard to produce the software life cycle.

software quality management. That aspect of the overall software management function that determines and implements the software quality policy.

software quality policy. The overall quality intentions and direction of an organization as regards software quality, as expressed by top management.

software system. Software that is the subject of a single software project.

supplier. The person, or persons, who produce a product for a customer. In the context of this document, the customer and the supplier may be members of the same organization. [5]

task. The smallest unit of work subject to management accountability. A task is a well-defined work assignment for one or more project members. Related tasks are usually grouped to form Activities. [17]

unit. A logically separable part of a program. [1]

user. The person, or persons, who operate or interact directly with the system. [5]

1.3.2 Acronyms. The following acronyms appear within the text of this standard:

CASE	Computer-Aided Software Engineering
I/O	Input/Output
PR&RP	Problem Report and Resolution Planned Information
SCMP	Software Configuration Management Planned Information
SDD	Software Design Description
SLC	Software Life Cycle
SLCM	Software Life Cycle Model
SPMP	Software Project Management Planned Information
SQA	Software Quality Assurance
SRS	Software Requirements
SVVP	Software Verification and Validation Planned Information

1.4 Organization of This Document. The organization of this standard provides a logical approach to the development, operation, and maintenance of software. The detailed requirements of this document are organized into 17 Processes, which are comprised of a total of 65 Activities. The Processes and their Activities are described in six major sections. Table 1 depicts this organization.

Table 1
Organization of the Standard

Section	Title	Processes
2	Software Life Cycle Model Process	Software Life Cycle Model
3	Project Management Processes	Project Initiation Project Monitoring and Control Software Quality Management
4	Pre-Development Processes	Concept Exploration System Allocation
5	Development Processes	Requirements Design Implementation
6	Post-Development Processes	Installation Operation and Support Maintenance Retirement
7	Integral Processes	Verification and Validation Software Configuration Management Documentation Development Training

All of a Process' required actions are specified in its constituent Activities. Each Activity discussion has three parts:

- (1) Input Information, which lists information that is to be used by the Activity, and its source.
- (2) Description, which details the actions to be performed.
- (3) Output Information, which lists the information that is generated by the Activity, and its destination.

Where information flows between Activities, it can be traced from its original Activity to the receiving Activity through the Input and Output Information tables.

As mentioned above, all Processes are mandatory. Activities, however, are categorized as mandatory² or "If Applicable."³ "If Applicable" Activities are marked "If Applicable" in the Activity title. All other Activities are mandatory. Each "If Applicable" Activity contains an explanation of the cases to which it will apply (e.g., 5.2.4, Design Data Base, applies when the software product contains a data base).

1.5 Use of This Standard. To facilitate the understanding and use of a standard of this magnitude, this section provides additional information.

1.5.1 Applicability. This standard applies to software development and maintenance projects.

This standard can be applied to commercial, scientific, and military software. Applicability is not restricted by size, complexity, or criticality of the software. This standard considers both the software and its context.

It is recognized that a project may be too small, in terms of schedule, budget, risk, nature, or use of software to be developed, used, and maintained, to warrant total application of the standard. In such cases, selected Activities may be applied even though compliance with this standard may not be claimed. This may also apply when only purchased software is involved.

A large project may be subdivided into smaller manageable projects, and this standard applied to each of the smaller projects and then to the whole. Similarly, some projects may be of long duration, and may be delivered in multiple versions or releases; it may be helpful in some cases to treat the development of each successive version as a separate project with its own life cycle.

1.5.2 Compliance. Compliance with this standard is defined as the performance of all mandatory Activities. Some mandatory Activities may occur in different instances (e.g., performing tests at various levels); compliance with this standard means the complete performance of each instance of the mandatory Activity. The standard does not specify instances of any Activity.

The performance of an Activity or an instance thereof is complete when all Input Information has been processed, and all Output Information has been generated. This may require several iterations of an Activity or instance.

All Input and Output Information are not required for a given occurrence of an iterative Activity. The presence of sufficient Input Information to permit processing by the Activity to begin constitutes the entry criterion, and the creation of any Output Information is a sufficient exit criterion.

This standard does not impose the order in which Activities must be performed. However, an order must be established by executing the Activities defined in Section 2 and the Activity in 3.1.3.

In some cases, certain Input and Output Information may not be required for completion of an Activity. These are indicated as "External" to the SLC and may not exist. To the extent that they exist, they must be processed by affected Activities.

This standard prescribes the *processes* of the software life cycle, not the *products* of that life cycle. Therefore, the standard does not require the completion of specific documents. The information generated by Activities, listed in the Output Information tables, may be collected into documents in any manner consistent with the selected SLCM.

In the event that this standard is contractually imposed, and one or more subcontractors are involved in the project, it is recommended that the requirements of this standard be imposed on the subcontractors.

²The term "mandatory" as used in this standard is synonymous with the term "essential."

³The term "if applicable" as used in this standard is synonymous with the terms "conditional" and "optional."

1.5.3 Intended Audience. This standard cannot be implemented by a single functional group within a development organization. Mapping this standard's Activities into an organization's SLCM, and coordinating these Activities with existing development and support methodologies and standards, require specific expertise and authority within the organization.

After mapping is complete, the Activities in Sections 3–7 are ready for execution. These Activity descriptions are directed to the functional specialist most likely to be performing them (e.g., the Requirements and Analysis Activities assume a basic familiarity with analysis techniques).

