IEEE/EIA 12207.0-1996

(A Joint Standard Developed by IEEE and EIA)

IEEE/EIA Standard

Industry Implementation of International Standard ISO/IEC 12207 : 1995

(ISO/IEC 12207) Standard for Information Technology—

Software life cycle processes

March 1998

Networking the World™

IEEE/EIA 12207.0-1996

(A Joint Standard Developed by IEEE and EIA)

IEEE/EIA Standard

Industry Implementation of International Standard ISO/IEC 12207: 1995

(ISO/IEC 12207) Standard for Information Technology—

Software life cycle processes

March 1998

Abstract: ISO/IEC 12207 provides a common framework for developing and managing software. IEEE/EIA 12207.0 consists of the clarifications, additions, and changes accepted by the Institute of Electrical and Electronics Engineers (IEEE) and the Electronic Industries Alliance (EIA) as formulated by a joint project of the two organizations. IEEE/EIA 12207.0 contains concepts and guidelines to foster better understanding and application of the standard. Thus this standard provides industry a basis for software practices that would be usable for both national and international business.
Keywords: acquisition process, audit, configuration management, development process, maintenance process, operation process, quality assurance, supply process, tailoring process, validation, verification

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

Print: ISBN 1-55937-977-4, SH94581
PDF: ISBN 0-7381-0428-0, SS94581

March 1998

NOTICE

IEEE and EIA Standards and Publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for his particular need. Use of such Standards and Publications is wholly voluntary. Existence of such standards and Publications shall not in any respect preclude any member or nonmember of IEEE and EIA from manufacturing or selling products not conforming to such Standards and Publications, nor shall the existence of such Standards and Publications preclude their voluntary use by those other than IEEE and EIA members, whether the standard is to be used either domestically or internationally.

Standards and publications are approved by IEEE and EIA in accordance with the American National Standards Institute (ANSI) patent policy. By such action, IEEE and EIA do not assume any liability to any patent owner, nor do they assume any obligation whatever to parties adopting the Standard or Publication.

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. IEEE and EIA shall not be responsible for identifying all patents for which a license may be required by an IEEE and EIA standard or for conducting inquiries into the legal validity or scope of those patents that are brought to their attention.

Contents

Foreword to IEEE/EIA 12207.0-1996 v

ISO/IEC 12207 : 1995

Foreword ix

Introduction x

1. Scope 1

2. Normative references 2

3. Definitions 3

4. Application of this International Standard 6

5. Primary life cycle processes 9

   5.1 Acquisition process 10
   5.2 Supply process 13
   5.3 Development process 16
   5.4 Operation process 23
   5.5 Maintenance process 24

6. Supporting processes 27

   6.1 Documentation process 28
   6.2 Configuration management process 29
   6.3 Quality assurance process 31
   6.4 Verification process 33
   6.5 Validation process 36
   6.6 Joint review process 38
   6.7 Audit process 40
   6.8 Problem resolution process 41

7. Organizational life cycle processes 42

   7.1 Management process 43
   7.2 Infrastructure process 45
   7.3 Improvement process 46
   7.4 Training process 47

Annexes

A—Tailoring process 48

B—Guidance on tailoring 49

C—Guidance on processes and organizations 53

D—Bibliography 57

IEEE/EIA 12207.0-1996

E—Basic concepts of ISO/IEC 12207 58

F—Compliance 64

G—Life cycle processes objectives 66

H—Life cycle data objectives 72

I—Relationships 74

J—Errata 75

Foreword to IEEE/EIA 12207.0-1996

1. The International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) published ISO/IEC 12207, *Information technology—Software life cycle processes*, in August, 1995. IEEE/EIA 12207.0 consists of the clarifications, additions, and changes accepted by the Institute of Electrical and Electronics Engineers (IEEE) and the Electronic Industries Alliance (EIA) as formulated by a joint project of the two organizations. IEEE/EIA 12207.0 contains concepts and guidelines to foster better understanding and application of the standard. Thus this standard provides industry a basis for software practices that would be usable for both national and international business. Readers are advised to refer to the information on Basic concepts (annex E), Compliance (annex F), Life cycle processes objectives (annex G), Life cycle data objectives (annex H), Relationships (annex I), and Errata (annex J) prior to implementing this standard. Annexes E through J have been added to IEEE/EIA 12207.0-1996 to enhance the readerís understanding of this standard.

2. IEEE/EIA 12207.0 may be used to

   a. Acquire, supply, develop, operate, and maintain software.

   b. Support the above functions in the form of quality assurance, configuration management, joint reviews, audits, verification, validation, problem resolution, and documentation.

   c. Manage and improve the organization's processes and personnel.

   d. Establish software management and engineering environments based upon the life cycle processes as adapted and tailored to serve business needs.

   e. Foster improved understanding between customers and vendors and among the parties involved in the life cycle of a software product.

   f. Facilitate world trade in software.

