IEEE Std 1074-1995
(Revision of IEEE Std 1074-1991)

IEEE Standard for Developing Software Life Cycle Processes

Sponsor

Software Engineering Standards Committee of the IEEE Computer Society

Approved September 21, 1995

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 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

ISBN 1-55937-588-4

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

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying all patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

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 Antczac
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 Pellegrin
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 Gershkovitch
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 Pelegrin
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 Gershkovitch
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*

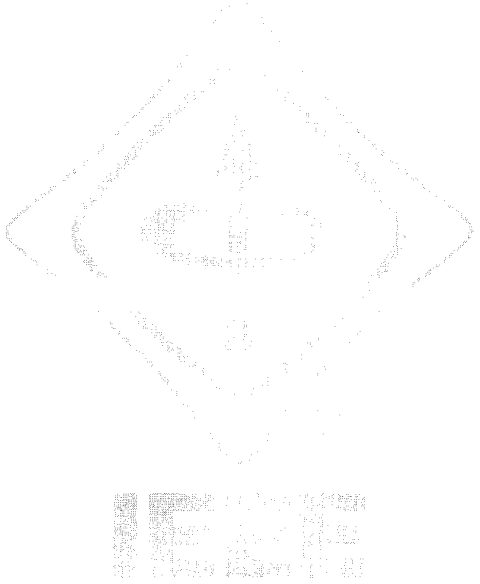
Gilles A. Baril
Clyde R. Camp
Joseph A. Cannatelli
Stephen L. Diamond
Harold E. Epstein
Donald C. Fleckenstein
Jay Forster*
Donald N. Heirman
Richard J. Holleman
Jim Isaak
Ben C. Johnson
Sonny Kasturi
Lorraine C. Kevra
Ivor N. Knight
Joseph L. Koepfinger*
D. N. "Jim" Logothetis
L. Bruce McClung
Marco W. Migliaro
Mary Lou Padgett
John W. Pope
Arthur K. Reilly
Gary S. Robinson
Ingo Rüsch
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
Chester C. Taylor

Angela M. Girardi
IEEE Standards Project Editor



Contents

ANNEXES AND INDEX PAGE

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].

[1] The 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:

a) Input Information, which lists information that is to be used by the Activity, and its source
b) Description, which details the actions to be performed
c) 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 mandatory[2] 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).

[2]The term *mandatory* as used in this standard is synonymous with the term *essential.*
[3]The 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)].

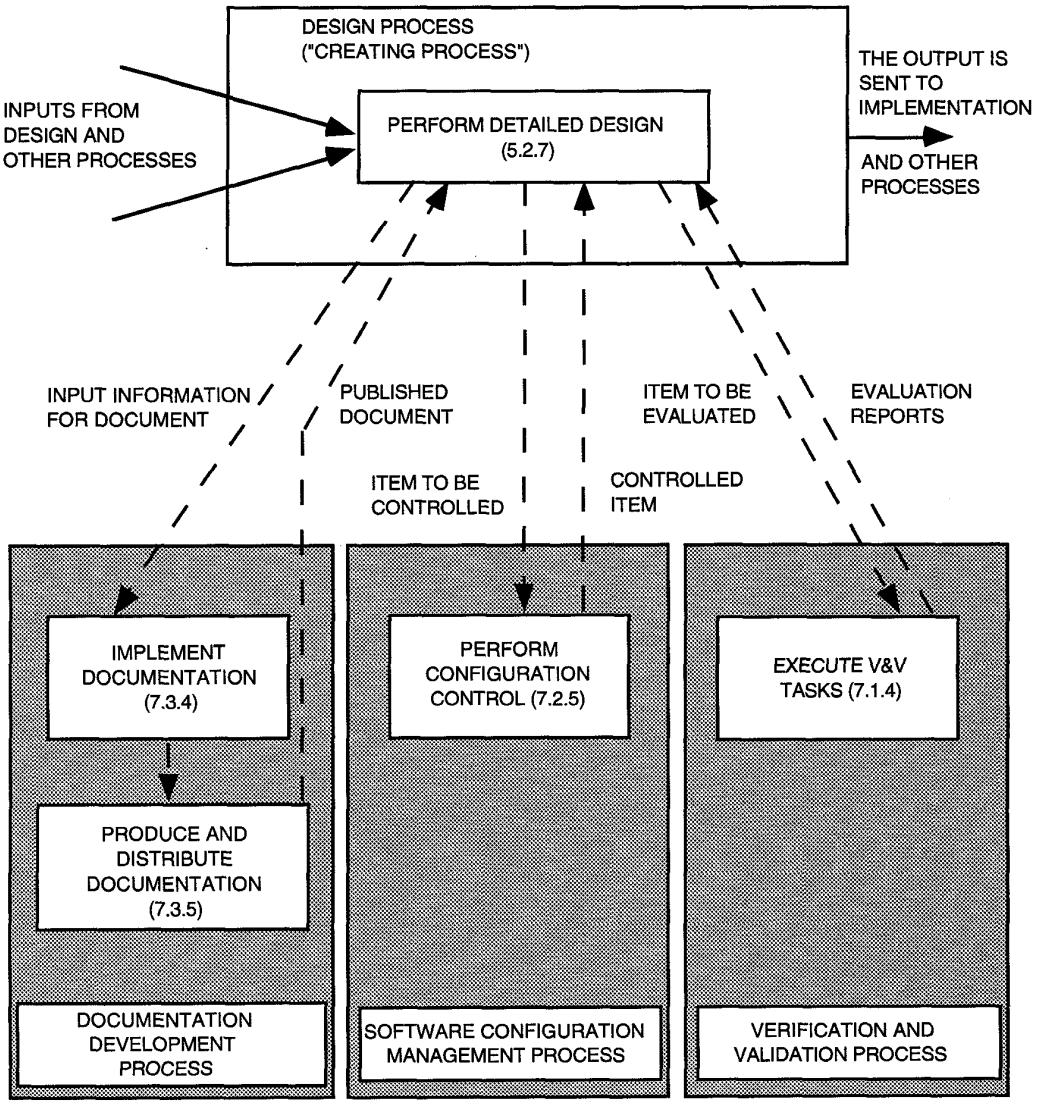


Figure 1—Example of Invoked Processes

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.

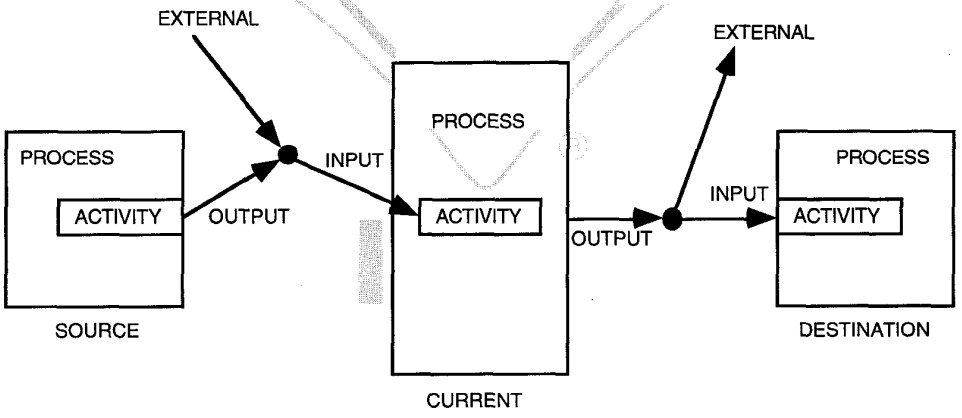


Figure 2—Information flow

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	

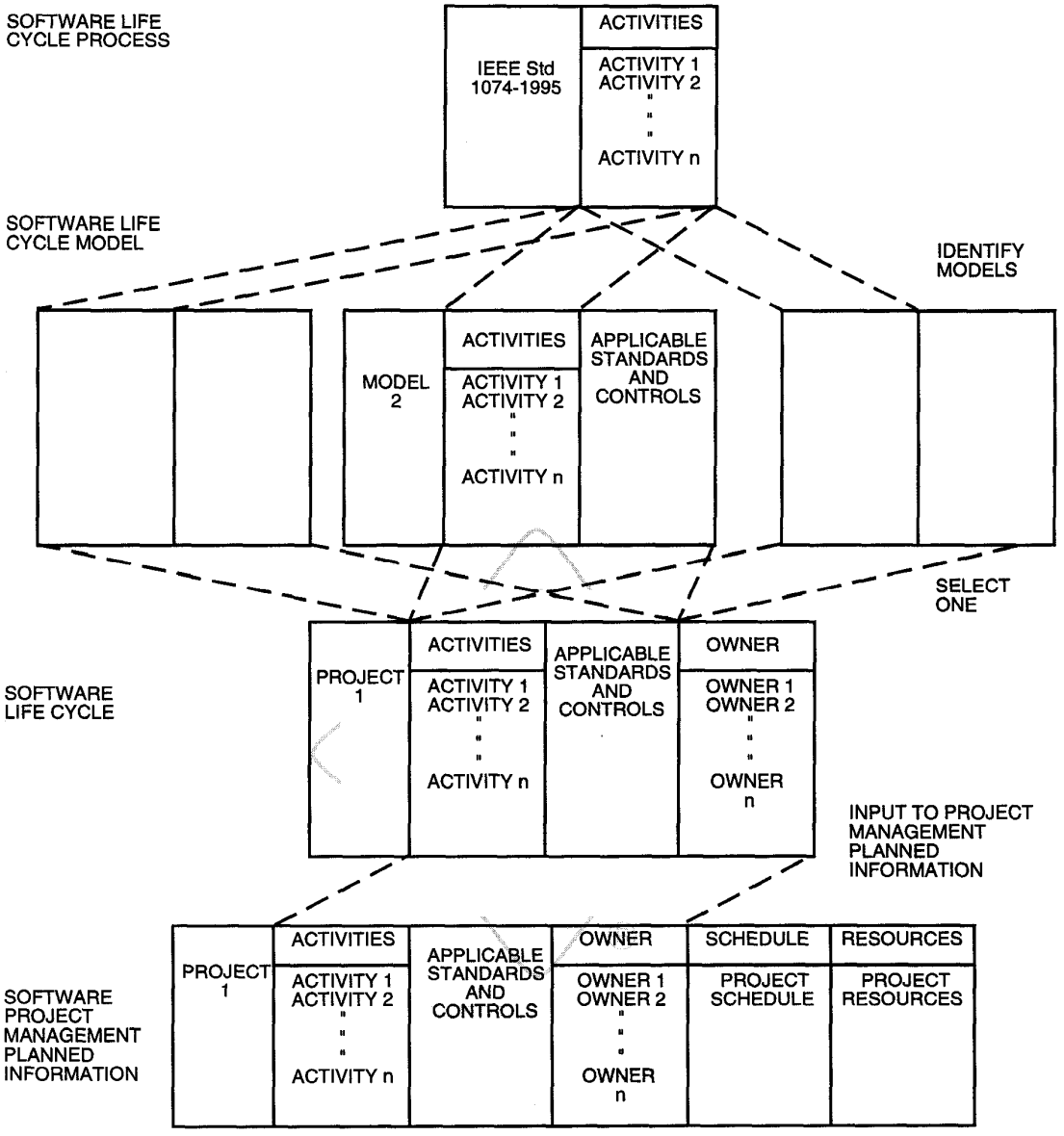


Figure 3—Software Life Cycle relationships

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 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 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

a) Analyze Risks
b) Perform Contingency Planning
c) Manage the Project
d) Retain Records
e) 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 (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 SPMPI, 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:

a) What the anomalies are
b) Source and cause of product or process problem
c) Product(s) or process(es) presumed to contain the error, including documentation
d) Problem severity
e) Course of corrective action

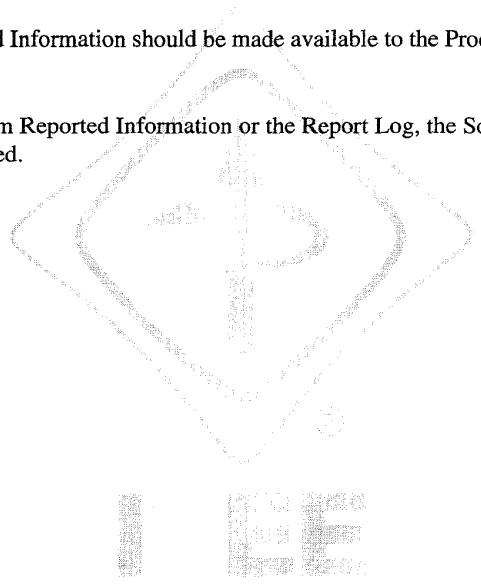
Problem reports that originate from an Activity not included in this standard are noted as resolved within this Activity and forwarded for appropriate action to the responsible authority.