1.5.4 Process and Activity Relationships

1.5.4.1 Project Management Processes. There are three Processes in this section: the Project Initiation Process, the Project Monitoring and Control Process, and the Software Quality Management Process. The Project Initiation Process consists of those Activities that create and maintain the project framework. The Activities within the Project Monitoring and Control Process and the Software Quality Management Process are performed throughout the life of the project to ensure the appropriate level of project management and compliance with the mandated Activities.

1.5.4.2 Development-Oriented Processes. These are the Processes that must be performed before, during, and after the development of the software. The Pre-Development Processes are Concept Exploration and System Allocation. Development Processes include Requirements, Design, and Implementation. Finally, the Post-Development Processes include Installation, Operation and Support, Maintenance, and Retirement.

1.5.4.3 Integral Processes. This section includes those Processes that are necessary to ensure the successful completion of a project, but are not Development Processes. The Integral Processes are Verification and Validation, Software Configuration Management, Documentation Development, and Training. All of these Processes contain two types of Activities:

- (1) Those that are performed discretely and are therefore mapped into an SLCM.
- (2) Those that are performed in the course of completing another Activity. These are invoked Activities and will not be mapped into the SLCM in every instance.

Many Activities invoke, or call like a subroutine, appropriate Integral Process(es). This is an intuitive method of getting a task, such as the evaluation or production of a document, done without specifying an exact control flow within this standard.

To track the flow of a product into, through, and back from an Integral Process, there are generic sources and destinations listed in the Input and Output Information tables called "Creating Process." Within the Integral Process that is invoked, the Activity that first receives the product has a generic Input Information whose source is Creating Process. This product passes through one or more Activities of that Integral Process, then is returned to the invoking Activity through an Activity's Output Information, whose destination is Creating Process. An example of this flow is shown in Fig 1, which illustrates how one Activity within the Design Process invokes the Integral Processes. This figure will be easier to follow if it is compared directly with 5.2.7, "Perform Detailed Design." The order of Invoked Processes in the figure differs from that in 5.2.7.

The invoked Process is specified in the text by the name of the Process and the number of the first Activity to be performed within that Process [e.g., Verification and Validation (7.1.4)].

1.5.4.4 Use of I/O Tables. The Input Information to and Output Information from each Activity are listed in tables that accompany the Activity descriptions. As a convention of this document, Input and Output Information names are always capitalized in the text.

The I/O tables show the flow of Information through the Activities. The pertinent Information is listed in the left-hand column. The source or destination of the Information (both Process and Activity) is shown in the right-hand columns. The Information names were chosen, where possible, to suggest the titles of documents commonly used in the software industry. The Information names also frequently relate to the documents described in the standards listed in Section 8. Some of these Information items are represented in the text by acronyms that were deliberately chosen to suggest the commonly used names of the corresponding documents.

External Input Information sources and External Output Information destinations are outside the scope of this standard. External Input Information may or may not exist, and if it does not exist, it is not required. When an External Input does exist and is therefore used, it is presumed to include any associated documentation. External output destinations also may or may not exist. External sources and destinations are not necessarily Processes, and no corresponding Activities are shown in the I/O tables.