3. IEEE/EIA 12207 is packaged in three parts. The three parts are, briefly, as follows:

   IEEE/EIA 12207.0, *Standard for Information Technology—Software life cycle processes*: Contains ISO/IEC 12207 in its original form and six additional annexes (E through J): Basic concepts; Compliance; Life cycle process objectives; Life cycle data objectives; Relationships; and Errata. A unique IEEE/EIA foreword is included.

   IEEE/EIA P12207.1[1], Guide for ISO/IEC 12207, *Standard for Information Technology—Software life cycle processes—Life cycle data*: Provides additional guidance on recording life cycle data.

   IEEE/EIA P12207.2, Guide for ISO/IEC 12207, *Standard for Information Technology—Software life cycle processes—Implementation considerations*: Provides additions, alternatives, and clarifications to the ISO/IEC 12207's life cycle processes as derived from U.S. practices.

[1] IEEE/EIA P12207.1 and P12207.2 are draft standards.

Participants in IEEE/EIA 12207.0

The unique front matter and annexes to IEEE/EIA 12207.0 were developed by the Joint Industrial Standard Working Group (JISWG) of IEEE and EIA, including the following persons:

Raghu Singh, *IEEE Co-chair* **Perry DeWeese**, *EIA Co-chair*
Dorothy Kuckuck, *Project Editor*

Dennis Ahern
Chuck Baker
Barbara Bankeroff
Johnny Barrett
Ron Berlack
Ed Beverly
Barry Boehm
John Bolland
John Bowers
Max Brown
Don Calvert
Stuart Campbell
Quyen Cao
Bruce Capehart
Dennis Carter
Myra M. Chern
John P. Chihorek
Jack Cooper
Raymond Coyle
Paul Croll
Dario de Angelis
Chris Denham
Perry DeWeese
Robert Didrikson
Jim Dobbins
Merlin Dorfman
Cheryl Dorsey
Bernadette Downward
Peter Eirich
Bob Elston
Richard Evans
Dennis Faulkner
Marv Gechman
William Gess, Jr.
Marilyn Ginsberg-Finner
Lewis Gray
John Halase
John Hamlin
Rick Hefner
Robert Hegland
James H. Heil
Mark Henley
John Hoover
Helmut Hummel
Allan Jaworski
John Kerr
Lynne Ketchie
Larry Klos
Robert Knickerbocker
Steve Kopp
Richard Kreke
Dorothy Kuckuck
Jerome Lake
Doug Lange
Amy Laughlin
Milton Lavin
Jonathan Liles
Joan Lovelace
Marv Lubofsky
David Maibor
Ed Martin
Zyggy Martynowkcz
Richard McClellen
Judy McCloskey
Sandy McGill
Fred Mintz
Aretha Moore
James Moore
Jackie Morman
Gary Motchan
Bart Nigro
George Nowinski
Al Olsen
Myrna L. Olson
Sherry Paquin
Alex Polack
Ken Ptack
Ralph Randall
Paul Reindollar
Walter Richter
Bill Schunk
Adrienne Scott
Keith Shewbridge
Carl A. Singer
Terry Snyder
Reed Sorensen
Don Sova
John Stolzenthaler
Norma A. Stopyra
Richard F. Storch
Duane Stratton
Robert Tausworthe
Booker T. Thomas
Frederick Tilsworth
Leonard L. Tripp
Ann Turner
Howard Verne
Ronald L. Wade
David Waxman
Charles Wilson
Grady Wright

The following persons were on the IEEE balloting committee:

Syed Ali
Mikhail Auguston
Robert E. Barry
Leo Beltracchi
Mordechai Ben-Menachem
H. Ronald Berlack
Christos Bezirtzoglou
William J. Boll
Audrey C. Brewer
Alan L. Bridges
Kathleen L. Briggs
M. Scott Buck
David W. Burnett
Edward R. Byrne
Stuart Ross Campbell
Leslie Chambers
Keith Chan
Betty P. Chao
S. V. Chiyyarath
Antonio M. Cicu
Theo Clarke
Francois Coallier
Rosemary Coleman
Virgil Lee Cooper
Geoff Cozens
Gregory T. Daich
Hillary Davidson
Bostjan K. Derganc
Sanjay Dewal
Harpal Dhama
James Do
Audrey Dorofee
Carl Einar Dragstedt
Leo Egan
Richard L. Evans
William Eventoff
Jonathan H. Fairclough
John W. Fendrich
Jay Forster
Kirby Fortenberry
Eva Freund
Adel N. Ghannam
Hiranmay Ghosh
Marilyn Ginsberg-Finner
M. Joel Gittleman
John Garth Glynn
Julio Gonzalez Sanz
Donald Gotterbarn
Lewis Gray
Eric Grosse
Lawrence M. Gunther
Jon Hagar
John Harauz
Rob Harker
Robert T. Harley
Herbert Hecht
William Hefley
Manfred Hein
Mark Heinrich
Mark Henley
Umesh P. Hiriyannaiah
John W. Horch
Jerry Huller
Peter L. Hung
Fabrizio Imelio
George Jackelen
Frank V. Jorgensen
Vladan V. Jovanovic
William S. Junk
George X. Kambic
Diana Kang
Chris F. Kemerer
Judy Kerner
Robert J. Kierzyk
Dwayne L. Knirk
Thomas M. Kurihara
John B. Lane
J. Dennis Lawrence
Randal Leavitt
Michael Lines
James J. Longbucco
Dieter Look
John Lord
M. Lubofsky
Austin J. Maher
David Maibor
Harold Mains
Philip P. Mak
Gianluca Marcellino
John R. Matras
Tomoo Matsubara
Mike McAndrew
Patrick McCray
Russell McDowell
Sue McGrath
Jerome W. Mersky
Fatma Mili
Alan Miller
Lisa Ming
Millard Allen Mobley
Charles S. Mooney
James W. Moore
Pavol Navrat
Dennis E. Nickle
Myrna L. Olson
Mike Ottewill
Gerald L. Ourada
Lalit Mohan Patnaik
Mark Paulk
John G. Phippen
Alex Polack
Peter T. Poon
Lawrence S. Przybylski
Larry K. Reed
Ann Reedy
Annette D. Reilly
Dennis Rilling
Patricia Rodriguez
Patti Rusher
Andrew P. Sage
Helmut Sandmayr
Hans Schaefer
Norman Schneidewind
David J. Schultz
Robert W. Shillato
Katsutoshi Shintani
Carl A. Singer
Raghu P. Singh
James M. Sivak
Nancy M. Smith
Alfred R. Sorkowitz
Donald W. Sova
Luca Spotorno
Julia Stesney
Norma Stopyra
Fred J. Strauss
Christine Brown Strysik
Michael Surratt
Toru Takeshita
Douglas H. Thiele
Booker Thomas
Patricia Trellue
Leonard L. Tripp
T. H. Tse
Richard D. Tucker
Margaret C. Updike
Theodore J. Urbanowicz
Glenn D. Venables
David W. Vickers
Udo Voges
Thomas E. Vollman
Ronald L. Wade
Dolores Wallace
John W. Walz
Charles J. Wertz
Camille S. White-Partain
Scott A. Whitmire
Paul A. T. Wolfgang
Paul R. Work
Forrest D. Wright
Weider D. Yu
Janusz Zalewski
Zhi Ying Zhou
Geraldine Zimmerman
Peter F. Zoll

When the IEEE Standards Board approved IEEE/EIA 12207.0 on 10 December 1996, it had the following membership:

Donald C. Loughry, *Chair* **Richard J. Holleman**, *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
Ben C. Johnson
E. G. "Al" Kiener
Joseph L. Koepfinger*
Stephen R. Lambert
Lawrence V. McCall
L. Bruce McClung
Marco W. Migliaro
Mary Lou Padgett
John W. Pope
Jose R. Ramos
Arthur K. Reilly
Ronald H. Reimer
Gary S. Robinson
Ingo Rüsch
John S. Ryan
Chee Kiow Tan
Leonard L. Tripp
Howard L. Wolfman

*Member Emeritus

Also included are the following nonvoting IEEE Standards Board liaisons:

Satish K. Aggarwal
Alan H. Cookson
Chester C. Taylor

Kristin M. Dittmann
IEEE Standards Project Editor

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 12207 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software engineering*.

Annex A forms an integral part of the International Standard. Annexes B and C are for information only.

Introduction

Software is an integral part of information technology and conventional systems, such as transportation, military, medical care, and finance. There is a proliferation of standards, procedures, methods, tools, and environments for developing and managing software. This proliferation has created difficulties in software management and engineering, especially in integrating products and services. The software discipline needs to migrate from this proliferation to a common framework that can be used by software practitioners to "speak the same language" to create and manage software. This International Standard provides such a common framework.

The framework covers the life cycle of software from conceptualization of ideas through retirement and consists of processes for acquiring and supplying software products and services. In addition, the framework provides for controlling and improving these processes.

The processes in this International Standard form a comprehensive set. An organization, depending on its purpose, can select an appropriate subset to fulfill that purpose. The International Standard is, therefore, designed to be tailored for an individual organization, project, or application. It is also designed to be used when software is a stand-alone entity, or an embedded or integral part of the total system.

Information technology — Software life cycle processes

1 Scope

1.1 Purpose

This International Standard establishes a common framework for software life cycle processes, with well-defined terminology, that can be referenced by the software industry. It contains processes, activities, and tasks that are to be applied during the acquisition of a system that contains software, a stand-alone software product, and software service and during the supply, development, operation, and maintenance of software products. Software includes the software portion of firmware.

This International Standard also provides a process that can be employed for defining, controlling, and improving software life cycle processes.

1.2 Field of application

This International Standard applies to the acquisition of systems and software products and services, to the supply, development, operation, and maintenance of software products, and to the software portion of firmware, whether performed internally or externally to an organization. Those aspects of system definition needed to provide the context for software products and services are included.