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 software 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

4.1.3 Identify Ideas or Needs

4.1.3.1 Input Information

Input Information	Source	
	Process	Activity
Changing Software Requirements	External	
Customer Requests	External	
Ideas from Within the Development Organization	External	
Marketing Information Sources	External	
User Requests	External	
Enhancement Problem Reported Information	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)
Maintenance Recommendations	Maintenance	Reapply Software Life Cycle (6.3.3)
Feedback Data	Operation and Support	Operate the System (6.2.3)

4.1.3.2 Description

An idea or a need for a new or modified system is generated from one or more of the sources identified in the table above. Input Information to the Preliminary Statement of Need shall be documented, outlining function and performance needs. Changing Software Requirements may come from legislation, regulations, national and international standards, maintenance, etc.

Prior to distribution of the Preliminary Statement of Need to other Activities, the Verification and Validation Process (7.1.4) may be invoked.

4.1.3.3 Output Information

Output Information	Destination	
	Process	Activity
Preliminary Statement of Need	Project Initiation	Plan Project Management (3.1.6)
	Concept Exploration	Formulate Potential Approaches (4.1.4)
		Conduct Feasibility Studies (4.1.5)
		Plan System Transition (if applicable) (4.1.6)
		Refine and Finalize the Idea or Need (4.1.7)

4.1.4 Formulate Potential Approaches

4.1.4.1 Input Information

Input Information	Source	
	Process	Activity
Development Resources and Budget	External	
Market Availability Data	External	
Resource Information	External	
Preliminary Statement of Need	Concept Exploration	Identify Ideas or Needs (4.1.3)

4.1.4.2 Description

Using Resource Information, budget data, and availability of third party software products, Potential Approaches shall be developed based upon the Preliminary Statement of Need and any data pertinent to the decision to develop or acquire the system. The Formulate Potential Approaches Activity shall also produce the constraints and benefits with regard to development of the software. The Constraints and Benefits should include all aspects of the life cycle.

Prior to release of Constraints and Benefits and Potential Approaches, the following Processes may be invoked:

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)

4.1.4.3 Output Information

Output Information	Destination	
	Process	Activity
Constraints and Benefits	Concept Exploration	Conduct Feasibility Studies (4.1.5)
		Refine and Finalize the Idea or Need (4.1.7)
Potential Approaches	Concept Exploration	Conduct Feasibility Studies (4.1.5)
		Refine and Finalize the Idea or Need (4.1.7)

4.1.5 Conduct feasibility studies

4.1.5.1 Input Information

Input Information	Source	
	Process	Activity
Preliminary Statement of Need	Concept Exploration	Identify Ideas or Needs (4.1.3)
Constraints and Benefits	Concept Exploration	Formulate Potential Approaches (4.1.4)
Potential Approaches	Concept Exploration	Formulate Potential Approaches (4.1.4)

4.1.5.2 Description

The feasibility study shall include the analysis of the idea or need, Potential Approaches, and all life cycle Constraints and Benefits. Modeling and prototyping techniques may be considered. In conducting the feasibility study, there may be a need to decide whether to make or buy the system, in part or in total. Justification for each Recommendation shall be fully documented and formally approved by all concerned organizations (including the user and the developer).

Prior to the distribution of the Recommendations, the Verification and Validation Process (7.1.4) may be invoked.

4.1.5.3 Output Information

Output Information	Destination	
	Process	Activity
Recommendations	Project Initiation	Plan Project Management (3.1.6)
	Concept Exploration	Plan System Transition (if applicable) (4.1.6)
		Refine and Finalize the Idea or Need (4.1.7)
	System Allocation	Analyze Functions (4.2.3)

4.1.6 Plan System Transition (if applicable)

4.1.6.1 Input Information

Input Information	Source	
	Process	Activity
Retirement Planned Information (for the system being replaced)	External	
Preliminary Statement of Need	Concept Exploration	Identify Ideas or Needs (4.1.3)
Recommendations	Concept Exploration	Conduct Feasibility Studies (4.1.5)

4.1.6.2 Description

This Activity is applicable only when an existing system (automated or manual) is being replaced with a new system. The transition shall be planned and documented in accordance with the Retirement Planned Information of the system being replaced, Preliminary Statement of Need, and recommended solutions. Transition strategies and tools shall be part of the Transition Planned Information. A Transition Impact Statement shall also be produced.

Prior to distribution of the Transition Planned Information, the following Processes may be invoked:

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)
c) Documentation Development (7.3.4)

4.1.6.3 Output Information

Output Information	Destination	
	Process	Activity
Transition Impact Statement	Project Monitoring and Control	Analyze Risks (3.2.3)
Transition Planned Information	Concept Exploration	Refine and Finalize the Idea or Need (4.1.7)
	Installation	Plan Installation (6.1.3)

4.1.7 Refine and finalize the Idea or Need

4.1.7.1 Input Information

Input Information	Source	
	Process	Activity
Preliminary Statement of Need	Concept Exploration	Identify Ideas or Needs (4.1.3)
Constraints and Benefits	Concept Exploration	Formulate Potential Approaches (4.1.4)
Potential Approaches	Concept Exploration	Formulate Potential Approaches (4.1.4)
Recommendations	Concept Exploration	Conduct Feasibility Studies (4.1.5)
Transition Planned Information (if applicable)	Concept Exploration	Plan System Transition (if applicable) (4.1.6)

4.1.7.2 Description

The idea or need shall be refined by analyzing the Preliminary Statement of Need, the Potential Approaches, Recommendations, and Transition Planned Information (if applicable). An approach shall be selected and documented that refines the initial idea or need.

Based upon the refined ideas or needs, a Statement of Need shall be generated that identifies the software idea, need, or desire, the recommended approach for its implementation, and any data pertinent to a management decision concerning the initiation of the described development effort.

Prior to distribution of the Statement of Need, the following Processes may be invoked:

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)
c) Documentation Development (7.3.4)

4.1.7.3 Output Information

Output Information	Destination	
	Process	Activity
Statement of Need	Project Initiation	Map Activities Software Life Cycle Model (3.1.3)
		Allocate Project Resources (3.1.4)
		Establish Project Environment (3.1.5)
		Plan Project Management (3.1.6)
	Project Monitoring and Control	Analyze Risks (3.2.3)
	System Allocation	Analyze Functions (4.2.3)
		Develop System Architecture (4.2.4)

4.2 System Allocation Process

4.2.1 Overview

The System Allocation Process is the bridge between Concept Exploration and the definition of software requirements. This Process maps the required functions to software and hardware.

The Statement of Need forms the basis for the analysis of the system, resulting in system requirements. This definition determines the inputs to the system, the processing to be applied to the inputs, and the required outputs. The software and hardware operational functions are also identified in these definitions.

The architecture of the system must be developed during the System Allocation Process. The system functions are derived from system requirements, and the hardware, software, and operational requirements are identified. These requirements are analyzed to produce System Functional Software Requirements and System Functional Hardware Requirements. The hardware, software, and operational interfaces must be defined and closely monitored. The hardware requirements analysis is not discussed in this document since it is beyond the scope of this standard.

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.

4.2.2 Activities list

a) Analyze Functions
b) Develop System Architecture
c) Decompose System Requirements

4.2.3 Analyze Functions

4.2.3.1 Input Information

Input Information	Source	
	Process	Activity
Recommendations	Concept Exploration	Conduct Feasibility Studies (4.1.5)
Statement of Need	Concept Exploration	Refine and Finalize the Idea or Need (4.1.7)

4.2.3.2 Description

The Statement of Need and Recommendations for solution shall be analyzed to identify the functions of the total system. Once the functions have been defined, they are delineated in the Functional Description of the System and used to develop the system architecture and identify the hardware and software functions.

Prior to the distribution of the Functional Description of the System, the following Processes shall be invoked:

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)

4.2.3.3 Output Information

Output Information	Destination	
	Process	Activity
Functional Description of the System	System Allocation	Develop System Architecture (4.2.4)
		Decompose System Requirements (4.2.5)
	Requirements	Define Interface Requirements (5.1.4)

4.2.4 Develop System Architecture

4.2.4.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Statement of Need	Concept Exploration	Refine and Finalize the Idea or Need (4.1.7)
Functional Description of the System	System Allocation	Analyze Functions (4.2.3)

4.2.4.2 Description

The Statement of Need and the Functional Description of the System shall be transformed into the System Architecture, using the methodology, standards, and tools established by the organization. The System Architecture becomes the basis for the Design Process and the determination of the hardware and software functions.

4.2.4.3 Output Information

Output Information	Destination	
	Process	Activity
System Architecture	System Allocation	Decompose System Requirements (4.2.5)
	Design	Perform Architectural Design (5.2.3)

4.2.5 Decompose System Requirements

4.2.5.1 Input Information

Input Information	Source	
	Process	Activity
Functional Description of the System	System Allocation	Analyze Functions (4.2.3)
System Architecture	System Allocation	Develop System Architecture (4.2.4)

4.2.5.2 Description

The system functions documented in the Functional Description of the System shall be divided according to the System Architecture to form software requirements, hardware requirements, and the system interfaces. The System Interface Requirements define the interfaces that are external to the system and the interfaces between configuration items that comprise the system. Note that the hardware requirements go to an external destination since they are beyond the scope of this standard. The decomposition of the system may result in requirements for more than one project. Each software project shall be managed individually.

Prior to distribution of the requirements produced by this Activity, 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)
d) Training (7.4.4)

4.2.5.3 Output Information

Output Information	Destination	
	Process	Activity
System Functional Hardware Requirements	External	
System Functional Software Requirements	Project Initiation	Allocate Resources (3.1.4)
	Requirements	Define and Develop Software Requirements (5.1.3)
		Define Interface Requirements (5.1.4)
System Interface Requirements (if applicable)	Requirements	Define and Develop Software Requirements (5.1.3)
		Define Interface Requirements (5.1.4)
	External	

5. Development Processes

These are the Processes that must be performed during the development of a software product.

5.1 Requirements Process

5.1.1 Overview

This Process includes those Activities directed toward the development of software requirements. In the development of a system containing both hardware and software components, the Requirements Process follows the development of total system requirements, and the functional allocation of those system requirements to hardware and software. For a system involving only software development, this effort begins once the Statement of Need is completed.

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.

5.1.2 Activities list

a) Define and Develop Software Requirements
b) Define Interface Requirements
c) Prioritize and Integrate Software Requirements

5.1.3 Define and Develop Software Requirements

5.1.3.1 Input Information

Input Information	Source	
	Process	Activity
Installation Support Requirements	External	
System Constraints	External	
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Analysis of Risks	Project Monitoring and Control	Analyze Risks (3.2.3)
System Functional Software Requirements	System Allocation	Decompose System Requirements (4.2.5)
System Interface Requirements (if applicable)	System Allocation	Decompose System Requirements (4.2.5)

5.1.3.2 Description

The first Activity in this Process, defining the software requirements, is iterative in nature. Whether the software development constitutes the entire project or is part of a system (hardware and software), software requirements, including constraints, shall be generated from Input Information documents and the results of modeling, prototyping, or other techniques.

Using the above Input Information, the developer shall analyze the software requirements to determine traceability, clarity, validity, testability, safety, and any other project-specific characteristics. The use of a comprehensive methodology is recommended to ensure that requirements are complete and consistent. Techniques such as structured analysis, modeling, prototyping, or transaction analysis are helpful in this Activity. When needed, the requirements for a data base shall be included in the requirements.

The Preliminary Software Requirements shall include consideration of System Constraints such as timing, sizing, language, marketing restrictions, and technology.

Further information related to this Activity may be found in [B5].

Prior to the distribution of the Preliminary Software Requirements and Installation Requirements, 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)

5.1.3.3 Output Information

Output Information	Destination	
	Process	Activity
Preliminary Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)
		Define Interface Requirements (5.1.4)
	Verification and Validation	Plan Testing (7.1.6)
Installation Requirements	Installation	Plan Installation (6.1.3)

5.1.4 Define Interface Requirements

5.1.4.1 Input Information

Input Information	Source	
	Process	Activity
System Constraints	External	
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Preliminary Software Requirements	Requirements	Define and Develop Software Requirements (5.1.3)
Functional Description of the System	System Allocation	Analyze Functions (4.2.3)
System Functional Software Requirements	System Allocation	Decompose System Requirements (4.2.5)
System Interface Requirements (if applicable)	System Allocation	Decompose System Requirements (4.2.5)

5.1.4.2 Description

All user, software, and hardware interfaces shall be defined using the applicable Input Information. These interfaces shall be defined either as requirements or as constraints and shall be reviewed by all involved parties.

The user interface is critical in determining the usability of the system. The user interface definition shall specify not only the information flow between the user and the system but also how a user goes about using the system. For a complex interactive system, user interface definition may be a separate document.

The Software Interface Requirements shall specify all software interfaces required to support the development and execution of the software system. Software interfaces may be affected by System Constraints including operating system, data base management system, language compiler, tools, utilities, network protocol drivers, and hardware interfaces.