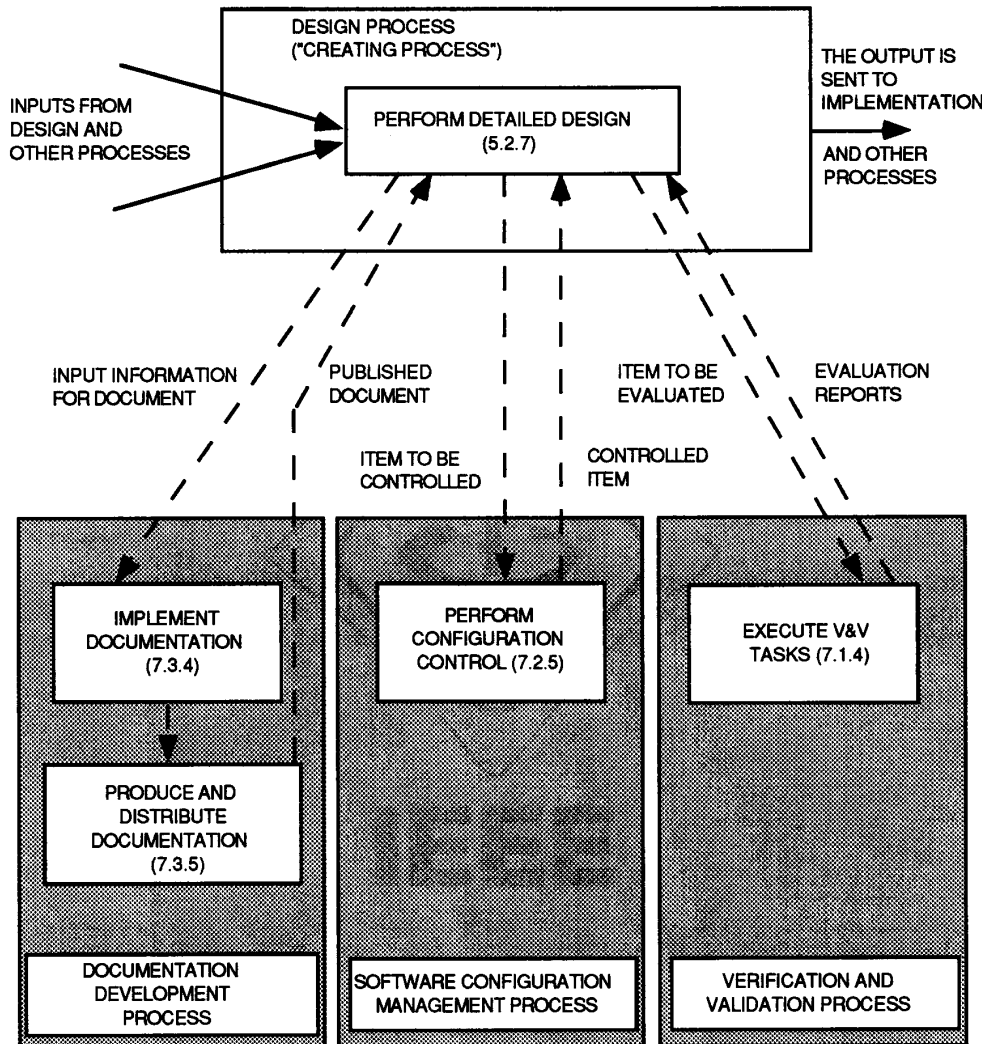


Fig 1
Example of Invoked Processes

In most cases, the Input Information and Output Information columns of the tables designate the specific information that enters or exits the Activity. However, since many Activities have Output Information whose destination is Retain Records (3.2.6), the various Input Information to Retain Records is collected under the term "Original Records." The corresponding Process and Activity columns refer simply to Originating Process and Originating Activity. Figure 2 depicts the conceptual flow of Input Information and Output Information into and out from an Activity, respectively.

1.5.5 Getting Started. Before beginning a project that will use this standard, the Activities need to be reviewed for applicability to a specific project and organized into a time sequence appropriate to that project. To perform that time sequencing, an SLCM must be chosen or developed and Activities mapped into the SLCM. This mapping is discussed in Section 2 and the mapping Activity in 3.1.3. Examples are given in Appendix A.

This mapping produces a temporal "road map" called the Software Life Cycle (SLC) used to follow this standard throughout the project. The mapped Activities must be initiated in their designated sequence.

1.5.6 Additional Considerations

1.5.6.1 Organizational Independence. This standard does not presume or dictate an organizational structure for a software project. Therefore, it is neither implied nor required that Activities within a Process be performed by the same organizational entity, nor that an organizational entity's involvement be concentrated in only one Process. To ensure that all Activities are assigned to an appropriate organizational entity, the concept of Activity Ownership is described in the Activity in 3.1.3.

1.5.6.2 Combining Documents. The Information developed in this standard, as shown in the Output Information tables, may carry generic names similar to those used in other IEEE standards. This does not imply that the format and content specified in other IEEE standards must be followed, nor that this information must be packaged into documents in any particular manner.

Combination of documents into a single document is acceptable as long as understanding is not compromised.

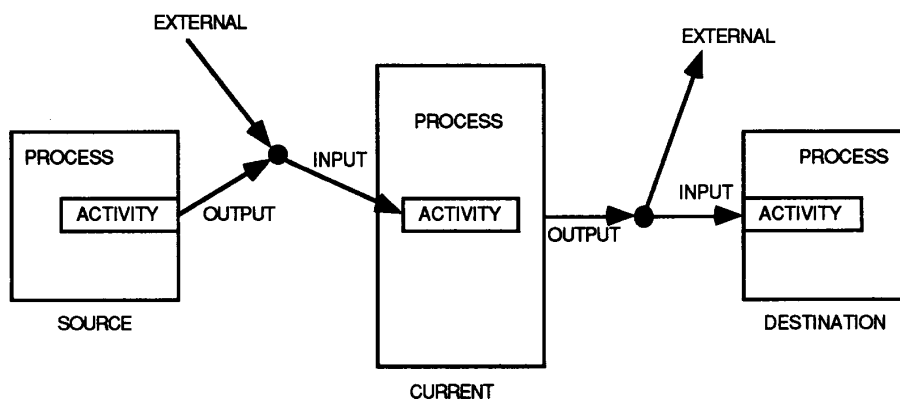


Fig 2
Information Flow

2. Software Life Cycle Model Process

2.1 Overview. Many variables affect an organization's selection of a software life cycle model (SLCM). While this standard neither dictates nor defines a specific software life cycle (SLC) or its underlying methodologies, it does require that an SLCM be chosen and used.

This Process provides the Activities required to identify candidate SLCMs and select the SLCM to be used by other Activities in the standard.

This standard includes the specification of the non-time-ordered set of "Mandatory" Activities that must be incorporated into an SLCM. An SLCM (e.g., Rapid Prototyping) defines a specific approach to producing software. It specifies a time-ordered set of Activities (including all of the "Mandatory" Activities identified in this standard), which is to be used as the basis for mapping the Activities of this standard. An SLCM may also propose standards for the performance of the Activities or the deliverables produced during the project.

The SLC (defined for a project by the Activity in 3.1.3) is the time-ordered set of Activities or instances of Activities to be performed. This set is to be mapped into a selected SLCM. The SLC also identifies specific responsibilities for each Activity. While the same SLCM may be valid for several projects, each project must define its own SLC.

Once an SLCM is selected, there are two additional required actions:

- (1) Mapping the Activities described in this standard into the chosen life cycle (3.1.3).
- (2) Identifying and documenting the standards and controls that govern the SLC (3.1.5).

Figure 3 illustrates this progression from this standard to a project-specific SLC.

Section 1.5, "Use of This Standard," provides background information necessary for the successful understanding and application of this material. It should be read prior to proceeding further in this section.