NOTE—The processes used during the software life cycle need to be compatible with the processes used during the system life cycle.

This International Standard is intended for use in a two-party situation and may be equally applied where the two parties are from the same organization. The situation may range from an informal agreement up to a legally binding contract. This International Standard may be used by a single party as self-imposed tasks.

This International Standard is not intended for off-the-shelf software products unless incorporated into a deliverable product.

This International Standard is written for acquirers of systems and software products and services and for suppliers, developers, operators, maintainers, managers, quality assurance managers, and users of software products.

1.3 Tailoring of the International Standard

This International Standard contains a set of processes, activities, and tasks designed to be tailored in respect of software projects. The tailoring process is deletion of non-applicable processes, activities, and tasks.

NOTE—Addition of unique or special processes, activities, and tasks may be provided in the contract.

1.4 Compliance

Compliance with this International Standard is defined as the performance of all the processes, activities, and tasks selected from this International Standard in the Tailoring Process (annex A) for the software project. The performance of a process or an activity is complete when all its required tasks are performed in accordance with the pre-established criteria and the requirements specified in the contract as applicable.

Any organization (for example, national, industrial association, company) imposing this International Standard, as a condition of trade, is responsible for specifying and making public the minimum set of required processes, activities, and tasks, which constitute suppliers' compliance with this International Standard.

1.5 Limitations

This International Standard describes the architecture of the software life cycle processes but does not specify the details of how to implement or perform the activities and tasks included in the processes.

This International Standard is not intended to prescribe the name, format, or explicit content of the documentation to be produced. This International Standard may require development of documents of similar class or type; various plans are an example. This International Standard, however, does not imply that such documents be developed or packaged separately or combined in some fashion. These decisions are left to the user of this International Standard.

This International Standard does not prescribe a specific life cycle model or software development method. The parties of this International Standard are responsible for selecting a life cycle model for the software project and mapping the processes, activities, and tasks in this International Standard onto that model. The parties are also responsible for selecting and applying the software development methods and for performing the activities and tasks suitable for the software project.

This International Standard is not intended to be in conflict with any organization's policies, standards or procedures that are already in place. However, any conflict needs to be resolved and any overriding conditions and situations need to be cited in writing as exceptions to the application of this International Standard.

Throughout this International Standard, "shall" is used to express a provision that is binding between two or more parties, "will" to express a declaration of purpose or intent by one party, "should" to express a recommendation among other possibilities, and "may" to indicate a course of action permissible within the limits of this International Standard.

In this International Standard, there are a number of lists for tasks; none of these is presumed to be exhaustive—they are intended as examples.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/AFNOR: 1989, *Dictionary of computer science*.

ISO 2382-1: 1993, *Information technology—Vocabulary—Part 1: Fundamental terms*.

ISO/IEC 2382-20: 1990, *Information technology—Vocabulary; Part 20: System development*.

ISO 8402: 1994, *Quality management and quality assurance—Vocabulary*.

ISO 9001: 1994, *Quality systems—Models for quality assurance in design, development, production, installation and servicing*.

ISO/IEC 9126: 1991, *Information technology—Software product evaluation—Quality characteristics and guidelines for their use*.

3 Definitions

For the purposes of this International Standard, the definitions given in ISO 8402, ISO/IEC 2382-1 and ISO/IEC 2382-20 apply, together with the following definitions.

NOTE—A product may be interpreted as a part of a system as applicable.

3.1 Acquirer: An organization that acquires or procures a system, software product or software service from a supplier.

NOTE—The acquirer could be one of the following: buyer, customer, owner, user, purchaser.

3.2 Acquisition: The process of obtaining a system, software product or software service.

3.3 Agreement: The definition of terms and conditions under which a working relationship will be conducted.

3.4 Audit: Conducted by an authorized person for the purpose of providing an independent assessment of software products and processes in order to assess compliance with requirements.

3.5 Baseline: A formally approved version of a configuration item, regardless of media, formally designated and fixed at a specific time during the configuration item's life cycle.

3.6 Configuration item: An entity within a configuration that satisfies an end use function and that can be uniquely identified at a given reference point.

3.7 Contract: A binding agreement between two parties, especially enforceable by law, or a similar internal agreement wholly within an organization, for the supply of software service or for the supply, development, production, operation, or maintenance of a software product.

3.8 Developer: An organization that performs development activities (including requirements analysis, design, testing through acceptance) during the software life cycle process.

3.9 Evaluation: A systematic determination of the extent to which an entity meets its specified criteria.

3.10 Firmware: The combination of a hardware device and computer instructions or computer data that reside as read-only software on the hardware device. The software cannot be readily modified under program control.

3.11 Life cycle model: A framework containing the processes, activities, and tasks involved in the development, operation, and maintenance of a software product, spanning the life of the system from the definition of its requirements to the termination of its use.

3.12 Maintainer: An organization that performs maintenance activities.

3.13 Monitoring: An examination of the status of the activities of a supplier and of their results by the acquirer or a third party.

