

IEEE Std 1074-1995

(revision of

IEEE Std 1074-1991)

IEEE Standard for Developing Software Life Cycle Processes

IEEE Standards Board

Approved September 21, 1995

Sponsor

Software Engineering Standards Committee

of the

IEEE Computer Society

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 provided. Non-software Activities, such as hardware development and purchasing, are outside of the scope of this standard. Also provided is associated Input and Output Information.

Keywords: activities, mapping, processes, software life cycle, software life cycle model, software life cycle process.

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

Copyright © 1996 by the Institute of Electrical and Electronics Engineers, Inc. All rights reserved. Published 1996. Printed in the United States of America.

ISBN ISBN 1-55937-588-4

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 that 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 08855-1331 USA

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; (508) 750-8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

(This introduction is not part of IEEE Std 1074-1995, IEEE Standard for Developing Software Life Cycle Processes.)

This introduction 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 introduction is nonbinding.

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.

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 standard 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.

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. Subsequently, the document was revised to correct some trivial errors (spelling, etc.) and was submitted for ballot in August, 1993 and for recirculation in April, 1995.

> 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*
Dennis E. Nickle, *Vice Chair*
Lynn D. Ihlenfeldt, *Secretary*

Joe Albano
Tom Antezac
Michael Buckley
Susan Burgess
David W. Burnett
David Burrows
Dan Chang
Paul Christensen
Raymond Day

Marvin Doran
Mike Ellwood
Arden Forrey
Jean Gilmore
Art Godin
John Graham
Daniel Gray
Joseph Guidos *
Rob Harker

Carolyn Harrison
Peter Harvey
Eric Hensel
Denise Holmes
John Horch
George Jackelen
Linell Jones
Laurel Kaleda
Phil Keys

Robert Kierzyk
Tom Kurihara
Bill Mar
Pat Marcinko
Darrell Marsh
Leroy May
Denis Meredith
Keith R. Middleton

Manijeh Moghis
Richard Morton
Brian Nejmeh
John Pellegri
Hans Schaefer
Isaac Shadman
Robert Shillato
David Smith
Kelley Stalder

John Swearingen
Diane Switzer
David Taylor
George Tucker
Odo Wang
Richard Werling
Allan Willey
Geraldine Zimmerman

* Deceased

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
Robert Bloth
William Blum
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 Gershtkovitch
Sam Godfrey
Ole Golubjatnikov
Jim Harkins
James Heil
Cheng Hu
Jim Hughes

Suzana Hutz
Ron Hysom
Phyllis Illyefalvi
Corbin Ingram
Ramon Izbinsky
John Jenkins
Tom Jepson
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
Noritoshi Murakami
Christopher Neubert
Rocco Novak
George O'Connell
John Patchen
Jeff Pattee
John Pelegri

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

The following persons were on the balloting committee:

A. F. Ackerman
Oddur Benediktsson
H. Ronald Berlack
B. P. Bhat
Mark Bilger
Ronald Blair
William J. Boll
Sandro Bologna
Fletcher Buckley
Susan M. Burgess
David W. Burnett
W. Larry Campbell
Neva M. Carlson
Francois Coallier
Geoff Cozens
Michael A. Daniels
Geoffrey Darnton

Taz Daughtrey
Neil Davis
Raymond Day
Bostjan K. Derganc
Leo G. Egan
Caroline L. Evans
Caroline L. Evans
John W. Fendrich
Peter Fillery
Jon J. Fineman
Eitan Froumine
Simon Gabrihelidis
Juan Garbajosa-Sopena
Yair Gershtkovitch
Jean Gilmore
John Garth Glynn
Donald Gotterbarn

Lawrence M. Gunther
David A. Gustafson
John Harauz
Rob Harker
Matthew S. Harriss
William Hefley
Mark Heinrich
John W. Horch
Lynn D. Ihlenfeldt
John O. Jenkins
David Johnson
Robert J. Kierzyk
Scotty Kilbourne
Peter Klopfenstein
Robert Kosinski
Thomas M. Kurihara
Helmut Kurzdorfer

Boniface Lau
J. Dennis Lawrence
Fang Chin Lim
Ben Livson
Joseph Maayan
Hosakere N. Mahabala
Harold Mains
Pat Marcinko
Robert A. Martin
Roger Martin
Sue McGrath
Bino Nanni
Ann Marie Neufelder
Dennis E. Nickle