Prior to the distribution of the Output 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)

5.1.4.3 Output Information

Output Information	Destination	
	Process	Activity
Software Interface Requirements	Project Monitoring and Control	Analyze Risks (3.2.3)
	Requirements	Prioritize and Integrate Software Requirements (5.1.5)
	Design	Design Interfaces (5.2.5)
	Implementation	Create Operating Documentation (5.3.6)

5.1.5 Prioritize and integrate Software Requirements

5.1.5.1 Input Information

Input Information	Source	
	Process	Activity
Preliminary Software Requirements	Requirements	Define and Develop Software Requirements (5.1.3)
Software Interface Requirements	Requirements	Define Interface Requirements (5.1.4)

5.1.5.2 Description

The functional and performance requirements shall be reviewed and a prioritized list of requirements shall be produced, addressing any tradeoffs that may be needed. The organization of the emerging Software Requirements shall be reviewed and revised as necessary. While completing the requirements, a particular design shall not be imposed (i.e., design decisions are made in the Design Process). The Software Requirements shall describe the functional, interface, and performance requirements. It shall also define the required operational and support environments. Further information related to this Activity may be found in [B5].

Prior to distribution of the Software Requirements, 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)

5.1.5.3 Output Information

Output Information	Destination	
	Process	Activity
Software Requirements	Project Monitoring and Control	Analyze Risks (3.2.3)
	Design	All Activities (5.2)
	Implementation	Create Test Data (5.3.3)
		Plan Integration (5.3.7)
	Verification and Validation	Plan Testing (7.1.6)
		Develop Test Specification(s) (7.1.7)
	Training	Plan Training Program (7.4.3)

5.2 Design Process

5.2.1 Overview

During the Design Process, major decisions are made that determine the structure of the system. The objective of the Design Process is to develop a coherent, well-organized representation of the software system that meets the Software Requirements.

The Design Process maps the "what to do" of requirements specifications into the "how to do it" of design specifications. At the architectural design level, the focus is on the functions and structure of the software components that comprise the software system. At the detailed design level, the emphasis is on the data structures and algorithms that are used within each software component.

The Perform Architectural Design and Perform Detailed Design Activities are usually carried out in sequence because detailed design is derived from the architectural design. They differ from each other in the level of design detail. Other Design Process Activities may be carried out in parallel with these Activities.

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.

5.2.2 Activities list

a) Perform Architectural Design
b) Design Data Base (if applicable)
c) Design Interfaces
d) Select or Develop Algorithms
e) Perform Detailed Design

5.2.3 Perform Architectural Design

5.2.3.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
System Architecture	System Allocation	Develop System Architecture (4.2.4)
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)

5.2.3.2 Description

The Perform Architectural Design Activity transforms the Software Requirements and the System Architecture into high-level design concepts. During this Activity the software components constituting the software system and their structures are identified. Purchased software and the contents of the software libraries (as referenced in the SPMPI) may influence the architectural design. Techniques such as modeling and prototyping may be used to evaluate alternative designs if called for in the SPMPI.

By the end of the Perform Architectural Design Activity, the design description of each of the software components shall have been completed. The data, relationships, and constraints shall be specified. In addition, all internal interfaces (among components) shall be defined. This Activity shall create the Software Architectural Design Description.

Prior to distribution of the Software Architectural Design Description, 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)

5.2.3.3 Output Information

Output Information	Destination	
	Process	Activity
Software Architectural Design Description	Design	Perform Detailed Design (5.2.7)

5.2.4 Design Data Base (if applicable)

5.2.4.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)

5.2.4.2 Description

The Design Data Base Activity applies when a data base is to be created as a part of the project. This Activity shall specify the information structure outlined in the Software Requirements and its characteristics within the software system. The Design Data Base Activity involves three separate but dependent steps: conceptual data base design, logical data base design, and physical data base design. Techniques such as data dictionary, data base optimization, and data modeling may be considered. Requirements are molded into an external schema that describes data entities, attributes, relationships, and constraints. The various external schemas are integrated into a single conceptual schema. The conceptual schema is then mapped into an implementation-dependent logical schema. Finally, the physical data structures and access paths are defined. The result of this Activity is to generate the Data Base Description.

Prior to distribution of the Data Base Description, 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)

5.2.4.3 Output Information

Output Information	Destination	
	Process	Activity
Data Base Description	Design	Perform Detailed Design (5.2.7)

5.2.5 Design Interfaces

5.2.5.1 Input Information

Input Information	Source	
	Process	Activity
Software Interface Requirements	Requirements	Define Interface Requirements (5.1.4)
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)

5.2.5.2 Description

The Design Interfaces Activity shall be concerned with the interfaces of the software system contained in the Software Requirements and Software Interface Requirements. This Activity shall consolidate these interface descriptions into a single Interface Description of the software system.

5.2.5.3 Output Information

Output Information	Destination	
	Process	Activity
Interface Description	Design	Perform Detailed Design (5.2.7)

5.2.6 Select or develop Algorithms

5.2.6.1 Input Information

Input Information	Source	
	Process	Activity
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)

5.2.6.2 Description

This Activity is concerned with selecting or developing a procedural representation of the functions specified in the Software Requirements for each software component and data structure. The Algorithms shall completely satisfy the applicable functional and/or mathematical specifications. To the extent possible, the use of existing algorithms should be considered.

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

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)

5.2.6.3 Output Information

Output Information	Destination	
	Process	Activity
Algorithm Descriptions	Design	Perform Detailed Design (5.2.7)

5.2.7 Perform Detailed Design

5.2.7.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)
Software Architectural Design Description	Design	Perform Architectural Design (5.2.3)
Data Base Description (if applicable)	Design	Design Data Base (if applicable) (5.2.4)
Interface Description	Design	Design Interfaces (5.2.5)
Algorithm Descriptions	Design	Select or Develop Algorithms (5.2.6)

5.2.7.2 Description

In the Perform Detailed Design Activity, design alternatives shall be chosen for implementing the functions specified for each software component. By the end of this Activity, the data structure, algorithm, and control information of each software component shall be specified. The Software Design Description (SDD) contains the consolidated data for all of the above Input Information. The details of the interfaces shall be identified within the SDD.

For further information on this topic, see [B11].

Prior to distribution of the SDD, 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)

5.2.7.3 Output Information

Output Information	Destination	
	Process	Activity
Software Design Description	Project Monitoring and Control	Analyze Risks (3.2.3)
	Implementation	Create Test Data (5.3.3)
		Create Source (5.3.4)
		Create Operating Documentation (5.3.6)
		Plan Integration (5.3.7)
	Verification and Validation	Plan Testing (7.1.6)
		Develop Test Specification(s) (7.1.7)
	Training	Develop Training Materials (7.4.4)

5.3 Implementation Process

5.3.1 Overview

The Activities completed during the Implementation Process result in the transformation of the Detailed Design representation of a software product into a programming language realization. This Process produces the source code, data base (if applicable), and the documentation constituting the physical manifestation of the design. In addition, the code and data base are integrated. Care must also be taken during the Implementation Process to apply the appropriate coding standards.

The output of this Process must be the subject of all subsequent testing and validation. The code and data base, along with documentation produced during previous Processes, are the first complete representation of the software product.

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.

5.3.2 Activities list

a) Create Test Data
b) Create Source
c) Generate Object Code
d) Create Operating Documentation
e) Plan Integration
f) Perform Integration

5.3.3 Create Test Data

5.3.3.1 Input Information

Input Information	Source	
	Process	Activity
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)
Software Design Description	Design	Perform Detailed Design (5.2.7)
Source Code (if applicable)	Implementation	Create Source (5.3.4)
Data Base (if applicable)	Implementation	Create Source (5.3.4)
Test Planned Information(s)	Verification and Validation	Plan Testing (7.1.6)
Test Requirements	Verification and Validation	Develop Test Requirements (7.1.7)

5.3.3.2 Description

Using the Software Requirements, the Software Design Description (SDD), and the Source Code (when required), Test Data shall be generated. The Test Planned Information(s) describe the test environment. Test Requirements define the type of test data to be used. To support the testing effort, test Stubs and Drivers may be generated at this time for each item to be tested. The test drivers allow the execution of software tests on an individual or integrated basis. Test Data may be loaded for use in testing the Data Base.

Further information may be found in [B9].

5.3.3.3 Output Information

Output Information	Destination	
	Process	Activity
Stubs and Drivers (if applicable)	Implementation	Perform Integration (5.3.8)
Test Data	Verification and Validation	Execute the Tests (7.1.8)

5.3.4 Create Source

5.3.4.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Software Design Description	Design	Perform Detailed Design (5.2.7)

5.3.4.2 Description

The Source Code, including suitable comments, shall be generated using the project environment, as found in the Software Project Management Planned Information (SPMPI) and the Software Design Description. If the software requires a Data Base, then the Data Base utilities may need to be coded. If Source Code is going to be used to create test data, the Source Code shall be made available to the Create Test Data Activity (5.3.3).

5.3.4.3 Output Information

Output Information	Destination	
	Process	Activity
Data Base (if applicable)	Implementation	Create Test Data (5.3.3)
		Generate Object Code (5.3.5)
Source Code (if applicable)	Implementation	Create Test Data (5.3.3)
Source Code	Implementation	Generate Object Code (5.3.5)

5.3.5 Generate Object Code

5.3.5.1 Input Information

Input Information	Source	
	Process	Activity
Data Base (if applicable)	Implementation	Create Source (5.3.4)
Source Code	Implementation	Create Source (5.3.4)

5.3.5.2 Description

The code shall be grouped into processable units. (This will be dictated by selected language and design information.) All assembly language units shall be assembled and all high-level language units compiled into Object Code. Syntactically incorrect code, identified by the assembler or compiler output, shall be reworked until the source code can be processed free of syntactical errors. These units shall be debugged. If a Data Base is coded, it too shall be debugged.

Further information may be found in [B9].

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

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)

5.3.5.3 Output Information

Output Information	Destination	
	Process	Activity
Corrected Data Base (if applicable)	Implementation	Perform Integration (5.3.8)
Corrected Source Code	Implementation	Perform Integration (5.3.8)
Object Code	Implementation	Perform Integration (5.3.8)

5.3.6 Create Operating Documentation

5.3.6.1 Input Information

Input Information	Source	
	Process	Activity
Software Design Description	Design	Perform Detailed Design (5.2.7)
Software Interface Requirements	Requirements	Define Interface Requirements (5.1.4)
Documentation Planned Information(s)	Documentation Development	Plan Documentation (7.3.3)

5.3.6.2 Description

This Activity shall produce the software project's operating documentation from the SDD and the Software Interface Requirements in accordance with the Documentation Planned Information. The Operating Documentation is required for installing, operating, and supporting the system throughout the life cycle.

For further information, [B20] may be used.

Prior to distribution of the documents listed below, 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)

5.3.6.3 Output Information

Output Information	Destination	
	Process	Activity
Operating Documentation	Installation	Distribute Software (6.1.4) Plan Installation (6.1.3)

5.3.7 Plan Integration

5.3.7.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)
Software Design Description	Design	Perform Detailed Design (5.2.7)
Test Planned Information(s)	Verification and Validation	Plan Testing (7.1.6)

5.3.7.2 Description

During the Plan Integration Activity, the Software Requirements and the SDD are analyzed to determine the order of combining software components into an overall system. The project environment, as defined in the SPMPI, shall be considered when planning integration. The integration methods shall be documented in the Integration Planned Information. The Integration Planned Information shall be coordinated with the Test Planned Information(s).

Prior to distribution of the Integration 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)

5.3.7.3 Output Information

Output Information	Destination	
	Process	Activity
Integration Planned Information	Project Monitoring and Control	Analyze Risks (3.2.3)
		Manage the Project (3.2.5)
	Implementation	Perform Integration (5.3.8)
	Verification and Validation	Plan Testing (7.1.6)

5.3.8 Perform Integration

5.3.8.1 Input Information

Input Information	Source	
	Process	Activity
Stubs and Drivers (if applicable)	Implementation	Create Test Data (5.3.3)
Corrected Data Base (if applicable)	Implementation	Generate Object Code (5.3.5)
Corrected Source Code	Implementation	Generate Object Code (5.3.5)
Integration Planned Information	Implementation	Plan Integration (5.3.7)
System Components	External	
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Object Code	Implementation	Generate Object Code (5.3.5)
Tested Software	Verification and Validation	Execute the Tests (7.1.8)

5.3.8.2 Description

This Activity shall execute the Integration Planned Information. This is accomplished by appropriately combining the Corrected Data Base, Corrected Source Code, Object Code, and Stubs and Drivers, as specified, into Integrated Software. Other necessary Object Code, from the project environment as defined in the SPMPI, shall also be integrated. If a system includes both hardware and software components, the system integration may be included as part of this Activity.

Prior to software integration and the distribution of the Integrated Software, the following Processes shall be invoked:

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)

5.3.8.3 Output Information

Output Information	Destination	
	Process	Activity
Integrated Software	Verification and Validation	Execute the Tests (7.1.8)

6. Post-development Processes

These are the Processes that must be performed to install, operate, support, maintain, and retire a software product.

6.1 Installation Process

6.1.1 Overview

Installation consists of the transportation and installation of a software system from the development environment to the target environment. It includes the necessary software modifications, checkout in the target environment, and customer acceptance. If a problem arises, it must be identified and reported; if necessary and possible, a temporary "work-around" may be applied.

During the Installation Process, the software to be delivered is installed, operationally checked out, and monitored. This effort culminates in formal customer acceptance. The scheduling of turnover and customer acceptance is defined 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.