2.2 Activities List

- (1) Identify Candidate Software Life Cycle Models
- (2) Select Project Model

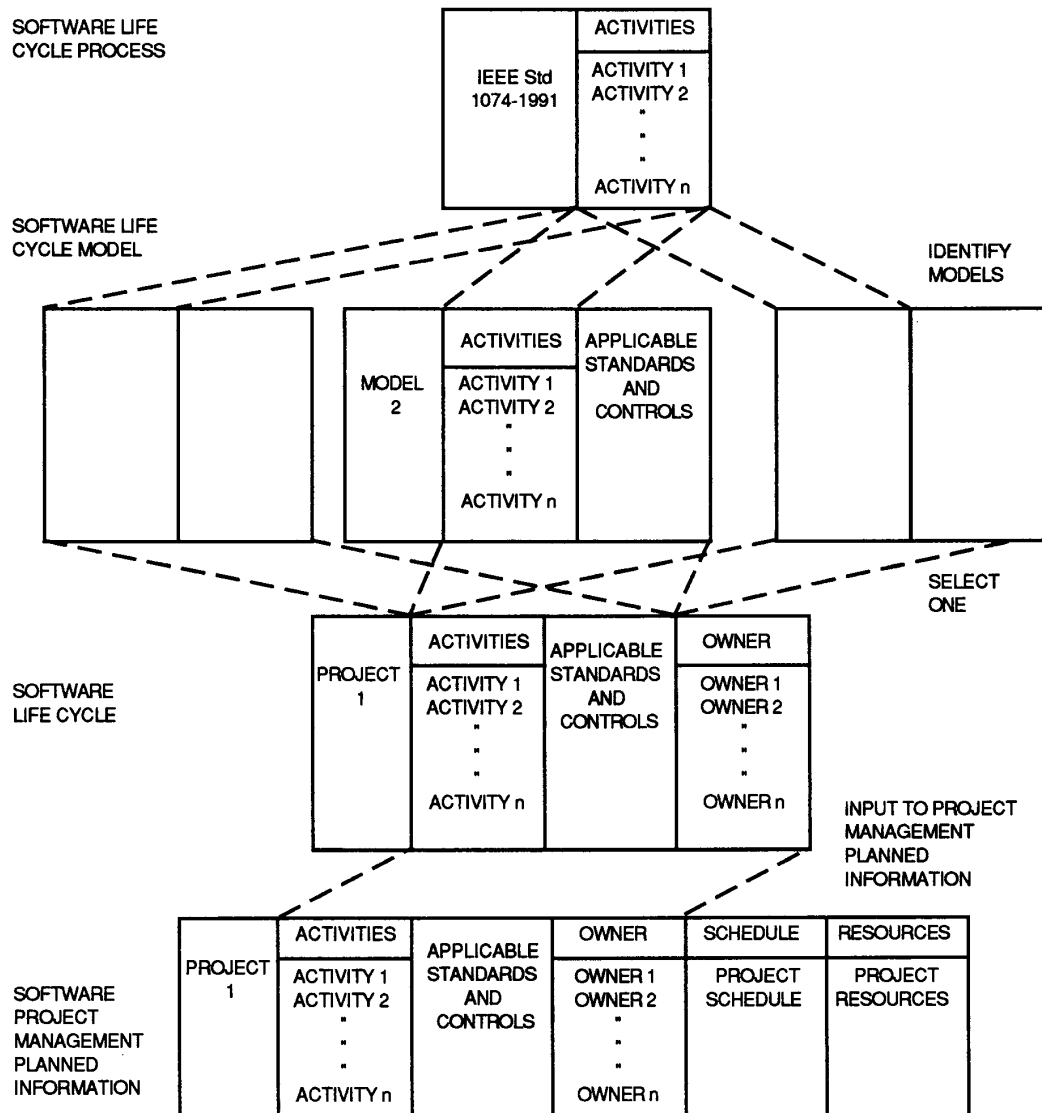


Fig 3
Software Life Cycle Relationships

2.3 Identify Candidate Software Life Cycle Models**2.3.1 Input Information**

Input Information	Source	
	Process	Activity
Available Software Life Cycle Models)	External	
Constraints	External	

2.3.2 Description. In this Activity, the set of Available SLCMs and applicable Constraints shall be considered and Candidate SLCMs identified. A new model may be constructed by combining elements of other SLCMs.

Maintenance is an iteration of the Software Life Cycle, and the SLCM must support this iteration.

2.3.3 Output Information

Output Information	Destination	
	Process	Activity
Candidate Software Life Cycle Model(s)	Software Life Cycle Model	Select Project Model (2.4)

2.4 Select Project Model**2.4.1 Input Information**

Input Information	Source	
	Process	Activity
Historical Project Records	External	
Constraints	External	
Candidate Software Life Cycle Model(s)	Software Life Cycle Model	Identify Candidate Software Life Cycle Models (2.3)

2.4.2 Description. In this Activity, one of the candidate SLCMs from 2.3.2 is selected for use.

Based on the type of product (interactive, batch, transaction processing, etc.), Constraints, and Historical Project Records, an SLCM analysis shall be conducted, and a decision made as to which model will best support the management of the project.

It is possible for an organization to have more than one SLCM, but only one model may be selected for a project. It is not necessary to have a single, organization-wide SLCM.

The SLCM shall provide the necessary framework for software projects to map the Activities to produce the SLC (as shown in Fig 3). The mapping effort is specified in the Project Initiation Process, Map Activities to Software Life Cycle Model (3.1.3).

2.4.3 Output Information

Output Information	Destination	
	Process	Activity
Selected Software Life Cycle Model	Project Initiation	Map Activities to Software Life Cycle Model (3.1.3)

3. Project Management Processes

These are the Processes that initiate, monitor, and control software projects throughout the software life cycle (SLC).