Mike Ottewill
Indradeb P. Pal
Joseph A. Palermo
Patricia Rodriguez
Sergiu Samuel
Stephen R. Schach
Norman Schneidewind
Wolf A. Schnoege
David J. Schultz
Gregory D. Schumacher
Leonard W. Seagren
Tony Sgarlatti
Robert W. Shillato
David M. Siefert

Alfred R. Sorkowitz
Vijaya Srivastava
Alfred Strohmeier
Robert N. Sulgrove
George D. Tice
Leonard L. Tripp
Margaret C. Updike
Tom Vaiskunas
Dolores Wallace
William M. Walsh
Richard Werling
Natalie C. Yopconka
Janusz Zalewski
Geraldine Zimmerman

When the IEEE Standards Board approved this standard on September 21, 1995, it had the following membership:

E. G. “Al” Kiener, *Chair*
Donald C. Loughry, *Vice Chair*
Andrew G. Salem, *Secretary*

Gilles A. Baril
Clyde R. Camp
Joseph A. Cannatelli
Stephen L. Diamond
Harold E. Epstein
Donald C. Fleckenstein
Jay Forster *
Donald N. Heirman
Richard J. Holleman

Jim Isaak
Ben C. Johnson
Sonny Kasturi
Lorraine C. Kevra
Ivor N. Knight
Joseph L. Koepfinger *
D. N. “Jim” Logothetis
L. Bruce McClung
Marco W. Migliaro

Mary Lou Padgett
John W. Pope
Arthur K. Reilly
Gary S. Robinson
Ingo Rüsç
Chee Kiow Tan
Leonard L. Tripp
Howard L. Wolfman

* Member Emeritus

Also included are the following nonvoting IEEE Standards Board liaisons:

Satish K. Aggarwal

Richard B. Engelman
Robert E. Hebner
Angela M. Girardi, *Editor*

Chester C. Taylor

Contents

CLAUSE.....	PAGE
1. Overview.....	1
1.1 Scope.....	1
1.2 References.....	1
1.3 Definitions and acronyms	2
1.4 Organization of this document.....	4
1.5 Use of this standard.....	5
2. Software Life Cycle Model Process	9
2.1 Overview.....	9
2.2 Activities list	10
2.3 Identify Candidate Software Life Cycle Models	10
2.4 Select Project Model	12
3. Project Management Processes.....	13
3.1 Project Initiation Process	13
3.2 Project Monitoring and Control Process.....	20
3.3 Software Quality Management Process	29
4. Pre-development Processes.....	34
4.1 Concept Exploration Process	34
4.2 System Allocation Process.....	40
5. Development Processes	44
5.1 Requirements Process	44
5.2 Design Process	48
5.3 Implementation Process	54
6. Post-development Processes	61
6.1 Installation Process	61
6.2 Operation and Support Process.....	66
6.3 Maintenance Process.....	70
6.4 Retirement Process.....	71
7. Integral Processes.....	74
7.1 Verification and Validation Process	74
7.2 Software Configuration Management Process.....	83
7.3 Documentation Development Process	88
7.4 Training Process.....	92
8 Bibliography	97
ANNEX	
Annex A (informative) Mapping Software Life Cycle Processes to various examples of SoftwareLife cycles	101
Annex B (informative) Software Project Management Tailoring Template	118
Annex C (informative) Process interrelationships.....	125

IEEE Standard for Developing Software Life Cycle Processes

1. Overview

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 clause 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 [B1].¹

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

1.3.1.2 analysis: Examination for the purpose of understanding.

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

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

1.3.1.5 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 [B5].

1.3.1.6 data base: A collection of data fundamental to a system [B21].

1.3.1.7 evaluation: Determination of fitness for use.

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

1.3.1.9 function: A specific purpose of an entity or its characteristic action [B21].

1.3.1.11 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 [B1].

1.3.1.10 Instance: The complete mapping of an Activity which processes all of its Input Information and generates all of its Output Information.

1.3.1.12 Invocation: The initiation of the invoked Activities of an Integral Process.

1.3.1.13 Iteration: Any execution of an Activity where at least some Input Information is processed and some Output Information is created. One or more Iterations comprise an Instance.