6.1.2 Activities list

a) Plan Installation
b) Distribute Software
c) Install Software
d) Accept Software in Operational Environment

6.1.3 Plan Installation

6.1.3.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Installation Requirements	Requirements	Define and Develop Software Requirements (5.1.3)
Transition Planned Information (if applicable)	Concept Exploration	Plan System Transition (4.1.6)
Operating Documentation	Implementation	Create Operating Documentation (5.3.6)

6.1.3.2 Description

The tasks to be performed during installation shall be described in Software Installation Planned Information. The Installation Requirements and the other Input Information shall be analyzed to guide the development of the Software Installation Planned Information. This Planned Information, the associated documentation, and the developed software shall be used to install the software product.

Prior to distribution of the Software Installation 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)
d) Training (7.4.4)

6.1.3.3 Output Information

Output Information	Destination	
	Process	Activity
Software Installation Planned Information	Installation	Distribute Software (6.1.4)

6.1.4 Distribute Software

6.1.4.1 Input Information

Input Information	Source	
	Process	Activity
Operating Documentation	Implementation	Create Operating Documentation (5.3.6)
Software Installation Planned Information	Installation	Plan Installation (6.1.3)
Data Base Data	External	
Tested Software	Verification and Validation	Execute the Tests (7.1.8)
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)

6.1.4.2 Description

During this Activity, the Tested Software with necessary Data Base Data, Operating Documentation, and Software Installation Planned Information shall be packaged onto their respective media as designated in the SPMPI. The Packaged Software is distributed to the appropriate site(s) for installation. The Packaged Installation Planned Information is distributed as appropriate to the site(s) to facilitate the installation efforts. The Packaged Operating Documentation shall be available for operation of the system.

Prior to distribution of the Output 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)

6.1.4.3 Output Information

Output Information	Destination	
	Process	Activity
Packaged Operating Documentation	Operation and Support	Operate the System (6.2.3)
	Installation	Install Software (6.1.5)
Packaged Installation Planned Information	Installation	Install Software (6.1.5)
Packaged Software	Installation	Install Software (6.1.5)

6.1.5 Install Software

6.1.5.1 Input Information

Input Information	Source	
	Process	Activity
Packaged Installation Planned Information	Installation	Distribute Software (6.1.4)
Packaged Operating Documentation	Installation	Distribute Software (6.1.4)
Packaged Software	Installation	Distribute Software (6.1.4)
Data Base Data	External	

6.1.5.2 Description

The Packaged Software and any required Data Base Data shall be installed in the target environment according to the procedures in the Software Installation Planned Information. This may include tailoring by the customer. The Installation Reported Information shall document the installation and any problems encountered.

6.1.5.3 Output Information

Output Information	Destination	
	Process	Activity
Installation Reported Information	Project Monitoring and Control	Manage the Project (3.2.5)
Installed Software	Installation	Accept Software in Operational Environment (6.1.6)

6.1.6 Accept Software in operational environment

6.1.6.1 Input Information

Input Information	Source	
	Process	Activity
User Acceptance Planned Information	External	
Test Summary Reported Information	Verification and Validation	Execute the Tests (7.1.8)
Installed Software	Installation	Install Software (6.1.5)

6.1.6.2 Description

The software acceptance shall consist of analysis of the Test Summary Reported Information(s) according to the User Acceptance Planned Information to assure that the Installed Software performs as expected. When the results of the analysis satisfy the requirements of the User Acceptance Planned Information, the Installed Software System is accepted by the User.

Prior to completion of accepting software in the operational environment, the following Processes should be invoked:

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)

6.1.6.3 Output Information

Output Information	Destination	
	Process	Activity
Customer Acceptance	External	
Installed Software System	Operation and Support	Operate the System (6.2.3)
	Retirement	Conduct Parallel Operations (if applicable) (6.4.4)

6.2 Operation and Support Process

6.2.1 Overview

The Operation and Support Process involves user operation of the system and ongoing support. Support includes providing technical assistance, consulting with the user, and recording user support requests by maintaining a Support Request Log. Thus the Operation and Support Process may trigger Maintenance Activities via the ongoing Project Monitoring and Control Process, which will provide information re-entering the 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.

6.2.2 Activities list

a) Operate the System
b) Provide Technical Assistance and Consulting
c) Maintain Support Request Log

6.2.3 Operate the System

6.2.3.1 Input Information

Input Information	Source	
	Process	Activity
Packaged Operating Documentation	Installation	Distribute Software (6.1.4)
Support Planned Information	Project Initiation	Plan Project Management (3.1.6)
Installed Software System	Installation	Accept Software in Operational Environment (6.1.6)

6.2.3.2 Description

During this Activity, the Installed Software System shall be utilized in the intended environment and in accordance with the operating instructions. Feedback Data are collected for product and documentation improvement and system tuning. The user shall analyze the Feedback Data and identify Anomalies (which include desired enhancements). Anomalies shall be reported.

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

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)

6.2.3.3 Output Information

Output Information	Destination	
	Process	Activity
Anomalies	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)
Operation Logs	External	
Feedback Data	Project Monitoring and Control	Manage the Project (3.2.5)
	Concept Exploration	Identify Ideas or Needs (4.1.3)

6.2.4 Provide technical assistance and consulting

6.2.4.1 Input Information

Input Information	Source	
	Process	Activity
Request for Support	External	
Support Planned Information	Project Initiation	Plan Project Management (3.1.6)

6.2.4.2 Description

This Activity applies after the user has accepted the software. The support function shall include providing responses to the user's technical questions or problems. A Support Response is sent to the Maintain Support Request Log Activity so that feedback can be provided to other Processes.

6.2.4.3 Output Information

Output Information	Destination	
	Process	Activity
Support Response	Operation and Support	Maintain Support Request Log (6.2.5)
	External	

6.2.5 Maintain Support Request Log

6.2.5.1 Input Information

Input Information	Source	
	Process	Activity
Support Planned Information	Project Initiation	Plan Project Management (3.1.6)
Support Response	Operation and Support	Provide Technical Assistance and Consulting (6.2.4)

6.2.5.2 Description

This Activity shall record support requests in the Support Request Log. Methodology regarding management of this Activity shall be identified in the Support Planned Information. Anomalies that are reported shall be reported also to the Project Monitoring and Control Process. Prior to release of the Support Request Log, the Verification and Validation Process (7.1.4) shall be invoked.

6.2.5.3 Output Information

Output Information	Destination	
	Process	Activity
Anomalies	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)
Support Request Log	Verification and Validation	Execute Verification and Validation Tasks (7.1.4)

6.3 Maintenance Process

6.3.1 Overview

The Maintenance Process is concerned with the resolution of software errors, faults, and failures. The requirement for software maintenance initiates software life cycle (SLC) changes. The SLC is remapped and executed, thereby treating the Maintenance Process as iterations of development.

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.

6.3.2 Activities list

a) Reapply Software Life Cycle

6.3.3 Reapply Software Life Cycle

6.3.3.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Correction Problem Reported Information	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)

6.3.3.2 Description

The information provided by the Correction Problem Reported Information and the current Software Project Management Planned Information (SPMPI) shall result in the generation of Maintenance Recommendations. These Maintenance Recommendations will then enter the SLC at the Concept Exploration Process to improve the quality of the software system.

6.3.3.3 Output Information

Output Information	Destination	
	Process	Activity
Maintenance Recommendations	Concept Exploration	Identify Ideas or Needs (4.1.3)

6.4 Retirement Process

6.4.1 Overview

The Retirement Process involves the removal of an existing system from its active support or use either by ceasing its operation or support, or by replacing it with a new system or an upgraded version of the existing 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.

6.4.2 Activities list

a) Notify User
b) Conduct Parallel Operations (if applicable)
c) Retire System

6.4.3 Notify user

6.4.3.1 Input Information

Input Information	Source	
	Process	Activity
Retirement Planned Information	Project Initiation	Plan Project Management (3.1.6)

6.4.3.2 Description

This Activity shall be the formal notification to any user (both internal and external customers) of an operating software system that is to be removed from active support or use. This notification can take any of several forms as appropriate for the individual environment. It is important that all users of the outgoing system be made aware that it will become unsupported. The actual dates of the removal of support are to be clearly specified and must allow time for current users to make whatever arrangements are necessary to respond to this notification. Included in the user notification should be one or more of the following:

a) Description of the replacement system including its date of availability
b) Statement as to why the system is not being supported
c) Description of possible other support

Prior to the distribution of the Official Notification, the Documentation Development Process (7.3.4) shall be invoked.

6.4.3.3 Output Information

Output Information	Destination	
	Process	Activity
Official Notification	Project Monitoring and Control	Retain Records (3.2.6)
	External	

6.4.4 Conduct Parallel Operations (if applicable)

6.4.4.1 Input Information

Input Information	Source	
	Process	Activity
Transition Planned Information (for the replacing system)	External	
Retirement Planned Information	Project Initiation	Plan Project Management (3.1.6)
Installed Software System	Installation	Accept Software in Operational Environment (6.1.6)

6.4.4.2 Description

If the outgoing system is being replaced by a new system, this Activity may apply. This Activity shall involve a period of dual operation utilizing the retiring system for official results, while completing the preparation of the new system for formal operation. It is a period of user training on the new system and validation of the new system. The Retirement Planned Information, as well as the Transition Planned Information, may be used to provide information to conduct parallel operations for the replacing system.

While conducting this Activity, the following Processes shall be invoked:

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)
c) Training (7.4.4)

6.4.4.3 Output Information

Output Information	Destination	
	Process	Activity
Parallel Operations Log	Project Monitoring and Control	Retain Records (3.2.6)

6.4.5 Retire System

6.4.5.1 Input Information

Input Information	Source	
	Process	Activity
Retirement Planned Information	Project Initiation	Plan Project Management (3.1.6)

6.4.5.2 Description

This Activity shall consist of the actual removal and archiving of the retiring system from regular usage according to the Retirement Planned Information. It may be spread over a period of time and take the form of a phased removal, or it may be the simple removal of the entire system from the active software library. Prior to the retirement, users must have been notified of the event. Any preparations for the use of a replacement system should have been completed. The Post-Operation Review Reported Information is generated at this time. The Retire System Activity must be documented in Archive Reported Information.

Prior to the final distribution of the Post-Operation Review Reported Information and Archive Reported 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)

6.4.5.3 Output Information

Output Information	Destination	
	Process	Activity
Post-Operation Review Reported Information	Project Monitoring and Control	Retain Records (3.2.6)
	Software Quality Management	Manage Software Quality (3.3.5)
Archive Reported Information	External	

7. Integral Processes

These are the Processes needed to successfully complete project Activities. These Processes are utilized to ensure the completion and quality of project functions.

7.1 Verification and Validation Process

7.1.1 Overview

The Verification and Validation Process includes planning and performing both Verification and Validation tasks. Verification tasks include reviews, configuration audits, and quality audits. Validation tasks include all phases of testing. These Verification and Validation tasks are conducted throughout the software life cycle (SLC) to ensure that all requirements are satisfied. This Process addresses each life cycle Process and product.

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.

7.1.2 Activities list

a) Plan Verification and Validation
b) Execute Verification and Validation Tasks
c) Collect and Analyze Metric Data
d) Plan Testing
e) Develop Test Requirements
f) Execute the Tests

7.1.3 Plan Verification and Validation

7.1.3.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Analysis of Risks	Project Monitoring and Control	Analyze Risks (3.2.3)
Software Quality Management Planned Information	Software Quality Management	Plan Software Quality Management (3.3.3)

7.1.3.2 Description

This Activity shall be responsive to the Software Project Management Planned Information and the Software Quality Management Planned Information by identifying Processes and Process Output Information to be verified and validated. The purpose and scope of the verification and validation task shall be defined for each Process and all Process Output Information. The planning shall include developing schedules, estimating resources, identifying special resources, staffing, and establishing exit or acceptance criteria. Verification and validation methods to be considered in this planning Activity include audits (e.g., functional and physical configuration, compliance), reviews (e.g., design, code, document), prototyping, inspection, formal proof, analysis, and demonstration. Special attention should be given to minimizing technical risks and verifying requirements traceability. This planning shall be documented in the Software Verification and Validation Planned Information (SVVPI).

Because of the importance of testing in the Verification and Validation Process, this standard addresses testing Activities separately. Test planning and execution may be included in the Verification and Validation Planning and Execution Activities.

Further information on verification and validation planning may be found in [B2], [B3], [B4], [B6], [B7], [B11], [B12], [B13], [B14], and [B15].

Prior to distribution of the SVVPI, 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)

7.1.3.3 Output Information

Output Information	Destination	
	Process	Activity
Software Verification and Validation Planned Information	Software Quality Management	Identify Quality Improvement Needs (3.3.6)
	Verification and Validation	Execute Verification and Validation Tasks (7.1.4)
		Plan Testing (7.1.6)

7.1.4 Execute Verification and Validation tasks

7.1.4.1 Input Information

Input Information	Source	
	Process	Activity
Item(s) to Be Evaluated	Creating Process	
Resolved Problem Reported Information	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)
Project Quality Assessments	Software Quality Management	Manage Software Quality (3.3.5)
Support Request Log	Operation and Support	Maintain Support Request Log (6.2.5)
Software Verification and Validation Planned Information	Verification and Validation	Plan Verification and Validation (7.1.3)
Basis or Bases for Evaluation	External	
	Creating Process	

7.1.4.2 Description

This Activity shall include performing the tasks specified in the SVVPI using the Input Information. Results shall be provided in Evaluation Reported Information. Anomalies identified during the performance of these tasks shall be reported.