3.1 Project Initiation Process

3.1.1 Overview. This Process contains those Activities that create the framework for the project. During this Process, the SLC is created for this project, and plans for managing the project are established. Standards, methodologies, and tools needed to manage and execute the project are identified and a plan prepared for their timely implementation.

Section 1.5, "Use of This Standard," provides background information necessary for the successful understanding and application of this material. It should be read prior to proceeding further in this section.

3.1.2 Activities List

- (1) Map Activities to Software Life Cycle Model
- (2) Allocate Project Resources
- (3) Establish Project Environment
- (4) Plan Project Management

3.1.3 Map Activities to Software Life Cycle Model

3.1.3.1 Input Information

Input Information	Source	
	Process	Activity
Contractual Requirements	External	
Selected Software Life Cycle Model	Software Life Cycle Model	Select Project Model (2.4)
Statement of Need	Concept Exploration	Refine and Finalize the Idea or Need (4.1.7)

3.1.3.2 Description. The Activities identified in this standard shall be mapped into the selected SLC Model (SLCM). Mapping involves establishing the chronological relationship of the Activities in this standard according to the selected SLCM. It may be necessary to use the Contractual Requirements and the Statement of Need to accomplish this mapping. Appendix A provides several examples of such mappings. Appendix B is a template for adding additional project-specific information, such as document titles and applicable standards, to the mapped Activities.

The use of certain software development methods defines the execution of some Activities to be automated. Compliance with this standard must be demonstrated by mapping those automated Activities into the appropriate points within the SLCM.

Each Activity shall be assigned a single "owner." An owner is a single point of contact, and is identified by organizational position. Ownership is assumed by the person currently filling that position. Each owner has the responsibility and authority to control and complete the Activity within the planned schedule and budget. In addition, each owner is accountable for the quality of the Activity outputs. If Activities are to be performed by multiple organizations, the owning organization and position of the owner shall be identified. In the case of multiple instances of an Activity, an owner for each instance shall be identified.

The resulting map of the Activities to be performed, with their corresponding owners, is the SLC for this project. All "If Applicable" Activities that do not apply to this project shall be identified and explained in the List of Activities Not Used.

3.1.3.3 Output Information

Output Information	Destination	
	Process	Activity
Software Life Cycle	Project Initiation	Allocate Project Resources (3.1.4)
		Establish Project Environment (3.1.5)
		Plan Project Management (3.1.6)
List of Activities Not Used	Project Monitoring and Control	Retain Records (3.2.6)

3.1.4 Allocate Project Resources

3.1.4.1 Input Information

Input Information	Source	
	Process	Activity
Historical Project Records	External	
Resources	External	
Statement of Need	Concept Exploration	Refine and Finalize the Idea or Need (4.1.7)
Software Life Cycle	Project Initiation	Map Activities to Software Life Cycle Model (3.1.3)
System Functional Software Requirements	System Allocation	Decompose System Requirements (4.2.5)

3.1.4.2 Description. Resource Allocations shall be identified at the Software Life Cycle's Activity level. Resources to be allocated include personnel, equipment, space, etc. Available Historical Project Records and the Statement of Need may provide valuable insight into Resource Allocation.

3.1.4.3 Output Information

Output Information	Destination	
	Process	Activity
Resource Allocations	Project Initiation	Establish Project Environment (3.1.5)
		Plan Project Management (3.1.6)
	Project Management and Control	Analyze Risks (3.2.3)

3.1.5 Establish Project Environment

3.1.5.1 Input Information

Input Information	Source	
	Process	Activity
Methodologies	External	
Standards	External	
Tools	External	
Software Library	External	
Purchased Software	External	
Contractual Requirements	External	
Analysis of Risks	Project Monitoring and Control	Analyze Risks (3.2.3)
Software Life Cycle	Project Initiation	Map Activities to Software Life Cycle Model (3.1.3)
Defined Metrics	Software Quality Management	Define Metrics (3.3.4)
Collection and Analysis Methods	Software Quality Management	Define Metrics (3.3.4)
Resource Allocations	Project Initiation	Allocate Project Resources (3.1.4)
Statement of Need	Concept Exploration	Refine and Finalize the Idea or Need (4.1.7)

3.1.5.2 Description. The needs of the project for procedural and technological Tools, Methodologies, and Standards shall be defined. Approaches to these needs shall be identified and evaluated. These approaches could include automated and nonautomated tools, modeling and prototyping methodologies, environment simulators, test beds, and software libraries. Selection criteria for Tools and Methodologies should include resource, schedule, safety, and security considerations, and the project requirements defined in the Statement of Need and Analysis of Risks. The project standards shall include requirements, design, coding, test, and documentation standards.

After evaluating the approaches, a set of Tools, Methodologies, Standards, and reusable or Purchased Software shall be selected to provide the Project Environment, considering the Input Information.

The selected tools shall be acquired or developed and installed for use in project Activities. The owner of this Activity shall ensure that applicable personnel are familiar with the Tools, Methodologies, and Standards selected for the project.

For assistance in identifying applicable standards, [9] should be consulted.

Prior to distribution of the Project Environment, the Training Process (7.4.4) shall be invoked.