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

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

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

1.3.1.17 problem: The inability of a system or component to perform its required functions within specified performance requirements [B1].

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

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

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

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

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

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

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

1.3.1.24 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.

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

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

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

1.3.1.28 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 [B5] .

1.3.1.29 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 [B16] .

1.3.1.30 unit: A logically separable part of a program [B1] .

1.3.1.31 user: The person, or persons, who operate or interact directly with the system [B5] .

1.3.2 Acronyms

The following acronyms appear within the text of this standard:

CASE	Computer-Aided Software Engineering
I/O	Input/Output
PR&RPI	Problem Report and Resolution Planned Information
SCMPI	Software Configuration Management Planned Information
SDD	Software Design Description
SLC	Software Life Cycle
SLCM	Software Life Cycle Model
SPMPI	Software Project Management Planned Information
SQA	Software Quality Assurance
SRS	Software Requirements
SVVPI	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 clauses. Table 1 depicts this organization.

Table 1 — Organization of the standard

Clause	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's required actions are specified in its constituent Activities. Each Activity discussion has three parts, as follows:

- Input Information, which lists information that is to be used by the Activity, and its source
- Description, which details the actions to be performed
- 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).

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 Use of this standard

To facilitate the understanding and use of a standard of this magnitude, the following provides additional information.

1.5.1 Applicability

This standard applies to software development and maintenance projects.

This standard can be applied to any software, such as: scientific, business, commercial, educational, military or any other 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 in this case 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 clause 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 those subcontractors.

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, requires specific expertise and authority within the organization.

After mapping is complete, the Activities in clauses 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 group: 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 group 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:

- a) Those that are performed discretely and are therefore mapped into an SLCM
- b) 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, accomplished 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 called 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 figure 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. 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)].