Further information related to this Activity may be found in [B2], [B3], [B11], [B12], and [B13].

Prior to distribution of Evaluation Reported Information, the following Processes shall be invoked:

a) Software Configuration Management (7.2.5)
b) Documentation Development (7.3.4)

7.1.4.3 Output Information

Output Information	Destination	
	Process	Activity
Evaluation Reported Information	Project Monitoring and Control	Manage the Project (3.2.5)
	Software Quality Management	Identify Quality Improvement Needs (3.3.6)
	Verification and Validation	Collect and Analyze Metric Data (7.1.5)
	Creating Process	
Anomalies	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)

7.1.5 Collect and Analyze Metric Data

7.1.5.1 Input Information

Input Information	Source	
	Process	Activity
Support Personnel Reported Information	External	
User Input Information	External	
Metric Data	Originating Process	Originating Activity
Correction Problem Reported Information	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)
Enhancement Problem Reported Information	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)
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)
Evaluation Reported Information	Verification and Validation	Execute Verification and Validation Tasks (7.1.4)

7.1.5.2 Description

This Activity collects Evaluation Reported Information, Problem Reported Information, and project-generated Metric Data, as stated in the Software Quality Management Planned Information. The data shall be analyzed using defined methodologies. This Activity shall identify improvements in both quality and requirements as a result of Support Personnel Reported Information and User Input Information. Analysis Reported Information shall be generated describing the results of metrics validation and analysis, defect trend analysis, and the user's view analysis.

Further information related to this Activity may be found in [B6], [B7], [B14], and [B15].

Prior to distribution of the Analysis Reported Information, the following Processes should be invoked:

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)
c) Documentation Development (7.3.4)

7.1.5.3 Output Information

Output Information	Destination	
	Process	Activity
Analysis Reported Information	Project Monitoring and Control	Analyze Risks (3.2.3)
		Perform Contingency Planning (3.2.4)
		Manage the Project (3.2.5)
	Software Quality Management	Manage Software Quality (3.3.5)
		Identify Quality Improvement Needs (3.3.6)

7.1.6 Plan Testing

7.1.6.1 Input Information

Input Information	Source	
	Process	Activity
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Preliminary Software Requirements	Requirements	Define and Develop Software Requirements (5.1.3)
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)
Software Verification and Validation Planned Information	Verification and Validation	Plan Verification and Validation (7.1.3)

7.1.6.2 Description

This Activity shall identify the overall scope, approach, resources, and schedule of the testing tasks over the entire SLC and document them in Test Planned Information(s). The Test Planned Information(s) shall define the generic levels of testing and the basic test environment and structure needed to support required levels of testing. Each Test Planned Information shall identify the items to be tested, the requirements to be tested, and the test pass-or-fail criteria based on the Software Requirements and the Software Design Description (SDD) (as soon as available). The Test Planned Information(s) shall identify test coverage criteria, the tools and approaches being applied, the environmental needs, the testing tasks to be performed, the organizational structure, the management controls and reporting procedures, and the risks and contingencies.

The Test Planned Information(s) shall be coordinated with, and may be combined with, the Integration Planned Information and SVVPI.

Further information related to this Activity may be found in [B4], [B9], [B11], and [B17].

Prior to distribution of the Test Planned Information(s), 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)

7.1.6.3 Output Information

Output Information	Destination	
	Process	Activity
Test Planned Information(s)	Project Monitoring and Control	Analyze Risks (3.2.3)
		Manage the Project (3.2.5)
	Software Quality Management	Identify Quality Improvement Needs (3.3.6)
	Implementation	Create Test Data (5.3.3)
		Plan Integration (5.3.7)
	Verification and Validation	Develop Test Requirements (7.1.7)
		Execute the Tests (7.1.8)

7.1.7 Develop Test Requirements

7.1.7.1 Input Information

Input Information	Source	
	Process	Activity
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)
Software Design Description	Design	Perform Detailed Design (5.2.7)
Test Planned Information(s)	Verification and Validation	Plan Testing (7.1.6)

7.1.7.2 Description

Test Requirements for each generic level of testing shall be developed to refine the test approach from the Test Planned Information(s) to item-specific test procedures used for test execution. The Test Requirements shall define what is to be tested, the data to be used in testing, expected results, the test environment components, and the procedures to be followed in testing. Information from the SRS, the SDD, and the Test Planned Information(s) is used to generate the Test Requirements.

Further information related to this Activity may be found in [B4], [B9], and [B10].

Prior to distribution of the Test Requirements, 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)

7.1.7.3 Output Information

Output Information	Destination	
	Process	Activity
Test Requirements	Implementation	Create Test Data (5.3.3)
	Verification and Validation	Execute the Tests (7.1.8)

7.1.8 Execute the tests

7.1.8.1 Input Information

Input Information	Source	
	Process	Activity
Test Environment Components	External	
Test Data	Implementation	Create Test Data (5.3.3)
Integrated Software	Implementation	Perform Integration (5.3.8)
Test Planned Information(s)	Verification and Validation	Plan Testing (7.1.6)
Test Requirements	Verification and Validation	Develop Test Requirements (7.1.7)

7.1.8.2 Description

This Activity shall configure the Test Environment Components as required by the Test Requirements. Each test shall be conducted on the Integrated Software using Test Data as defined in its associated Test Requirements and in accordance with the Test Planned Information(s).

This Activity could be iterative, with several instances performed during the software's life. Not all Input Information and Output Information are required for a given iteration; the presence of any Input Information is sufficient as an entry criterion, and the creation of any Output Information is a sufficient exit criterion.

Based on comparison of actual results with expected results, according to the pass-fail criteria, a pass-fail determination shall be made and recorded in a test log. Each anomalous event that occurs during execution which requires further investigation shall be reported. The impact on the validity of the test should also be noted.

Test Summary Reported Information shall summarize the results of a test based on its Test Requirements and test log. Tested Software is that software which has successfully passed all tests at the appropriate level and met the specified criteria and requirements. Tested Software may then be further integrated with other software or sent for installation.

Further information related to this Activity may be found in [B4], [B9], and [B10].

Prior to distribution of the Output Information from this Activity, 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)

7.1.8.3 Output Information

Output Information	Destination	
	Process	Activity
Test Summary Reported Information	Project Monitoring and Control	Analyze Risks (3.2.3)
		Manage the Project (3.2.5)
	Installation	Accept Software in Operational Environment (6.1.6)
	External	
Tested Software	Implementation	Perform Integration (5.3.8)
	Installation	Distribute Software (6.1.4)
Anomalies	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)

7.2 Software Configuration Management Process

7.2.1 Overview

Software Configuration Management identifies the items in a software development project and provides both for control of the identified items and for the generation of Status Reported Information for management visibility and accountability throughout the software life cycle (SLC). Items to be managed are those defined in Software Configuration Management Planned Information (SCMPI). Examples to be considered for inclusion in the SCMPI are code, documentation, plans, and specifications. Configuration audits, if required by the Project, should be addressed in the Verification and Validation Process. The Software Configuration Management approach for a given project should be compatible with the Configuration Management approach being used on associated systems.

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.

7.2.2 Activities list

a) Plan Configuration Management
b) Develop Configuration Identification
c) Perform Configuration Control
d) Perform Status Accounting

7.2.3 Plan Configuration Management

7.2.3.1 Input Information

Input Information	Source	
	Process	Activity
Contract Deliverable List	External	
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Configuration Identification	Software Configuration Management	Develop Configuration Identification (7.2.4)

7.2.3.2 Description

This Activity shall plan and document specific software configuration management organizations and responsibilities, procedures, tools, techniques, and methodologies in an SCMPI. The SCMPI shall also describe how and when such procedures are to be performed.

Overall software configuration management objectives are derived using internal guidelines as well as contractual requirements from the Software Project Management Planned Information (SPMPI).

Further information related to this Activity may be found in [B3] and [B13].

Prior to distribution of the Software Configuration 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)

7.2.3.3 Output Information

Output Information	Destination	
	Process	Activity
Software Configuration Management Planned Information	Project Monitoring and Control	Retain Records (3.2.6)
	Software Configuration Management	Develop Configuration Identification (7.2.4)
		Perform Configuration Control (7.2.5)
		Perform Status Accounting (7.2.6)

7.2.4 Develop Configuration Identification

7.2.4.1 Input Information

Input Information	Source	
	Process	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)

7.2.4.2 Description

This Activity shall define a Configuration Identification that includes project baseline definition, titling, labeling, and numbering to reflect the structure of the product for tracking. The SCMPI identifies those configuration items to be addressed by the Configuration Identification. The identification shall support the software throughout the SLC, and shall be documented in the SCMPI. The Configuration Identification shall also define the documentation required to record the functional and physical characteristics of each Configuration Item.

A series of baselines shall be established as the product moves from initial idea to the maintenance phase, as required by the SPMPI.

Further information related to this Activity may be found in [B3] and [B12].

7.2.4.3 Output Information

Output Information	Destination	
	Process	Activity
Configuration Identification	Software Configuration Management	Plan Configuration Management (7.2.3)

7.2.5 Perform Configuration Control

7.2.5.1 Input Information

Input Information	Source	
	Process	Activity
Items to Be Controlled	Creating Process	
Software Configuration Management Planned Information	Software Configuration Management	Plan Configuration Management (7.2.3)

7.2.5.2 Description

This Activity controls the configuration of products. Changes to controlled products shall be tracked to assure that the configuration of the product is known at all times. Each baseline shall be established and all subsequent changes tracked relative to it. All items specified in the SCMPI are subject to this change management discipline. The history of changes to each configuration item shall be maintained throughout the SLC for status accounting.

Changes to Controlled Items shall be allowed only with approval of the responsible authority. This may result in establishment of a formal software configuration control board. Controlled Items shall be maintained in a software library.

Further information related to this Activity may be found in [B3] and [B13].

7.2.5.3 Output Information

Output Information	Destination	
	Process	Activity
Change Status	Software Configuration Management	Perform Status Accounting (7.2.6)
Controlled Item	Creating Process	
	Project Monitoring and Control	Implement Problem Reporting Method (3.2.7)

7.2.6 Perform Status Accounting

7.2.6.1 Input Information

Input Information	Source	
	Process	Activity
Software Configuration Management Planned Information	Software Configuration Management	Plan Configuration Management (7.2.3)
Change Status	Software Configuration Management	Perform Configuration Control (7.2.5)

7.2.6.2 Description

This Activity shall include the receipt of Change Status from the Perform Configuration Control Activity and the preparation of Status Reported Information that reflects the status and history of controlled items. Status Reported Information may include such data as number of changes to date for the project, number of releases, and the latest version and revision identifiers.

Further information related to this Activity may be found in [B3] and [B13].

Prior to the distribution of the Status Reported Information, the following Processes shall be invoked:

a) Verification and Validation (7.1.4)
b) Documentation Development (7.3.4)

7.2.6.3 Output Information

Output Information	Destination	
	Process	Activity
Status Reported Information	Project Monitoring and Control	Manage the Project (3.2.5)
	External	

7.3 Documentation Development Process

7.3.1 Overview

The Documentation Development Process for software development and usage is the set of Activities that plan, design, implement, edit, produce, distribute, and maintain those documents needed by developers and users. The purpose of the Documentation Development Process is to provide timely software documentation to those who need it, based on Input Information from the invoking Processes.

This Process covers both product- and procedure-oriented documentation for internal and external users. Examples of internal users include those who plan, design, implement, or test software. External users may include those who install, operate, apply, or maintain the software.

The Documentation Development Process occurs over various phases of the software life cycle (SLC) depending on the individual document and the timing of its development. Typically there will be multiple documents, each at different stages of development.

The Documentation Development Process has Activities that must be performed concurrently with the software development or usage. Since the software is seldom stable during development or testing, this requires effective communication and timely response between the software personnel and documentation personnel.

7.3.2 Activities list

a) Plan Documentation
b) Implement Documentation
c) Produce and Distribute Documentation

7.3.3 Plan Documentation

7.3.3.1 Input Information

Input Information	Source	
	Process	Activity
Contractual Requirements	External	
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)

7.3.3.2 Description

In this Activity, information such as the Software Project Management Planned Information (SPMPI) product descriptions, schedules, and resource constraints shall be assimilated to create a consistent and disciplined approach to achieving the required documentation. The approach shall identify required documents, document production schedules and delivery schedules, and documentation standards. Responsible organizations, information sources, and intended audiences shall be defined for each document. The approach shall be documented in the Documentation Planning Information. The Documentation Planning Information shall include resource allocations for this Activity.

Additional guidance for the development of documentation can be found in [B4], [B5], [B11], [B16], and [B20].

Prior to distribution of the Documentation Planning Information, the following Processes shall be invoked:

a) Verification and Validation (7.1.4)
b) Documentation Development (7.3.4)

The Software Configuration Management Process (7.2.5) should also be invoked.