3.1.5.3 Output Information

Output Information	Destination	
	Process	Activity
Project Environment	Project Initiation	Plan Project Management (3.1.6)

3.1.6 Plan Project Management**3.1.6.1 Input Information**

Input Information	Source	
	Process	Activity
Contractual Requirements	External	
Software Life Cycle	Project Initiation	Map Activities to Software Life Cycle Model (3.1.3)
Resource Allocations	Project Initiation	Allocate Project Resources (3.1.4)
Project Environment	Project Initiation	Establish Project Environment (3.1.5)
Contingency Planned Information	Project Monitoring and Control	Perform Contingency Planning (3.2.4)
Project Management Reported Information	Project Monitoring and Control	Manage the Project (3.2.5)
Preliminary Statement of Need	Concept Exploration	Identify Ideas or Needs (4.1.3)
Recommendations	Concept Exploration	Conduct Feasibility Studies (4.1.5)
Statement of Need	Concept Exploration	Refine and Finalize the Idea or Need (4.1.7)

3.1.6.2 Description. Project management planning requires collection and synthesis of a great deal of information into a coherent and organized Software Project Management Planned Information (SPMP) based on the SLC. This Activity shall initially define and subsequently update the SPMP using the Input Information. This Activity shall detail the project organization and assign responsibilities. Standards, methodologies, and tools for configuration management, quality assurance, verification and validation, training, documentation, and development shall be specified. This Activity shall apportion the project budget and staffing, and define schedules, using the applicable Input Information. It also shall define procedures for scheduling, tracking, and reporting, and shall address considerations such as regulatory approvals, required certifications, user involvement, subcontracting, and security.

This Activity shall include planning for support, problem reporting, and retirement. Support planning shall include methods for supporting the software in the operational environment. Problem Reporting and Resolution Planning Information shall include, at a minimum, defining a method for logging, routing, and handling problem reports; categories of severity; and the method for verifying problem resolution. Retirement Planned Information shall address issues such as probable retirement date, archiving, replacement, and residual support issues.

As new or revised Input Information is received in this Activity, project plans shall be updated and further project planning shall be based upon these updated plans.

Additional guidance for SPMPs can be found in [17].

Prior to distribution of the SPMP, the following Processes shall be invoked:

- (1) Verification and Validation (7.1.4)
- (2) Software Configuration Management (7.2.5)
- (3) Documentation Development (7.3.4)

3.1.6.3 Output Information

Output Information	Destination	
	Process	Activity
Problem Reporting and Resolution Planned Information	Project Monitoring and Control	Manage the Project (3.2.5)
		Analyze Risks (3.2.3)
		Implement Problem Reporting Method (3.2.7)
Retirement Planned Information	Project Monitoring and Control Retirement	Manage the Project (3.2.5)
		Notify User (6.4.3)
		Conduct Parallel Operations (If Applicable) (6.4.4)
Software Project Management Planned Information	Most Processes	Most Activities
Support Planned Information	Project Monitoring and Control	Analyze Risks (3.2.3)
		Manage the Project (3.2.5)
	Operation and Support	Maintain Support Request Log (6.2.5)
		Operate the System (6.2.3)
		Provide Technical Assistance and Consulting (6.2.4)

3.2 Project Monitoring and Control Process

3.2.1 Overview. Monitoring and control is an iterative Process of tracking, reporting, and managing costs, schedules, problems, and performance of a project throughout its life cycle. The progress of a project is reviewed and measured against project milestones established in the Software Project Management Planned Information (SPMP).

Section 1.5, "Use of This Standard," provides background information necessary for the successful understanding and application of this material. It should be read prior to proceeding further in this section.

3.2.2 Activities List

- (1) Analyze Risks
- (2) Perform Contingency Planning
- (3) Manage the Project
- (4) Retain Records
- (5) Implement Problem Reporting Method

3.2.3 Analyze Risks

3.2.3.1 Input Information

Input Information	Source	
	Process	Activity
Procurement/Lease Data	External	
System Constraints	External	
Historical Project Records	External	
Support Planned Information	Project Initiation	Plan Project Management (3.1.6)
Resource Allocations	Project Initiation	Allocate Project Resources (3.1.4)
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Problem Reporting and Resolution Planned Information	Project Initiation	Plan Project Management (3.1.6)
Transition Impact Statement (If Applicable)	Concept Exploration	Plan System Transition (If Applicable) (4.1.6)
Statement of Need	Concept Exploration	Refine and Finalize the Idea or Need (4.1.7)
Software Interface Requirements	Requirements	Define Interface Requirements (5.1.4)
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)
Software Design Description	Design	Perform Detailed Design (5.2.7)
Integration Planned Information	Implementation	Plan Integration (5.3.7)
Analysis Reported Information	Verification and Validation	Collect and Analyze Metric Data (7.1.5)
Test Planned Information(s)	Verification and Validation	Plan Testing (7.1.6)
Test Summary Reported Information	Verification and Validation	Execute the Tests (7.1.8)

3.2.3.2 Description. Because risk management often involves trade-offs between many factors, risk analysis is an iterative Activity performed throughout a project's life. This analysis

shall consider project risks, including technical, economic, operational support, and schedule risks.

Factors that may impair, prevent, or require technical trade-offs for accomplishing the technical objectives of the project or product shall be identified and analyzed. Technical factors may include such items as real-time performance, safety considerations, security considerations, implementation considerations, testability, and maintainability. Analytical approaches for technical risk assessment may include static and dynamic modeling and simulation, prototyping, independent reviews, and audits.

Cost, resource factors, earnings, liabilities, or other economic measures involved in the project shall be identified and analyzed. The objective of this analysis is to identify potential economic opportunities, losses, and trade-offs. Analytical approaches for economic risk assessment may include financial analysis, such as return on investment and possible incentive and penalty contract clauses.