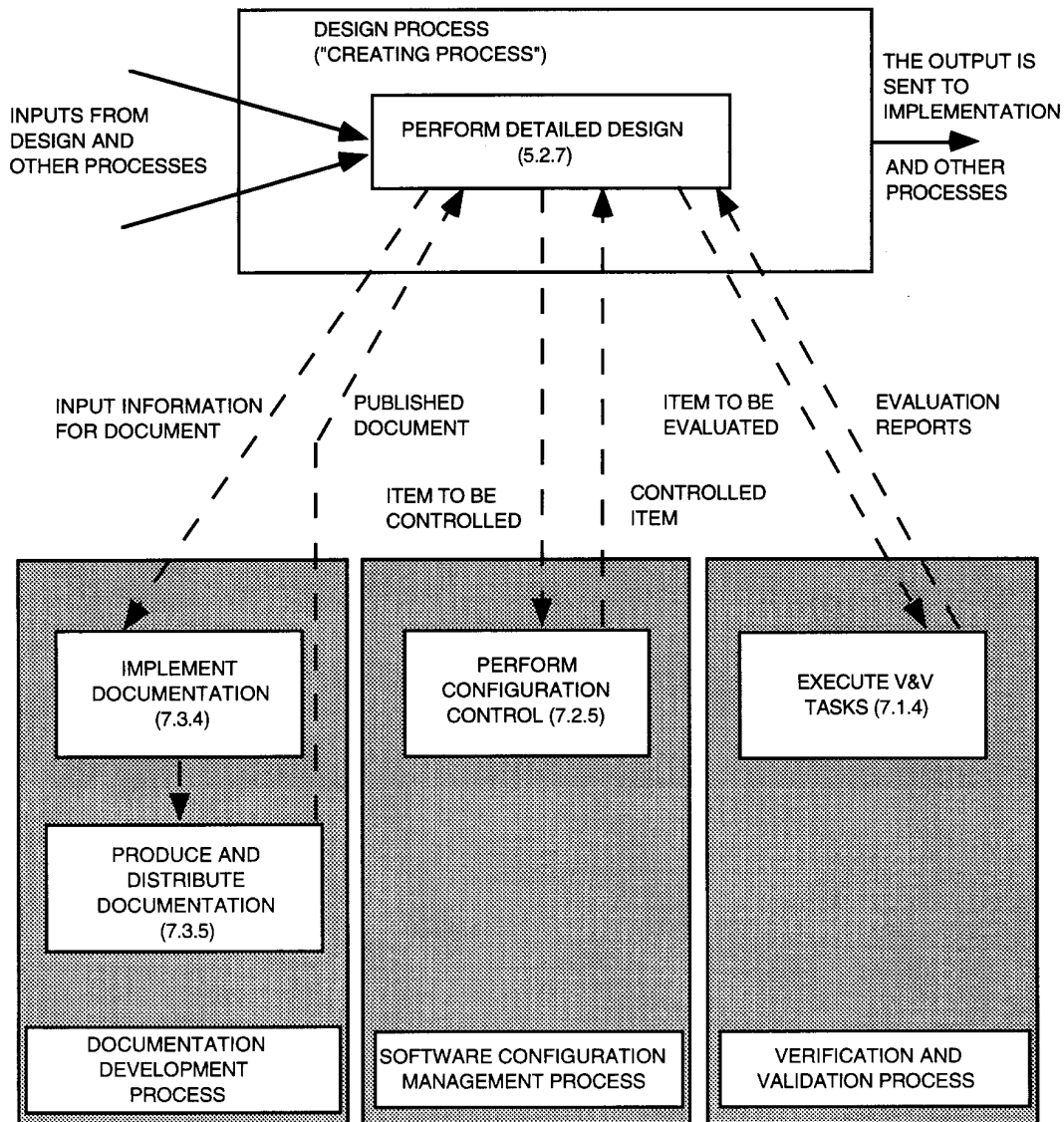


Figure 1 — Example of Invoked Processes

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 clause 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.

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 clause 2 and the mapping Activity in 3.1.3. Examples are given in Annex 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.

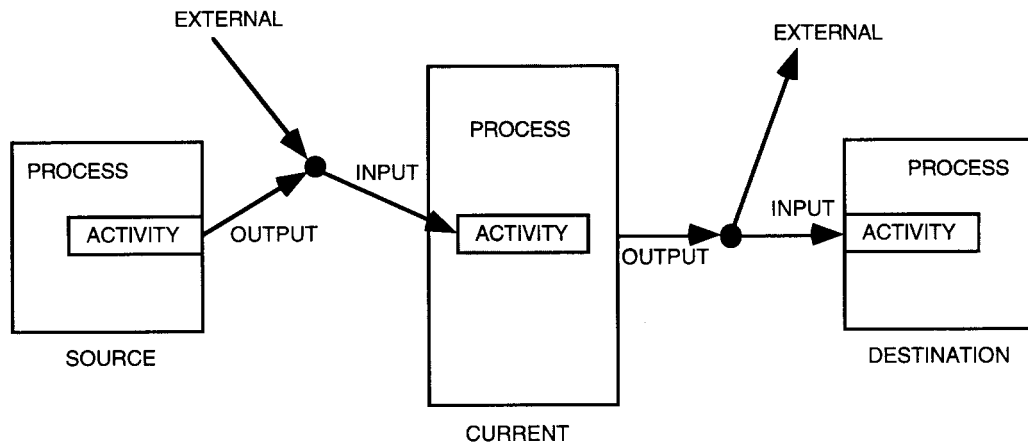


Figure 2 — Information flow

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 framework 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 to 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, as follows:

- a) Mapping the Activities described in this standard into the chosen life cycle (3.1.3)
- b) 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.

Background information necessary for the successful understanding and application of this material is provided in 1.5 . It should be read prior to proceeding further.

2.2 Activities list

- a) Identify Candidate Software Life Cycle Models
- b) Select Project Model

2.3 Identify Candidate Software Life Cycle Models

2.3.1 Input Information

Input Information	Source	
	Process	Activity
Available Software Life Cycle Model(s)	External	
Constraints	External	