7.3.3.3 Output Information

Output Information	Destination	
	Process	Activity
Documentation Planned Information	Project Monitoring and Control	Retain Records (3.2.6)
	Implementation	Create Operating Documentation (5.3.6)
	Documentation Development	All Activities (7.3)

7.3.4 Implement Documentation

7.3.4.1 Input Information

Input Information	Source	
	Process	Activity
Input Information for Document	Creating Process	
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Documentation Planned Information	Documentation Development	Plan Documentation (7.3.3)

7.3.4.2 Description

This Activity includes the design, preparation, and maintenance of documentation. Those documents identified in the Documentation Planned Information shall be formulated in terms of audience, approach, content, structure, and graphics. Arrangements may be made with word or text processing and graphics facilities for their support of implementation.

Input Information shall be used to produce the document, including related graphics. This involves extensive use of information sources, close communication with the responsible subject matter experts, and utilization of word or text processing and graphics tools.

Following a documentation review, any changes shall be incorporated to produce a technically correct document. Format, style, and production rules shall be applied to produce a final document.

Prior to distribution of the Document, the following Processes should be invoked:

a) Verification and Validation (7.1.4)
b) Software Configuration Management (7.2.5)

7.3.4.3 Output Information

Output Information	Destination	
	Process	Activity
Document	Documentation Development	Produce and Distribute Documentation (7.3.5)

7.3.5 Produce and distribute documentation

7.3.5.1 Input Information

Input Information	Source	
	Process	Activity
Documentation Planned Information	Documentation Development	Plan Documentation (7.3.3)
Document	Documentation Development	Implement Documentation (7.3.4)

7.3.5.2 Description

This Activity shall provide the intended audience with the needed information collected in the document, as specified in the Documentation Planned Information. Document production and distribution may involve electronic file management, paper document reproduction and distribution, or other media handling techniques.

7.3.5.3 Output Information

Output Information	Destination	
	Process	Activity
Published Document	Project Monitoring and Control	Retain Records (3.2.6)
	Creating Process	
	External	

7.4 Training Process

7.4.1 Overview

The development of quality software products is largely dependent upon knowledgeable and skilled people. These include the developer's technical staff and management. Customer personnel may also have to be qualified to install, operate, and maintain the software. Training is therefore essential for developers, technical support staff, and customers. It is essential that Training Planned Information be completed early in the software life cycle, prior to the time when personnel would be expected to apply required expertise to the project. Plans for customer training should be prepared and reviewed with the customer.

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.

7.4.2 Activities list

a) Plan the Training Program
b) Develop Training Materials
c) Validate the Training Program
d) Implement the Training Program

7.4.3 Plan the Training Program

7.4.3.1 Input Information

Input Information	Source	
	Process	Activity
Applicable Information	External	
Skills Inventory	External	
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Software Requirements	Requirements	Prioritize and Integrate Software Requirements (5.1.5)
Training Feedback	Training	Implement the Training Program (7.4.6)
		Validate the Training Program (7.4.5)

7.4.3.2 Description

This Activity shall identify the needs for different types of training and the categories of people requiring training for each need. Customer and project information shall be reviewed along with existing personnel inventories. This information is used to produce Training Planned Information. Implementation schedules shall also be generated and resources allocated to the training program. Implementation schedules, resource allocations, and training needs shall be specified in the Training Planned Information.

Prior to distribution of the Training 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)

7.4.3.3 Output Information

Output Information	Destination	
	Process	Activity
Training Planned Information	Software Quality Management	Identify Quality Improvement Needs (3.3.6)
	Training	All Activities (7.4)

7.4.4 Develop Training Materials

7.4.4.1 Input Information

Input Information	Source	
	Process	Activity
Applicable Information	External	
	Creating Process	
Software Project Management Planned Information	Project Initiation	Plan Project Management (3.1.6)
Software Design Description	Design	Perform Detailed Design (5.2.7)
Training Planned Information	Training	Plan Training Program (7.4.3)

7.4.4.2 Description

This Activity shall consist of identification and review of all available materials that appear pertinent to the training objectives. Included in the Develop Training Materials Activity shall be the development of the substance of the training, training manual, and materials to be used in presenting the training, such as outlines, text, exercises, case studies, visuals, and models.

Prior to distribution of the Training Manual and Training Materials, the following Processes shall be invoked:

a) Verification and Validation (7.1.4)
b) Documentation Development (7.3.4)

The Software Configuration Management Process (7.2.5) should also be invoked.

7.4.4.3 Output Information

Output Information	Destination	
	Process	Activity
Training Manual	Training	Validate Training Program (7.4.5)
Training Materials	Training	Validate Training Program (7.4.5)

7.4.5 Validate the Training Program

7.4.5.1 Input Information

Input Information	Source	
	Process	Activity
Training Planned Information	Training	Plan Training Program (7.4.3)
Training Manual	Training	Develop Training Materials (7.4.4)
Training Materials	Training	Develop Training Materials (7.4.4)

7.4.5.2 Description

This Activity shall consist of presenting the training to a class of evaluators using the preliminary training manual and materials. The evaluators shall assess the training presentation and materials in detail. The purpose is to evaluate the effectiveness of the delivery and the validity of the material presented. Lessons learned in the test of the training program shall be incorporated into the material prior to a general offering. All training manuals and materials shall be evaluated and, if necessary, updated at this time.

Prior to distribution of Updated Training Manuals and Materials, the following Processes shall be invoked:

a) Verification and Validation (7.1.4)
b) Documentation Development (7.3.4)

The Software Configuration Management Process (7.2.5) should also be invoked.

7.4.5.3 Output Information

Output Information	Destination	
	Process	Activity
Updated Training Manual	Training	Implement Training Program (7.4.6)
Updated Training Materials	Training	Implement Training Program (7.4.6)
Training Feedback	Training	Plan Training Program (7.4.3)

7.4.6 Implement the Training Program

7.4.6.1 Input Information

Input Information	Source	
	Process	Activity
Staff Participants	External	
Students	External	
Training Planned Information	Training	Plan Training Program (7.4.3)
Updated Training Manual	Training	Validate Training Program (7.4.5)
Updated Training Materials	Training	Validate Training Program (7.4.5)

7.4.6.2 Description

This Activity shall ensure the provision of all necessary materials, arrange the locations and facilities for training, assign instructors and, if necessary, train them. Included in this Activity shall be the enrolling of students and monitoring of the course effectiveness.

Lessons learned and information needed for updating the materials for the next training cycle shall be fed back into the beginning of the Training Process.

7.4.6.3 Output Information

Output Information	Destination	
	Process	Activity
Trained Personnel	Creating Process	
Training Feedback	Training	Plan Training Program (7.4.3)
Updated Skills Inventory	External	

8. Bibliography

The IEEE standards listed below and other subsequent standards should be consulted when using this document. However, compliance with this standard neither requires nor implies compliance with the listed standards. Table 2 provides a cross reference of specific Activities to other IEEE standards.

[B1] IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology (ANSI).[4]

[B2] IEEE Std 730-1989, IEEE Standard for Software Quality Assurance Plans (ANSI).

[B3] IEEE Std 828-1990, IEEE Standard for Software Configuration Management Plans (ANSI).

[B4] IEEE Std 829-1983 (Reaff 1991), IEEE Standard for Software Test Documentation (ANSI).

[B5] IEEE Std 830-1993, IEEE Recommended Practice for Software Requirements Specifications (ANSI).

[B6] IEEE Std 982.1-1988, IEEE Standard Dictionary of Measures to Produce Reliable Software (ANSI).

[B7] IEEE Std 982.2-1988, IEEE Guide for the Use of IEEE Standard Dictionary of Measures to Produce Reliable Software (ANSI).

[B8] IEEE Std 1002-1987 (Reaff 1992), IEEE Standard Taxonomy for Software Engineering Standards (ANSI).

[B9] IEEE Std 1008-1987 (Reaff 1993), IEEE Standard for Software Unit Testing (ANSI).

[B10] IEEE Std 1012-1986 (Reaff 1992), IEEE Standard for Software Verification and Validation Plans (ANSI).

[B11] IEEE Std 1016-1987 (Reaff 1993), IEEE Recommended Practice for Software Design Descriptions (ANSI).

[B12] IEEE Std 1028-1988 (Reaff 1993), IEEE Standard for Software Reviews and Audits (ANSI).

[B13] IEEE Std 1042-1987 (Reaff 1993), IEEE Guide to Software Configuration Management (ANSI).

[B14] IEEE Std 1044-1993, IEEE Standard Classification for Software Anomalies.

[B15] IEEE Std 1045-1992, IEEE Standard for Software Productivity Metrics (ANSI).

[B16] IEEE Std 1058.1-1987 (Reaff 1993), IEEE Standard for Software Project Management Plans (ANSI).

[B17] IEEE Std 1059-1993, IEEE Guide for Software Verification and Validation Plans(ANSI).

[B18] IEEE Std 1061-1992, IEEE Standard for a Software Quality Metrics Methodology.

[B19] IEEE Std 1062-1993, IEEE Recommended Practice for Software Acquisition (ANSI).

[B20] IEEE Std 1063-1987 (Reaff 1993), IEEE Standard for Software User Documentation (ANSI).

[B21] ANSI Technical Report, American National Dictionary for Information Processing, X3/TR-1-77, September, 1977.

[4]IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

Table 2—Cross reference of IEEE standards and authorized standards projects

	Applicability of Standards							
Activities (Paragraph Numbers) / Referenced IEEE Standards and Authorized Standards Projects	1.3.1	3.1.5	3.1.6	3.2.7	3.3.3	3.3.4	5.1.3 5.1.5	5.2.7
[B1] 610.12-1990	X							
[B2] 730-1989					X			
[B3] 828-1990								
[B4] 829-1983								
[B5] 830-1993	X						X	
[B6] 982.1-1988						X		
[B7] 982.2-1988						X		
[B8] 1002-1987		X						
[B9] 1008-1987								
[B10] 1012-1986								
[B11] 1016-1987								X
[B12] 1028-1988								
[B13] 1042-1987								
[B14] 1044-1993				X				
[B15] 1045-1992						X		
[B16] 1058.1-1987	X		X					
[B17] 1059-1993								
[B18] 1061-1992						X		
[B19] 1062-1993								
[B20] 1063-1987								

Table 2—Cross reference of IEEE standards and authorized standards projects *(continued)*

Applicability of Standards									
Activities (Paragraph Numbers) / Referenced IEEE Standards	5.3.3 5.3.5	5.3.6	7.1.3	7.1.4	7.1.5	7.1.6	7.1.7 7.1.8	7.2.3 7.2.4 7.2.5 7.2.6	7.3.3
[B1] 610.12-1990									
[B2] 730-1989			X	X					
[B3] 828-1990			X	X				X	
[B4] 829-1983			X			X	X		
[B5] 830-1993									
[B6] 982.1-1988			X		X				
[B7] 982.2-1988			X		X				
[B8] 1002-1987									
[B9] 1008-1987	X					X	X		
[B10] 1012-1986			X	X		X	X		
[B11] 1016-1987									
[B12] 1028-1988			X	X					
[B13] 1042-1987			X	X				X	
[B14] 1044-1993			X		X				
[B15] 1045-1992			X		X				
[B16] 1058.1-1987									
[B17] 1059-1993						X			
[B18] 1061-1992									
[B19] 1062-1993									
[B20] 1063-1987		X							X

Annex A

(informative)

Mapping Software Life Cycle Processes to various examples of Software Life cycles

This annex demonstrates the mappings of the Activities in this standard to four different software life cycles (SLCs). This annex is not intended to be comprehensive. Many other SLCs are possible, for example, small development, quick reaction, and the spiral model. The SLCs presented here are examples only, and the user of this document is not required to select any of these SLCs.

Table A.1 demonstrates a mapping of Activities to an eight-phase SLC.

Table A.2 demonstrates a mapping of Activities to a five-phase SLC.

Table A.3 demonstrates a mapping of Activities to an SLC that uses prototyping to establish requirements and design.

Table A.4 demonstrates a mapping of Activities to an SLC that includes a theoretical, highly automated software development mode.