Operational and support risk analysis shall determine the probability that the delivered software will meet the user's requirements. Operational and support requirements such as interoperability, security, performance, installability, and maintainability shall be considered. Both completeness of, and conformance to, these requirements shall be analyzed.

Cost, resource, technical, and other requirements shall be evaluated for their impact on project schedule. This analysis should consider project interdependence and the effect of schedule adjustments. Analytical approaches for schedule risk assessment may include critical path analysis and resource leveling techniques.

3.2.3.3 Output Information

Output Information	Destination	
	Process	Activity
Analysis of Risks	Project Initiation	Establish Project Environment (3.1.5)
	Project Monitoring and Control	Perform Contingency Planning (3.2.4)
	Requirements	Define and Develop Software Requirements (5.1.3)
	Verification and Validation	Plan Verification and Validation (7.1.3)

3.2.4 Perform Contingency Planning

3.2.4.1 Input Information

Input Information	Source	
	Process	Activity
Analysis of Risks	Project Monitoring and Control	Analyze Risks (3.2.3)
Analysis Reported Information	Verification and Validation	Collect and Analyze Metric Data (7.1.5)

3.2.4.2 Description. This Activity shall define alternative actions in the event that a given risk materializes, using the Input Information. Contingency Planned Information shall include resource planning and the establishment of trigger conditions that would invoke a contingency action. Contingency actions may include consideration of revised requirements, delay, or cancellation of the project.

3.2.4.3 Output Information

Output Information	Destination	
	Process	Activity
Contingency Planned Information	Project Initiation	Plan Project Management (3.1.6)
	Project Monitoring and Control	Manage the Project (3.2.5)

3.2.5 Manage the Project**3.2.5.1 Input Information**

Input Information	Source	
	Process	Activity
Problem Reporting and Resolution Planned Information	Project Initiation	Plan Project Management (3.1.6)
Retirement Planned Information	Project Initiation	Plan Project Management (3.1.6)
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Support Planned Information	Project Initiation	Plan Project Management (3.1.6)
Contingency Planned Information	Project Monitoring and Control	Perform Contingency Planning (3.2.4)
Software Quality Management Planned Information	Software Quality Management	Plan Software Quality Management (3.3.3)
Quality Improvement Recommendations	Software Quality Management	Identify Quality Improvement Needs (3.3.6)
Integration Planned Information	Implementation	Plan Integration (5.3.7)
Installation Reported Information	Installation	Install Software (6.1.5)
Evaluation Reported Information	Verification and Validation	Execute Verification and Validation Tasks (7.1.4)
Analysis Reported Information	Verification and Validation	Collect and Analyze Metric Data (7.1.5)
Test Planned Information(s)	Verification and Validation	Plan Testing (7.1.6)
Test Summary Reported Information	Verification and Validation	Execute the Tests (7.1.8)
Status Reported Information	Software Configuration Management	Perform Status Accounting (7.2.6)
Feedback Data	Operation and Support	Operate the System (6.2.3)

3.2.5.2 Description. Throughout the life cycle, the progress of the project shall be reviewed and measured against the established milestones and budget in the plan(s) (i.e., predicted and planned progress versus actual progress, and budgeted versus actual expenditures). Project tracking and reporting includes analyzing the Input Information, collecting other pertinent data, and monitoring project Activities. Anomalies may result. Risk management procedures must be implemented to control risk.

This Activity also encompasses the day-to-day management of the project needed to ensure successful project completion. Information collected within this Activity is used to improve the performance of the project.

Prior to distributing the Project Management Reported Information, the Verification and Validation Process (7.1.4) should be invoked.

3.2.5.3 Output Information

Output Information	Destination	
	Process	Activity
Project Management Reported Information	Project Initiation	Plan Project Management (3.1.6)
	Project Monitoring and Control	Retain Records (3.2.6)
	External	
Anomalies	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)

3.2.6 Retain Records

3.2.6.1 Input Information

Input Information	Source	
	Process	Activity
Documentation Retention Standards	External	
Original Records	Originating Process	Originating Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Software Configuration Management Planned Information	Software Configuration Management	Plan Configuration Management (7.2.3)
Documentation Planned Information	Documentation Development	Plan Documentation (7.3.3)
Published Document	Documentation Development	Produce and Distribute Documentation (7.3.5)

3.2.6.2 Description. This Activity accepts the original project documentation and records from each originating Process. The records shall be retained in accordance with the SPMP, Software Configuration Management Planned Information, and any external document retention standards. Input Information documentation becomes part of the Historical Project Records of the organization. Uses for these records may include project audits, future project planning, and corporate accounting.

3.2.6.3 Output Information

Output Information	Destination	
	Process	Activity
Historical Project Records	External	

3.2.7 Implement Problem Reporting Method**3.2.7.1 Input Information**

Input Information	Source	
	Process	Activity
Anomalies	External	
	Creating Process	
Controlled Item	Software Configuration Management	Perform Configuration Control (7.2.5)
Problem Reporting and Resolution Planned Information	Project Initiation	Plan Project Management (3.1.6)

3.2.7.2 Description. This Activity accepts Anomalies from any source and prepares a problem report. The problem report shall contain information as specified in the Problem Reporting and Resolution Planned Information (PR&RP). Possible problem solutions may be suggested by the problem reporter. Problems may be resolved through corrections or enhancements (as defined in the PR&RP). Corrections are documented in the Correction Problem Reported Information for further consideration. Enhancements may be documented in the Enhancement Problem Reported Information and are possible candidates for new projects. A Report Log shall be maintained to assure that all problems are tracked until they are resolved and the resolution has been approved.