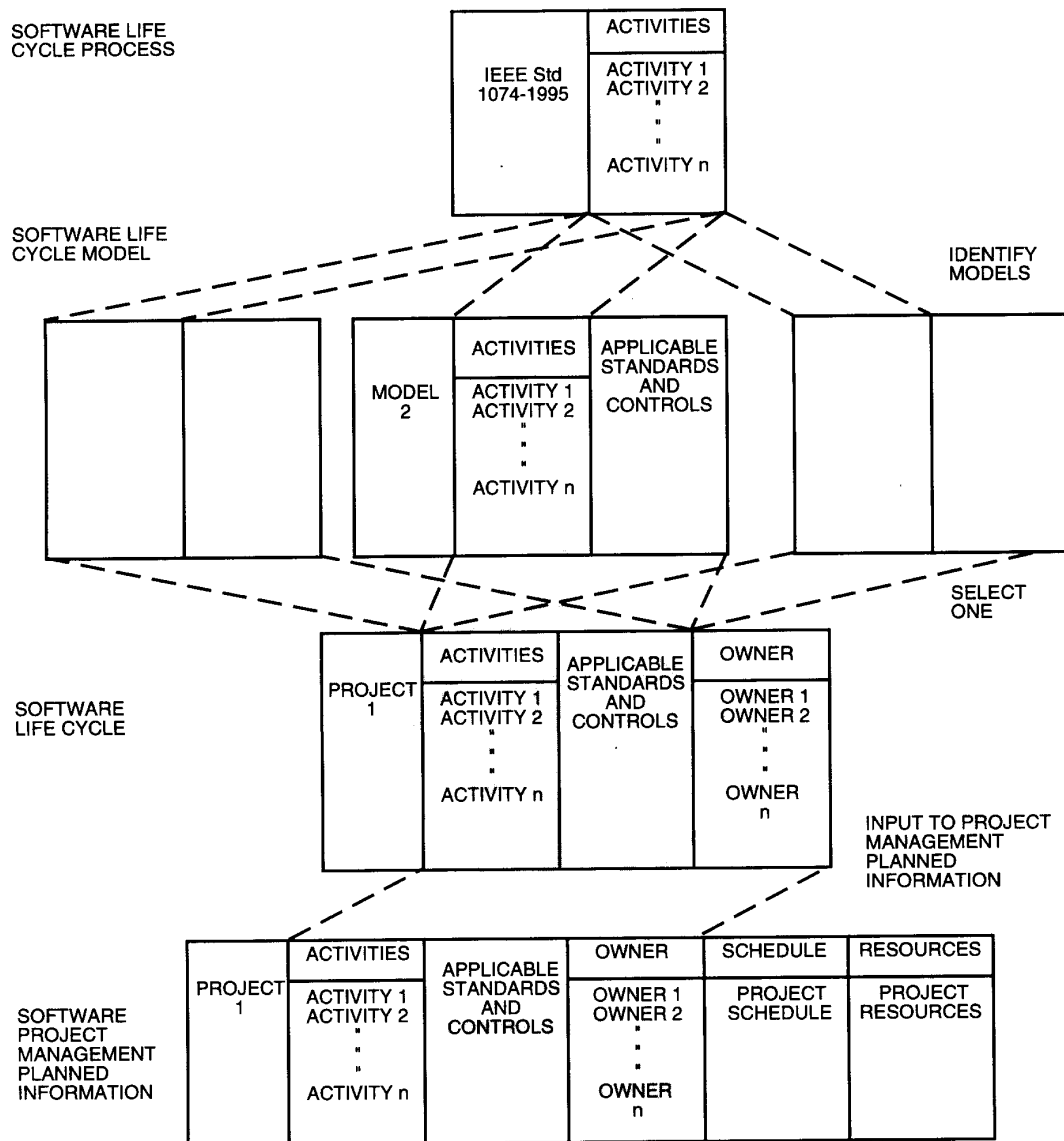


Figure 3 — Software Life Cycle relationships

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 figure 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.

Background information necessary for the successful understanding and application of this material is provided in 1.5. It should be read prior to proceeding further.

3.1.2 Activities list

- a) Map Activities to Software Life Cycle Model
- b) Allocate Project Resources
- c) Establish Project Environment
- d) 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, annex A provides several examples of such mappings. Annex 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 to 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 Requirement	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 Allocation	Project Initiation	Establish Project Environment (3.1.5)
		Plan Project Management (3.1.6)
	Project Monitoring 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 non-automated 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, [B8] 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 (SPMPI) based on the SLC. This Activity shall initially define and subsequently update the SPMPI 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 SPMPIs can be found in [B16].

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

- a) Verification and Validation (7.1.4)
- b) Software Configuration Management (7.2.5)
- c) 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	Manage the Project (3.2.5)
	Retirement	Notify User (6.4.3)
		Conduct Parallel Operations (if applicable) (6.4.4)
		Retire System (6.4.5)
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 (SPMPI).