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

Table A.1—Software Life Cycle example based on eight phases

Concept Exploration (CE)
Design (DE)
Test (TE)
Operation and Maintenance (OM)
Requirements (RQ)
Implementation (IM)
Installation and Checkout (IN)
Retirement (RT)

Activities	CE	RQ	DE	IM	TE	IN	OM	RT
SOFTWARE LIFE CYCLE PROCESS								
Identify Candidate SLC Models	X							
Select Project Model	X							
PROJECT MANAGEMENT PROCESSES								
Project Initiation Process								
Map Activities to SLC Model	X							
Allocate Project Resources	X	X	X	X	X	X	X	X
Establish Project Environment	X	X	X					
Plan Project Management	X	X						
Project Monitoring and Control Process								
Analyze Risks	X	X	X	X	X	X		
Perform Contingency Planning		X	X	X	X	X		
Manage the Project	X	X	X	X	X	X	X	X
Retain Records	X	X	X	X	X	X	X	X
Implement Problem Reporting System		X	X	X	X	X	X	X
Software Quality Management Process								
Plan Software Quality Management	X	X	X					
Define Metrics		X	X					
Manage Software Quality	X	X	X	X	X	X	X	X
Identify Quality Improvement Needs	X	X	X	X	X	X	X	X
PRE-DEVELOPMENT PROCESSES								
Concept Exploration Process								
Identify Ideas or Needs	X							
Formulate Potential Approaches	X	X						
Conduct Feasibility Studies	X	X	X	X	X	X	X	X
Plan System Transition (if applicable)	X	X						X
Refine and Finalize the Idea or Need		X						
System Allocation Process								
Analyze Functions		X	X					
Develop System Architecture		X	X					
Decompose System Requirements		X						

Table A.1—Software Life Cycle example based on eight phases *(continued)*

Activities	CE	RQ	DE	IM	TE	IN	OM	RT
DEVELOPMENT PROCESSES								
Requirements Process								
Define and Develop Software Requirements		X	X					
Define Interface Requirements		X	X					
Prioritize and Integrate Software Requirements		X	X					
Design Process								
Perform Architectural Design			X					
Design Data Base (if applicable)			X					
Design Interfaces			X					
Select or Develop Algorithms		X	X					
Perform Detailed Design			X					
Implementation Process								
Create Test Data			X	X				
Create Source			X	X				
Generate Object Code			X	X				
Create Operating Documentation			X	X				
Plan Integration			X					
Perform Integration				X	X			
POST-DEVELOPMENT PROCESSES								
Installation Process								
Plan Installation					X			
Distribute Software						X		
Install Software						X		
Accept Software in Operational Environment						X		
Operation and Support Process								
Operate the System							X	
Provide Tech. Asst. & Consult.							X	
Maintain Support Request Log							X	
Maintenance Process								
Reapply Software Life Cycle							X	
Retirement Process								
Notify User							X	X
Conduct Parallel Operations (if applicable)								X
Retire System								X

Table A.1—Software Life Cycle example based on eight phases *(continued)*

Activities	CE	RQ	DE	IM	TE	IN	OM	RT
INTEGRAL PROCESSES								
Verification and Validation Process								
Plan Verification and Validation	X	X	X					
Execute V&V Tasks	X	X	X	X	X	X	X	X
Collect and Analyze Metric Data		X	X	X	X	X	X	X
Plan Testing		X	X	X				
Develop Test Requirements			X	X				
Execute the Tests				X	X	X		
Software Configuration Management Process								
Plan Configuration Management		X	X					
Perform Configuration Identification		X	X	X	X			
Perform Configuration Control			X	X	X	X	X	X
Perform Status Accounting			X	X	X	X	X	X
Documentation Development Process								
Plan Documentation		X	X					
Implement Documentation		X	X					
Produce and Distribute Documentation				X	X			
Training Process								
Plan the Training Program		X	X					
Develop Training Materials			X	X	X			
Validate the Training Program					X	X		
Implement the Training Program						X		

Table A.2—Software Life Cycle example based on five phases

Project Initiation (PI)
Concept Development (CD)
Definition and Design (DD)
System Development (SD)
Installation and Operation (IO)

Activities	PI	CD	DD	SD	IO
SOFTWARE LIFE CYCLE PROCESS					
Identify Candidate SLC Models	X				
Select Project Model	X				
PROJECT MANAGEMENT PROCESSES					
Project Initiation Process					
Map Activities to SLC Model	X				
Allocate Project Resources	X	X	X	X	X
Establish Project Environment	X	X	X		
Plan Project Management	X	X			
Project Monitoring and Control Process					
Analyze Risks	X	X	X	X	X
Perform Contingency Planning		X	X	X	X
Manage the Project	X	X	X	X	X
Retain Records	X	X	X	X	X
Implement Problem Reporting System		X	X	X	X
Software Quality Management Process					
Plan Software Quality Management	X	X	X		
Define Metrics		X	X		
Manage Software Quality	X	X	X	X	X
Identify Quality Improvement Needs	X	X	X	X	X
PRE-DEVELOPMENT PROCESSES					
Concept Exploration Process					
Identify Ideas or Needs	X				
Formulate Potential Approaches	X	X			
Conduct Feasibility Studies	X	X			
Plan System Transition (if applicable)	X	X			
Refine and Finalize the Idea or Need		X			
System Allocation Process					
Analyze Functions		X	X		
Develop System Architecture		X	X		
Decompose System Requirements		X			

Table A.2—Software Life Cycle example based on five phases *(continued)*

Activities	PI	CD	DD	SD	IO
DEVELOPMENT PROCESSES					
Requirements Process					
Define and Develop Software Requirements		X	X		
Define Interface Requirements		X	X		
Prioritize and Integrate Software Requirements		X	X		
Design Process					
Perform Architectural Design			X		
Design Data Base (if applicable)			X		
Design Interfaces			X		
Select or Develop Algorithms		X	X		
Perform Detailed Design			X		
Implementation Process					
Create Test Data			X	X	
Create Source			X	X	
Generate Object Code			X	X	
Create Operating Documentation			X	X	
Plan Integration			X		
Perform Integration				X	
POST-DEVELOPMENT PROCESSES					
Installation Process					
Plan Installation				X	
Distribute Software					X
Install Software					X
Accept Software in Operational Environment					X
Operation and Support Process					
Operate the System					X
Provide Tech. Asst. & Consult.					X
Maintain Support Request Log					X
Maintenance Process					
Reapply Software Life Cycle					X
Retirement Process					
Notify User					X
Conduct Parallel Operations (if applicable)					X
Retire System					X

Table A.2—Software Life Cycle example based on five phases *(continued)*

Activities	PI	CD	DD	SD	IO
INTEGRAL PROCESSES					
Verification and Validation Process					
Plan Verification and Validation	X	X	X		
Execute V&V Tasks		X	X	X	X
Collect and Analyze Metric Data		X	X	X	X
Plan Testing			X	X	
Develop Test Requirements			X	X	
Execute the Tests			X	X	X
Software Configuration Management Process					
Plan Configuration Management		X	X		
Perform Configuration Identification		X	X	X	
Perform Configuration Control			X	X	X
Perform Status Accounting			X	X	X
Documentation Development Process					
Plan Documentation		X	X		
Implement Documentation			X	X	
Produce and Distribute Documentation				X	X
Training Process					
Plan the Training Program		X	X		
Develop Training Materials			X	X	
Validate the Training Program				X	X
Implement the Training Program					X

Table A.3—Software Life Cycle example based on prototyping

Concept Exploration (CE)
Implementation (IM)
Installation and Checkout (IN)
Retirement (RT)
Prototyping (PT)
Test (TE)
Operation and Maintenance (OM)

Activities	CE	PT	IM	TE	IN	OM	RT
SOFTWARE LIFE CYCLE PROCESS							
Identify Candidate SLC Models	X						
Select Project Model	X						
PROJECT MANAGEMENT PROCESSES							
Project Initiation Process							
Map Activities to SLC Model	X						
Allocate Project Resources	X	X	X	X	X	X	X
Establish Project Environment	X	X					
Plan Project Management	X	X					
Project Monitoring and Control Process							
Analyze Risks	X	X	X	X	X		
Perform Contingency Planning		X	X	X	X		
Manage the Project	X	X	X	X	X	X	X
Retain Records	X	X	X	X	X	X	X
Implement Problem Reporting System		X	X	X	X	X	X
Software Quality Management Process							
Plan Software Quality Management		X					
Define Metrics		X					
Manage Software Quality	X	X	X	X	X	X	X
Identify Quality Improvement Needs	X	X	X	X	X	X	X
PRE-DEVELOPMENT PROCESSES							
Concept Exploration Process							
Identify Ideas or Needs	X						
Formulate Potential Approaches	X	X					
Conduct Feasibility Studies	X	X					
Plan System Transition (if applicable)	X	X					X
Refine and Finalize the Idea or Need		X					
System Allocation Process							
Analyze Functions		X					
Develop System Architecture		X					
Decompose System Requirements		X					

Table A.3—Software Life Cycle example based on prototyping *(continued)*

Activities	CE	PT	IM	TE	IN	OM	RT
DEVELOPMENT PROCESSES							
Requirements Process							
Define and Develop Software Requirements		X					
Define Interface Requirements		X					
Prioritize and Integrate Software Requirements.		X					
Design Process							
Perform Architectural Design		X					
Design Data Base (if applicable)		X					
Design Interfaces		X					
Select or Develop Algorithms		X					
Perform Detailed Design		X					
Implementation Process							
Create Test Data		X	X				
Create Source		X	X				
Generate Object Code		X	X	X			
Create Operating Documentation		X	X				
Plan Integration		X					
Perform Integration			X	X			
POST-DEVELOPMENT PROCESSES							
Installation Process							
Plan Installation				X			
Distribute Software					X		
Install Software					X		
Check out Software in Operational Envnmt.					X		
Operation and Support Process							
Operate the System						X	
Provide Tech. Asst. & Consult.						X	
Maintain Support Request Log						X	
Maintenance Process							
Reapply Software Life Cycle						X	

Table A.3—Software Life Cycle example based on prototyping *(continued)*

Activities	CE	PT	IM	TE	IN	OM	RT
Retirement Process							
Notify User						X	X
Conduct Parallel Operations (if applicable)							X
Retire System							X
INTEGRAL PROCESSES							
Verification and Validation Process							
Plan Verification and Validation		X					
Execute V&V Tasks		X	X	X	X	X	X
Collect and Analyze Metric Data		X	X	X	X	X	X
Plan Testing		X	X				
Develop Test Specifications		X	X				
Execute the Tests			X	X	X		
Software Configuration Management Process							
Plan Configuration Management		X					
Perform Configuration Identification		X	X	X			
Perform Configuration Control		X	X	X	X	X	X
Perform Status Accounting		X	X	X	X	X	X
Documentation Development Process							
Plan Documentation		X					
Implement Documentation		X	X				
Produce and Distribute Documentation				X	X		
Training Process							
Plan the Training Program		X					
Develop Training Materials		X	X	X			
Validate the Training Program				X	X		
Implement the Training Program					X		

Table A.4—Software Life Cycle example based on an Operational Specification[5]

System Requirements (SR) Operational Specification (OS)
Transformed Specification (TS) Delivered System (DS)

Activities	SR	OS	TS	DS
SOFTWARE LIFE CYCLE PROCESS				
Identify Candidate SLC Models	X			
Select Project Model	X			
PROJECT MANAGEMENT PROCESSES				
Project Initiation Process				
Map Activities to SLC Model	X			
Allocate Project Resources	X	X	X	X
Establish Project Environment	X	X		
Plan Project Management	X			
Project Monitoring and Control Process				
Analyze Risks	X	X	X	X
Perform Contingency Planning	X	X	X	X
Manage the Project	X	X	X	X
Retain Records	X	X	X	X
Implement Problem Reporting System		X	X	X
Software Quality Management Process				
Plan Software Quality Management	X	X		
Define Metrics	X	X		
Manage Software Quality	X	X	X	X
Identify Quality Improvement Needs	X	X	X	X
PRE-DEVELOPMENT PROCESSES				
Concept Exploration Process				
Identify Ideas or Needs	X			
Formulate Potential Approaches	X			
Conduct Feasibility Studies	X	X		
Plan System Transition (if applicable)	X			X
Refine and Finalize the Idea or Need	X			
System Allocation Process				
Analyze Functions	X	X		
Develop System Architecture	X	X		
Decompose System Requirements	X			

[5]As defined in the IEEE Tutorial, *New Paradigms for software development,* by William W. Agresti.

Table A.4—Software Life Cycle example based on an Operational Specification *(continued)*

Activities	SR	OS	TS	DS
DEVELOPMENT PROCESSES				
Requirements Process				
Define and Develop Software Requirements	X	X		
Define Interface Requirements	X	X		
Prioritize and Integrate Software Requirements	X	X		
Design Process				
Perform Architectural Design		X		
Design Data Base (if applicable)		X		
Design Interfaces		X		
Select or Develop Algorithms	X	X		
Perform Detailed Design		X		
Implementation Process				
Create Test Data		X	X	
Create Source		X	X	
Generate Object Code		X	X	
Create Operating Documentation		X	X	
Plan Integration		X		
Perform Integration			X	
POST-DEVELOPMENT PROCESSES				
Installation Process				
Plan Installation			X	
Distribute Software				X
Install Software				X
Check out Software in Operational Environment				X
Operation and Support Process				
Operate the System				X
Provide Tech. Asst. & Consult.				X
Maintain Support Request Log				X
Maintenance Process				
Reapply Software Life Cycle				X

Table A.4—Software Life Cycle example based on an Operational Specification *(continued)*

Activities	SR	OS	TS	DS
INTEGRAL PROCESSES				
Retirement Process				
Notify User				X
Conduct Parallel Operations (if applicable)				X
Retire System				X
Verification and Validation Process				
Plan Verification and Validation	X	X		
Execute V&V Tasks	X	X	X	X
Collect and Analyze Metric Data	X	X	X	X
Plan Testing		X	X	
Develop Test Specifications		X	X	
Execute the Tests			X	X
Software Configuration Management Process				
Plan Configuration Management	X	X		
Perform Configuration Identification	X	X	X	
Perform Configuration Control		X	X	X
Perform Status Accounting		X	X	X
Documentation Development Process				
Plan Documentation	X	X		
Implement Documentation		X	X	
Produce and Distribute Documentation			X	X
Training Process				
Plan the Training Program	X	X		
Develop Training Materials		X	X	
Validate the Training Program			X	X
Implement the Training Program				X

Annex B

(informative)

Software Project Management Tailoring Template

The Software Project Management Tailoring Template is designed to assist project managers in identifying project-critical deliverables and assuring their completion as needed.