3.14 Non-deliverable item: Hardware or software product that is not required to be delivered under the contract but may be employed in the development of a software product.

3.15 Off-the-shelf product: Product that is already developed and available, usable either "as is" or with modification.

3.16 Operator: An organization that operates the system.

3.17 Process: A set of interrelated activities, which transform inputs into outputs.

NOTE—The term "activities" covers use of resources.

3.18 Qualification: The process of demonstrating whether an entity is capable of fulfilling specified requirements. [See ISO 8402: 1994, 2.13.]

3.19 Qualification requirement: A set of criteria or conditions that have to be met in order to qualify a software product as complying with its specifications and being ready for use in its target environment.

3.20 Qualification testing: Testing, conducted by the developer and witnessed by the acquirer (as appropriate), to demonstrate that the software product meets its specifications and is ready for use in its target environment.

3.21 Quality assurance: All the planned and systematic activities implemented within the quality system, and demonstrated as needed, to provide adequate confidence that an entity will fulfill requirements for quality.

NOTES

1 There are both internal and external purposes for quality assurance:

a) Internal quality assurance: within an organization, quality assurance provides confidence to management;

b) External quality assurance: in contractual situations, quality assurance provides confidence to the customer or others.

2 Some quality control and quality assurance actions are interrelated.

3 Unless requirements for quality fully reflect the needs of the user, quality assurance may not provide adequate confidence.

[ISO 8402: 1994, 3.5]

3.22 Release: A particular version of a configuration item that is made available for a specific purpose (for example, test release).

3.23 Request for proposal [tender]: A document used by the acquirer as the means to announce its intention to potential bidders to acquire a specified system, software product or software service.

3.24 Retirement: Withdrawal of active support by the operation and maintenance organization, partial or total replacement by a new system, or installation of an upgraded system.

3.25 Security: The protection of information and data so that unauthorized persons or systems cannot read or modify them and authorized persons or systems are not denied access to them.

3.26 Software product: The set of computer programs, procedures, and possibly associated documentation and data.

3.27 Software service: Performance of activities, work, or duties connected with a software product, such as its development, maintenance, and operation.

3.28 Software unit: A separately compilable piece of code.

3.29 Statement of work: A document used by the acquirer as the means to describe and specify the tasks to be performed under the contract.

3.30 Supplier: An organization that enters into a contract with the acquirer for the supply of a system, software product or software service under the terms of the contract.

NOTES

1 The term "supplier" is synonymous with contractor, producer, seller, or vendor.

2 The acquirer may designate a part of its organization as supplier.

3.31 System: An integrated composite that consists of one or more of the processes, hardware, software, facilities and people, that provides a capability to satisfy a stated need or objective.

3.32 Test coverage: The extent to which the test cases test the requirements for the system or software product.

3.33 Testability: The extent to which an objective and feasible test can be designed to determine whether a requirement is met.

3.34 User: An individual or organization that uses the operational system to perform a specific function.

NOTE—The user may perform other roles such as acquirer, developer, or maintainer.

3.35 Validation: Confirmation by examination and provision of objective evidence that the particular requirements for a specific intended use are fulfilled.

NOTES

1 In design and development, validation concerns the process of examining a product to determine conformity with user needs.

2 Validation is normally performed on the final product under defined operating conditions. It may be necessary in earlier stages.

3 "Validated" is used to designate the corresponding status.

4 Multiple validations may be carried out if there are different intended uses.

[ISO 8402: 1994, 2.18]

3.36 Verification: Confirmation by examination and provision of objective evidence that specified requirements have been fulfilled.

NOTES

1 In design and development, verification concerns the process of examining the result of a given activity to determine conformity with the stated requirement for that activity.

2 "Verified" is used to designate the corresponding status.

[ISO 8402: 1994, 2.17]

3.37 Version: An identified instance of an item.

NOTE—Modification to a version of a software product, resulting in a new version, requires configuration management action.

4 Application of this International Standard

This clause presents the software life cycle processes that can be employed to acquire, supply, develop, operate, and maintain software products. The objective is to provide a road map for the users of this International Standard so that they can orient themselves in it and apply it judiciously.

4.1 Organization of this International Standard

4.1.1 Life cycle processes

This International Standard groups the activities that may be performed during the life cycle of software into five primary processes, eight supporting processes, and four organizational processes. Each life cycle process is divided into a set of activities; each activity is further divided into a set of tasks. Subclause numbering a.b denotes a process, a.b.c an activity, and a.b.c.d a task. These life cycle processes are introduced below and depicted in figure 1.

4.1.1.1 Primary processes

The primary processes (clause 5) consist of five processes that serve primary parties during the life cycle of software. A primary party is one that initiates or performs the development, operation, or maintenance of software products. These primary parties are the acquirer, the supplier, the developer, the operator, and the maintainer of software products. The primary processes are:

1) *Acquisition process* (subclause 5.1). Defines the activities of the acquirer, the organization that acquires a system, software product or software service.