Background information necessary for the successful understanding and application of this material is provided in 1.5. It should be read prior to proceeding further.

3.2.2 Activities list

- Analyze Risks
- Perform Contingency Planning
- Manage the Project
- Retain Records
- 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, testabil-

ity, 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 (See 1.5.4.4)	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 SPMPPI, 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&RPI). Possible problem solutions may be suggested by the problem reporter. Problems may be resolved through corrections or enhancements (as defined in the PR&RPI). 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 ensure 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:

- What the anomalies are
- Source and cause of product or process problem
- Product(s) or process(es) presumed to contain the error, including documentation
- Problem severity
- 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, determine (according to the PR&RPI) that the implementation of the solution by the responsible Process has been completed, and 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 [B14].

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).

Background information necessary for the successful understanding and application of this material is provided in 1.5. It should be read prior to proceeding further.

3.3.2 Activities list

- a) Plan Software Quality Management
- b) Define Metrics
- c) Manage Software Quality
- d) 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 [B2] .

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

- a) Verification and Validation (7.1.4)
- b) Software Configuration Management (7.2.5)
- c) 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 [B6], [B7], [B15], and [B18].

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)

3.3.5 Manage Software Quality

3.3.5.1 Input Information

Input Information	Source	
	Process	Activity
Report Log	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)
Resolved Problem Reported Information	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)
Software Quality Management Planned Information	Software Quality Management	Plan Software Quality Management (3.3.3)
Defined Metrics	Software Quality Management	Define Metrics (3.3.4)
Collection and Analysis Methods	Software Quality Management	Define Metrics (3.3.4)
Quality Improvement Recommendations	Software Quality Management	Identify Quality Improvement Needs (3.3.6)
Analysis Reported Information	Verification and Validation	Collect and Analyze Metric Data (7.1.5)
Post-Operation Review Reported Information	Retirement	Retire System (6.4.5)

3.3.5.2 Description

Using the listed Input Information, this Activity implements the provisions of the Software Quality Management Planned Information. Based on the Software Quality Management Planned Information quality objectives and milestones, progress shall be measured and reported in Project Quality Assessments.

3.3.5.3 Output Information

Output Information	Destination	
	Process	Activity
Project Quality Assessments	Verification and Validation	Execute Verification and Validation Tasks (7.1.4)

3.3.6 Identify Quality Improvement needs**3.3.6.1 Input Information**

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Software Quality Management Planned Information	Software Quality Management	Plan Software Quality Management (3.3.3)
Software Verification and Validation Planned Information	Verification and Validation	Plan Verification and Validation (7.1.3)
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)
Training Planned Information(s)	Training	Plan Training Program (7.4.3)

3.3.6.2 Description

This Activity identifies needs for quality improvements and outputs the Quality Improvement Recommendations in accordance with the Software Quality Management Planned Information. This is accomplished by using the Input Information. These recommendations shall include their impact on the quality of the soft-

ware delivered. In addition, applicable tools, techniques, and methods for implementation of these recommendations should be identified.

3.3.6.3 Output Information

Output Information	Destination	
	Process	Activity
Quality Improvement Recommendations	Project Monitoring and Control	Manage the Project (3.2.5)
	Software Quality Management	Manage Software Quality (3.3.5)
	External	

4. Pre-development Processes

These are the Processes that must be performed before software development can begin.

4.1 Concept Exploration Process

4.1.1 Overview

A development effort is initiated with the identification of an idea or need for a system to be developed, whether it is a new effort or a change to all or part of an existing application. The Concept Exploration Process examines the requirements at the system level, producing a Statement of Need that initiates the System Allocation or Requirements Process. The Concept Exploration Process includes the identification of an idea or need, its evaluation and refinement, and, once boundaries are placed around it, generation of a Statement of Need for developing a system.

Background information necessary for the successful understanding and application of this material is provided in 1.5. It should be read prior to proceeding further.

4.1.2 Activities list

- a) Identify Ideas or Needs
- b) Formulate Potential Approaches
- c) Conduct Feasibility Studies
- d) Plan System Transition (if applicable)
- e) Refine and Finalize the Idea or Need