This template may be used to assist in the project-specific mapping of information into the required project documentation.

Table B.1—Software Project Management Tailoring Template

SOFTWARE PROJECT MANAGEMENT TAILORING TEMPLATE			
Process or Activity Name	Clause	Output Information	Mapped Deliverables or Activities
PROCESS GROUP **Process** Activity		Required Output Information	
SOFTWARE LIFE CYCLE PROCESS	2		
Identify Candidate SLC Models	2.3	Candidate SLC Model(s)	
Select Project Model	2.4	Selected SLC Model	
PROJECT MANAGEMENT PROCESSES	3		
Project Initiation	3.1		
Map Activities to SLC Model	3.1.3	Software Life Cycle	
		List of Activities Not Used	
Allocate Project Information	3.1.4	Resource Allocations	
Establish Project Environment	3.1.5	Project Environment	
Plan Project Management	3.1.6	Problem Reporting & Resolution Planned Info.	
		Retirement Planned Info.	
		Software Project Management Planned Info.	
		Support Planned Info.	
Project Monitoring and Control	3.2		
Analyze Risks	3.2.3	Analysis of Risks	
Perform Contingency Planning	3.2.4	Contingency Planned Info.	
Manage the Project	3.2.5	Project Management Reported Info.	
		Anomalies	
Retain Records	3.2.6	Historical Project Records	
Implement Problem Reporting Method	3.2.7	Resolved Problem Reported Info.	
		Report Log	
		Enhancement Problem Reported Info.	
		Corrections Problem Reported Info.	

Table B.1—Software Project Management Tailoring Template *(continued)*

SOFTWARE PROJECT MANAGEMENT TAILORING TEMPLATE			
Process or Activity Name	Clause	Output Information	Mapped Deliverables or Activities
Software Quality Management	3.3		
Plan Software Quality Management	3.3.3	Software Quality Management Planned Info.	
Define Metrics	3.3.4	Defined Metrics	
		Collection and Analysis Methods	
Manage Software Quality	3.3.5	Project Quality Assessments	
Identify Quality Improvement Needs	3.3.6	Quality Improvement Recommendations	
PRE-DEVELOPMENT PROCESSES	**4**		
Concept Exploration	**4.1**		
Identify Ideas or Needs	4.1.3	Prelim. Statement of Need	
Formulate Potential Approaches	4.1.4	Constraints and Benefits	
		Potential Approaches	
Conduct Feasibility Studies	4.1.5	Recommendations	
Plan System Transition (if applicable)	4.1.6	Transition Impact Statement	
		Transition Planned Info.	
Refine and Finalize the Idea or Need	4.1.7	Statement of Need	
System Allocation	**4.2**		
Analyze Functions	4.2.3	Funct'l Description of System	
Develop System Architecture	4.2.4	System Architecture	
Decompose System Requirements	4.2.5	Funct'l Hardware Rqmts.	
		Funct'l Software Rqmts.	
		System Interface Requirements (if applicable)	
DEVELOPMENT PROCESSES	**5**		
Requirements	**5.1**		
Define and Develop Software Rqmts.	5.1.3	Prelim. Software Rqmts.	
		Installation Rqmts.	
Define Interface Requirements	5.1.4	Software Interface Rqmts.	
Prioritize and Integrate Software Rqmts.	5.1.5	Software Rqmts.	

Table B.1—Software Project Management Tailoring Template *(continued)*

SOFTWARE PROJECT MANAGEMENT TAILORING TEMPLATE			
Process or Activity Name	Clause	Output Information	Mapped Deliverables or Activities
Design	**5.2**		
Perform Architectural Design	5.2.3	Software Architectural Design Description	
Design Data Base (if applicable)	5.2.4	Data Base Description	
Design Interfaces	5.2.5	Interface Description	
Select or Develop Algorithms	5.2.6	Algorithm Descriptions	
Perform Detailed Design	5.2.7	Software Design Description	
Implementation	**5.3**		
Create Test Data	5.3.3	Stubs and Drivers (if applicable)	
		Test Data	
Create Source	5.3.4	Data Base (if applicable)	
		Source Code	
Generate Object Code	5.3.5	Corrected Data Base (if applicable)	
		Corrected Source Code	
		Object Code	
Create Operating Documentation	5.3.6	Operating Documentation	
Plan Integration	5.3.7	Integration Planned Info.	
Perform Integration	5.3.8	Integrated Software	
POST-DEVELOPMENT PROCESSES	**6**		
Installation	**6.1**		
Plan Installation	6.1.3	Software Installation Planned Info.	
Distribute Software	6.1.4	Packaged Operating Docs.	
		Packaged Software	
		Packaged Installation Planned Info.	
Install Software	6.1.5	Installation Reported Info.	
		Installed Software	
Accept Software in Operational Envrmt.	6.1.6	Installed Software System	
		Customer Acceptance	

Table B.1—Software Project Management Tailoring Template *(continued)*

SOFTWARE PROJECT MANAGEMENT TAILORING TEMPLATE			
Process or Activity Name	Clause	Output Information	Mapped Deliverables or Activities
Operation and Support	**6.2**		
Operate the System	6.2.3	System Anomalies	
		Operation Logs	
		Feedback Data	
Provide Tech. Asst. & Consult.	6.2.4	Support Response	
Maintain Support Request Log	6.2.5	Anomalies	
		Support Request Log	
Maintenance	**6.3**		
Reapply Software Life Cycle	6.3.3	Maintenance Recommendations	
Retirement	**6.4**		
Notify User	6.4.3	Official Notification	
Conduct Parallel Operations	6.4.4	Parallel Operations Log	
Retire System	6.4.5	Post-Operation Review Reported Information	
		Archive Reported Info.	
INTEGRAL PROCESSES	**7**		
Verification and Validation	**7.1**		
Plan V&V	7.1.3	Software V&V Planned Info.	
Execute V&V Tasks	7.1.4	Evaluation Reported Info.	
		Anomalies	
Collect and Analyze Metric Data	7.1.5	Analysis Reported Info.	
Plan Testing	7.1.6	Test Planned Info.	
Develop Test Requirements	7.1.7	Test Requirements	
Execute the Tests	7.1.8	Test Summary Reported Info.	
		Tested Software	
		Anomalies	

Table B.1—Software Project Management Tailoring Template *(continued)*

SOFTWARE PROJECT MANAGEMENT TAILORING TEMPLATE			
Process or Activity Name	Clause	Output Information	Mapped Deliverables or Activities
Software Configuration Management	**7.2**		
Plan Configuration Management	7.2.3	Software Config. Management Planned Info.	
Develop Configuration Identification	7.2.4	Config. Identification	
Perform Configuration Control	7.2.5	Change Status	
		Controlled Item	
Perform Status Accounting	7.2.6	Status Reported Info.	
Documentation Development	**7.3**		
Plan Documentation	7.3.3	Documentation Planned Info.	
Implement Documentation	7.3.4	Document	
Produce and Distribute Documentation	7.3.5	Published Document	
Training	**7.4**		
Plan the Training Program	7.4.3	Training Planned Info.	
Develop Training Materials	7.4.4	Training Manual	
		Training Materials	
Validate the Training Program	7.4.5	Updated Training Manual	
		Updated Training Materials	
		Training Feedback	
Implement the Training Program	7.4.6	Trained Personnel	
		Training Feedback	
		Updated Skills Inventory	

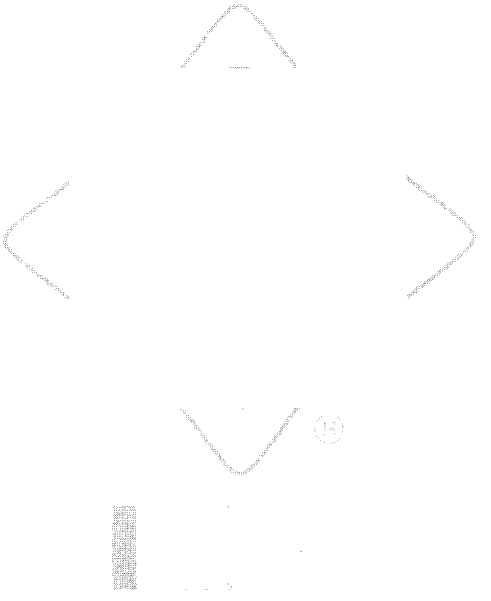
Annex C

(informative)

Process interrelationships

Figure C.1 in this annex shows the interrelationships between the Processes. The boxes in figure C.1 are Processes; the directed lines show the flow of input and output information between Processes. The Management Processes are grouped in the center of the figure, with the Development-Oriented Processes arranged around them.

This figure does not attempt to show Process invocations. All the directed lines to and from the Invoked Processes represent explicit information passed between those Processes.



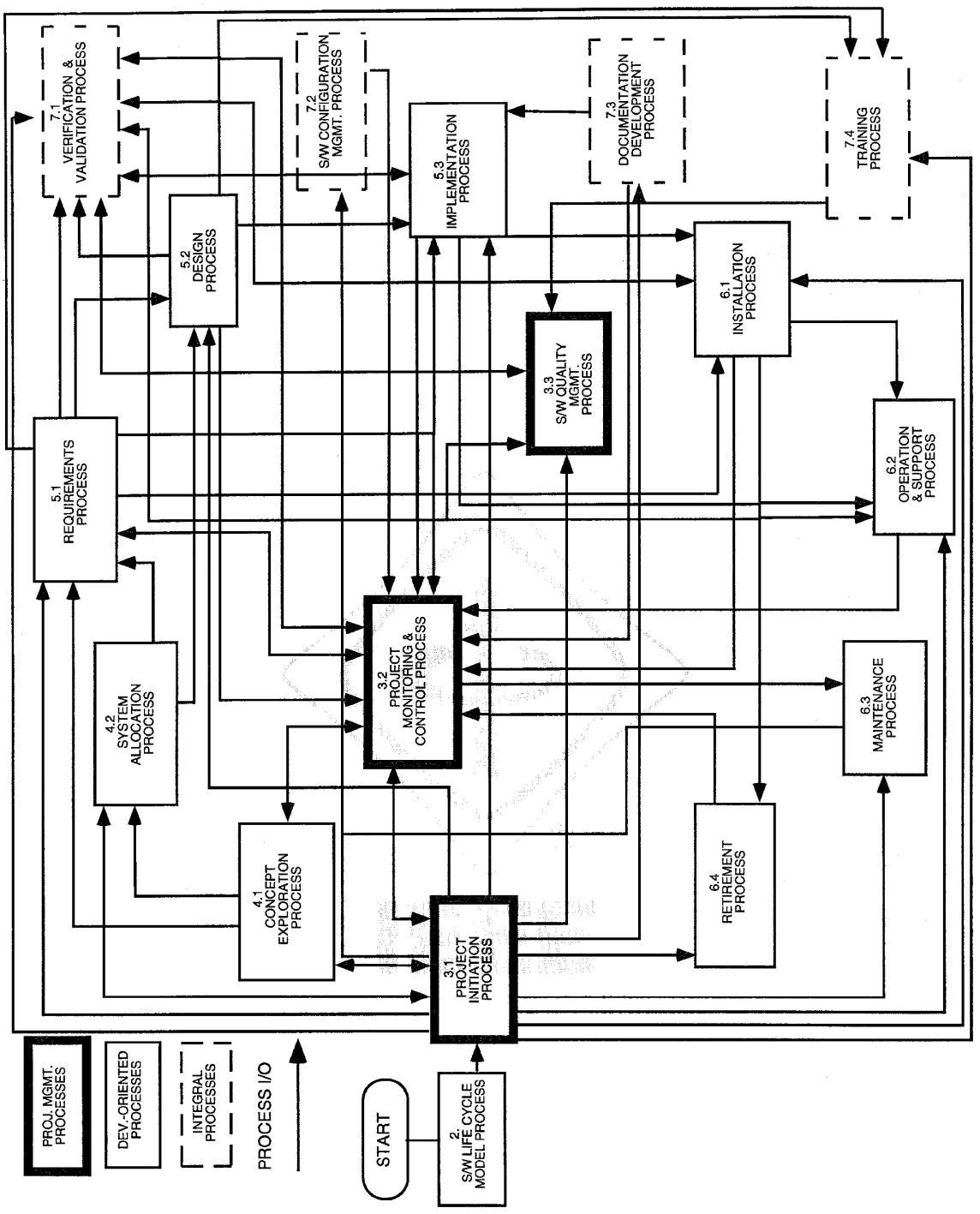


Figure C.1—Process interrelationships