2) *Supply process* (subclause 5.2). Defines the activities of the supplier, the organization that provides the system, software product or software service to the acquirer.

3) *Development process* (subclause 5.3). Defines the activities of the developer, the organization that defines and develops the software product.

4) *Operation process* (subclause 5.4). Defines the activities of the operator, the organization that provides the service of operating a computer system in its live environment for its users.

5) *Maintenance process* (subclause 5.5). Defines the activities of the maintainer, the organization that provides the service of maintaining the software product; that is, managing modifications to the software product to keep it current and in operational fitness. This process includes the migration and retirement of the software product.

4.1.1.2 Supporting life cycle processes

The supporting life cycle processes (clause 6) consist of eight processes. A supporting process supports another process as an integral part with a distinct purpose and contributes to the success and quality of the software project. A supporting process is employed and executed, as needed, by another process. The supporting processes are:

1) *Documentation process* (subclause 6.1). Defines the activities for recording the information produced by a life cycle process.

2) *Configuration management process* (subclause 6.2). Defines the configuration management activities.

3) *Quality assurance process* (subclause 6.3). Defines the activities for objectively assuring that the software products and processes are in conformance with their specified requirements and adhere to their established plans. Joint Reviews, Audits, Verification, and Validation may be used as techniques of Quality Assurance.

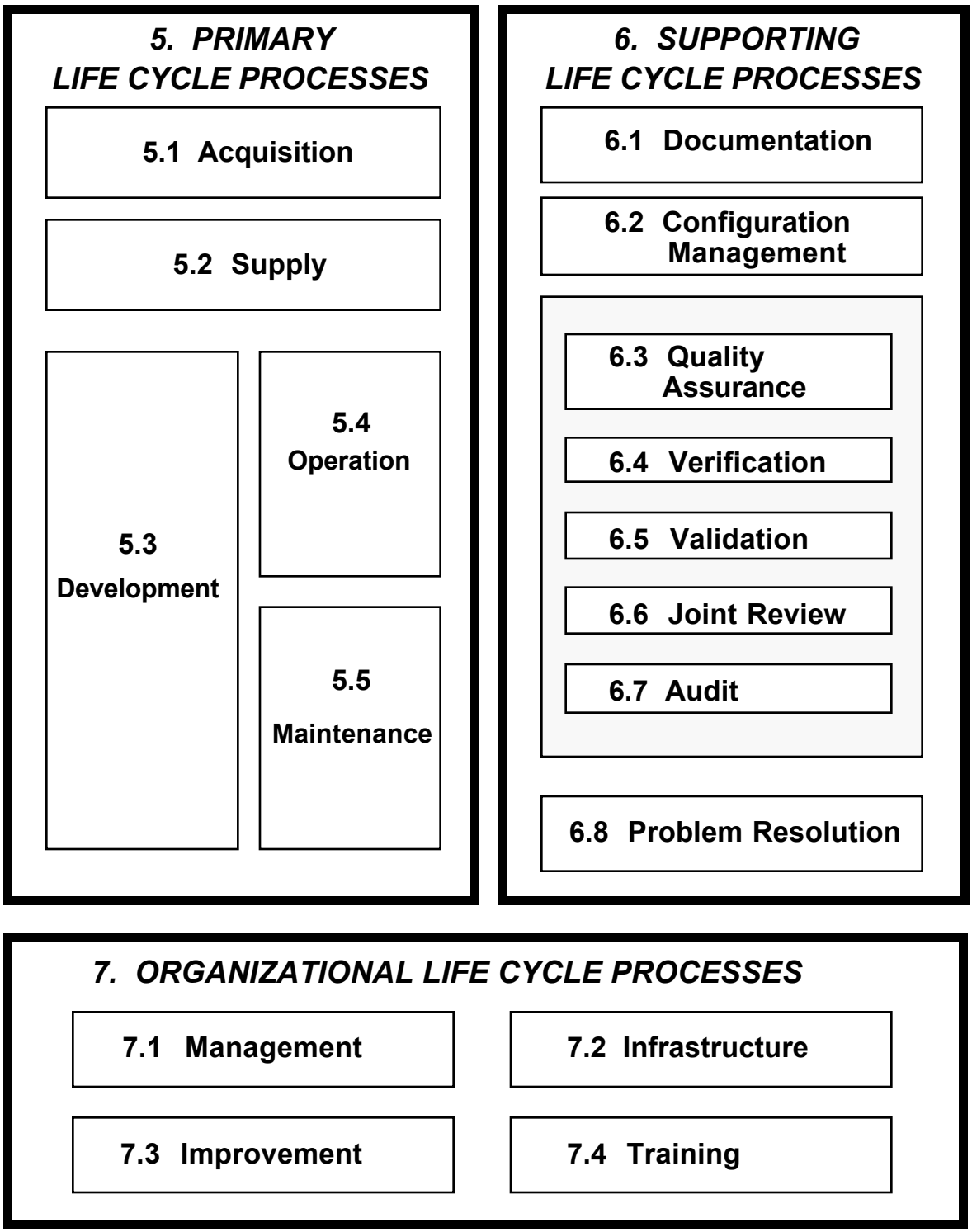


Figure 1. Structure of this International Standard

4) *Verification process* (subclause 6.4). Defines the activities (for the acquirer, the supplier, or an independent party) for verifying the software products and services in varying depth depending on the software project.