This Activity shall also analyze the problem including the Controlled Item, the problem report, and the Report Log to make the following determinations:

- (1) What the anomalies are.
- (2) Source and cause of product or process problem.
- (3) Product(s) or process(es) presumed to contain the error, including documentation.
- (4) Problem severity.
- (5) Course of corrective action.

Problem reports that originate from an Activity not included in this standard are noted as resolved within this Activity and forwarded for appropriate action to the responsible authority.

This Activity shall monitor the problem correction efforts performed by the responsible Process, shall determine (according to the PR&RD) that the implementation of the solution by the responsible Process has been completed, and shall then record the resolution of the problem in the Resolved Problem Reported Information. The Resolved Problem Reported Information shall be distributed as specified in the Problem Reporting and Resolution Planned Information.

Further information related to this Activity may be found in [15].

The Resolved Problem Reported Information should be made available to the Process or external source that reported the problem.

Prior to distribution of a Problem Reported Information or the Report Log, the Software Configuration Management Process (7.2.5) should be invoked.

3.2.7.3 Output Information

Output Information	Destination	
	Process	Activity
Resolved Problem Reported Information	External	
	Creating Process	
	Software Quality Management	Manage Software Quality (3.3.5)
	Verification and Validation	Execute Verification and Validation Tasks (7.1.4)
		Collect and Analyze Metric Data (7.1.5)
Report Log	Software Quality Management	Manage Software Quality (3.3.5)
	Verification and Validation	Collect and Analyze Metric Data (7.1.5)
Enhancement Problem Reported Information	Concept Exploration	Identify Ideas or Needs (4.1.3)
	Verification and Validation	Collect and Analyze Metric Data (7.1.5)
Correction Problem Reported Information	Maintenance	Reapply Software Life Cycle (6.3.3)
	Verification and Validation	Collect and Analyze Metric Data (7.1.5)

3.3 Software Quality Management Process

3.3.1 Overview. An important role of Software Quality Management is to address the planning and administration of the Software Quality Assurance (SQA) program. It further addresses such concerns as client satisfaction (which transcends adherence only to established technical requirements), and internal quality improvement programs. The responsibilities, functions, obligations, and duties of an SQA program are properly a constituent part of all Activities in the Software Life Cycle, and thus are interspersed into each Activity as appropriate. Software Quality Management is the methodology used in this standard for tying the SQA responsibilities together with other Quality concerns. The Activities of this Process span the entire Software Life Cycle (SLC).

Section 1.5, "Use of This Standard," provides background information necessary for the successful understanding and application of this material. It should be read prior to proceeding further in this section.

3.3.2 Activities List

- (1) Plan Software Quality Management
- (2) Define Metrics
- (3) Manage Software Quality
- (4) Identify Quality Improvement Needs

3.3.3 Plan Software Quality Management

3.3.3.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Defined Metrics	Software Quality Management	Define Metrics (3.3.4)
Collection and Analysis Methods	Software Quality Management	Define Metrics (3.3.4)

3.3.3.2 Description. A Software Quality Management program shall be initiated and documented.

It shall include a Software Quality Assurance program, which may be documented separately.

The goals of the Software Quality Management program are to identify SQA actions, describe supplier quality requirements, address client satisfaction, and provide for the identification of quality improvement needs.

Overall quality objectives are derived using the organizational guidelines and contractual requirements from the Software Project Management Planned Information.

The program information shall include the Software Quality Management organization and responsibilities, and the tools, techniques, and methodologies to implement the program.

The goals and standards to be applied to the project shall also be identified.

The goals are further expanded into quality objectives and milestones in the Software Quality Management Planned Information.

Further information related to this Activity may be found in [2] and [8].

Prior to distribution of the Software Quality Management Planned Information, the following Processes shall be invoked:

- (1) Verification and Validation (7.1.4)
- (2) Software Configuration Management (7.2.5)
- (3) Documentation Development (7.3.4)

3.3.3.3 Output Information

Output Information	Destination	
	Process	Activity
Software Quality Management Planned Information	Project Monitoring and Control	Manage the Project (3.2.5)
	Software Quality Management	Define Metrics (3.3.4)
		Manage Software Quality (3.3.5)
		Identify Quality Improvement Needs (3.3.6)
	Verification and Validation	Plan Verification and Validation (7.1.3)
		Collect and Analyze Metric Data (7.1.5)

3.3.4 Define Metrics

3.3.4.1 Input Information

Input Information	Source	
	Process	Activity
Software Quality Management Planned Information	Software Quality Management	Plan Software Quality Management (3.3.3)
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)

3.3.4.2 Description. The metrics required for the project, based on the Software Project Management Planned Information, shall be defined. Metrics should be applied to the products of the project and to the processes that affect the project. The metrics shall be used throughout the SLC. For each Defined Metric, Collection and Analysis Methods shall be specified.

Further information related to this Activity may be found in [6], [7], [16], and [19].

Prior to the distribution of Defined Metrics, the Verification and Validation Process (7.1.4) shall be invoked.

3.3.4.3 Output Information

Output Information	Destination	
	Process	Activity
Defined Metrics	Software Quality Management	Manage Software Quality (3.3.5)
		Plan Software Quality Management (3.3.3)
	Project Initiation	Establish Project Environment (3.1.5)
	Verification and Validation	Collect and Analyze Metric Data (7.1.5)
Collection and Analysis Methods	Software Quality Management	Plan Software Quality Management (3.3.3)
		Manage Software Quality (3.3.5)
	Verification and Validation	Collect and Analyze Metric Data (7.1.5)
	Project Initiation	Establish Project Environment (3.1.5)