5) *Validation process* (subclause 6.5). Defines the activities (for the acquirer, the supplier, or an independent party) for validating the software products of the software project.

6) *Joint review process* (subclause 6.6). Defines the activities for evaluating the status and products of an activity. This process may be employed by any two parties, where one party (reviewing party) reviews another party (reviewed party) in a joint forum.

7) *Audit process* (subclause 6.7). Defines the activities for determining compliance with the requirements, plans, and contract. This process may be employed by any two parties, where one party (auditing party) audits the software products or activities of another party (audited party).

8) *Problem resolution process* (subclause 6.8). Defines a process for analyzing and removing the problems (including nonconformances), whatever their nature or source, that are discovered during the execution of development, operation, maintenance, or other processes.

4.1.1.3 Organizational life cycle processes

The organizational life cycle processes (clause 7) consist of four processes. They are employed by an organization to establish and implement an underlying structure made up of associated life cycle processes and personnel and continuously improve the structure and processes. They are typically employed outside the realm of specific projects and contracts; however, lessons from such projects and contracts contribute to the improvement of the organization. The organizational processes are:

1) *Management process* (subclause 7.1). Defines the basic activities of the management, including project management, related to the execution of a life cycle process.

2) *Infrastructure process* (subclause 7.2). Defines the basic activities for establishing the underlying structure of a life cycle process.

3) *Improvement process* (subclause 7.3). Defines the basic activities that an organization (that is, acquirer, supplier, developer, operator, maintainer, or the manager of another process) performs for establishing, measuring, controlling, and improving its life cycle process.

4) *Training process* (subclause 7.4). Defines the activities for providing adequately trained personnel.

4.1.2 Tailoring process. Annex A, which is normative, defines the basic activities needed to perform tailoring of this International Standard. Annex B contains a brief guidance on tailoring the requirements of this International Standard; it lists the key factors upon which tailoring decisions may be made.

4.1.3 Relationship between the processes and organizations

This International Standard contains various processes that are applied throughout the life cycle of software by various organizations depending on their needs and goals. For understandability, Annex C presents the relationships between the life cycle processes and related parties.

5 Primary life cycle processes

This clause defines the following primary life cycle processes:

1) Acquisition process;
2) Supply process;
3) Development process;
4) Operation process;
5) Maintenance process.

The activities and tasks in a primary process are the responsibility of the organization initiating and performing that process. This organization ensures that the process is in existence and functional.

5.1 Acquisition process

The Acquisition Process contains the activities and tasks of the acquirer. The process begins with the definition of the need to acquire a system, software product or software service. The process continues with the preparation and issue of a request for proposal, selection of a supplier, and management of the acquisition process through to the acceptance of the system, software product or software service.

The individual organization having the need may be called the owner. The owner may contract any or all of the acquisition activities to an agent who will in turn conduct these activities according to the Acquisition Process. The acquirer in this subclause may be the owner or the agent.

The acquirer manages the Acquisition Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4).

List of activities: This process consists of the following activities:

1) Initiation;
2) Request-for-Proposal [-tender] preparation;
3) Contract preparation and update;
4) Supplier monitoring;
5) Acceptance and completion.

5.1.1 Initiation. This activity consists of the following tasks:

5.1.1.1 The acquirer begins the acquisition process by describing a concept or a need to acquire, develop, or enhance a system, software product or software service.

5.1.1.2 The acquirer will define and analyze the system requirements. The system requirements should include business, organizational and user as well as safety, security, and other criticality requirements along with related design, testing, and compliance standards and procedures.

5.1.1.3 If the acquirer retains a supplier to perform system requirements analysis, the acquirer will approve the analyzed requirements.

5.1.1.4 The acquirer may perform the definition and analysis of software requirements by itself or may retain a supplier to perform this task.

5.1.1.5 The Development Process (5.3) should be used to perform the tasks in 5.1.1.2 and 5.1.1.4.

5.1.1.6 The acquirer will consider options for acquisition against analysis of appropriate criteria to include risk, cost and benefits for each option. Options include:

a) Purchase an off-the-shelf software product that satisfies the requirements.
b) Develop the software product or obtain the software service internally.
c) Develop the software product or obtain the software service through contract.
d) A combination of a, b, and c above.
e) Enhance an existing software product or service.

5.1.1.7 When an off-the-shelf software product is to be acquired, the acquirer will ensure the following conditions are satisfied:

a) The requirements for the software product are satisfied.
b) The documentation is available.
c) Proprietary, usage, ownership, warranty and licensing rights are satisfied.
d) Future support for the software product is planned.

5.1.1.8 The acquirer should prepare, document and execute an acquisition plan. The plan should contain the following:

a) Requirements for the system;
b) Planned employment of the system;
c) Type of contract to be employed;
d) Responsibilities of the organizations involved;
e) Support concept to be used;
f) Risks considered as well as methods to manage the risks.

5.1.1.9 The acquirer should define and document the acceptance strategy and conditions (criteria).

5.1.2 Request-for-proposal [-tender] preparation. This activity consists of the following tasks:

5.1.2.1 The acquirer should document the acquisition requirements (e.g., request for proposal), the content of which depends upon the acquisition option selected in 5.1.1.6. The acquisition documentation should include, as appropriate:

a) System requirements;
b) Scope statement;
c) Instructions for bidders;
d) List of software products;
e) Terms and conditions;
f) Control of subcontracts;
g) Technical constraints (e.g., target environment).

5.1.2.2 The acquirer should determine which processes, activities, and tasks of this International Standard are appropriate for the project and should tailor them accordingly. Especially, the acquirer should specify the applicable supporting processes (clause 6) and their performing organizations, including responsibilities (if other than supplier), so that the suppliers may, in their proposals, define the approach to each of the specified supporting processes. The acquirer will define the scope of those tasks that reference the contract.

5.1.2.3 The acquisition documentation will also define the contract milestones at which the supplier's progress will be reviewed and audited as part of monitoring the acquisition (see 6.6 and 6.7).

5.1.2.4 The acquisition requirements should be given to the organization selected for performing the acquisition activities.

5.1.3 Contract preparation and update. This activity consists of the following tasks:

5.1.3.1 The acquirer should establish a procedure for supplier selection including proposal evaluation criteria and requirements compliance weighting.

5.1.3.2 The acquirer should select a supplier based upon the evaluation of the suppliers' proposals, capabilities, and other factors that need to be considered.

5.1.3.3 The acquirer may involve other parties, including potential suppliers, before contract award, in tailoring this International Standard for the project. However, the acquirer will make the final decision on the tailoring. The acquirer will include or reference the tailored International Standard in the contract.

5.1.3.4 The acquirer will then prepare and negotiate a contract with the supplier, that addresses the acquisition requirements, including the cost and schedule, of the software product or service to be delivered. The contract will address proprietary, usage, ownership, warranty and licensing rights associated with the reusable off-the-shelf software products.

5.1.3.5 Once the contract is underway, the acquirer will control changes to the contract through negotiation with the supplier as part of a change control mechanism. Changes to the contract shall be investigated for impact on project plans, costs, benefits, quality, and schedule.

NOTE—The acquirer determines whether the term "contract" or "agreement" is to be used in the application of this International Standard.

5.1.4 Supplier monitoring. This activity consists of the following tasks:

5.1.4.1 The acquirer will monitor the supplier's activities in accordance with the Joint Review Process (6.6) and the Audit Process (6.7). The acquirer should supplement the monitoring with the Verification Process (6.4) and the Validation Process (6.5) as needed.

5.1.4.2 The acquirer will cooperate with the supplier to provide all necessary information in a timely manner and resolve all pending items.

5.1.5 Acceptance and completion. This activity consists of the following tasks:

5.1.5.1 The acquirer should prepare for acceptance based on the defined acceptance strategy and criteria. The preparation of test cases, test data, test procedures, and test environment should be included. The extent of supplier involvement should be defined.

5.1.5.2 The acquirer will conduct acceptance review and acceptance testing of the deliverable software product or service and will accept it from the supplier when all acceptance conditions are satisfied. The acceptance procedure should comply with the provisions of 5.1.1.9.

5.1.5.3 After acceptance, the acquirer should take the responsibility for the configuration management of the delivered software product (see 6.2).

NOTE—The acquirer may install the software product or perform the software service in accordance with instructions defined by the supplier.

5.2 Supply process

The Supply Process contains the activities and tasks of the supplier. The process may be initiated either by a decision to prepare a proposal to answer an acquirer's request for proposal or by signing and entering into a contract with the acquirer to provide the system, software product or software service. The process continues with the determination of procedures and resources needed to manage and assure the project, including development of project plans and execution of the plans through delivery of the system, software product or software service to the acquirer.

The supplier manages the Supply Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4).

List of activities: This process consists of the following activities:

1) Initiation;
2) Preparation of response;
3) Contract;
4) Planning;
5) Execution and control;
6) Review and evaluation;
7) Delivery and completion.

5.2.1 Initiation. This activity consists of the following tasks:

5.2.1.1 The supplier conducts a review of requirements in the request for proposal taking into account organizational policies and other regulations.

5.2.1.2 The supplier should make a decision to bid or accept the contract.

5.2.2 Preparation of response. This activity consists of the following task:

5.2.2.1 The supplier should define and prepare a proposal in response to the request for proposal, including its recommended tailoring of this International Standard.

5.2.3 Contract. This activity consists of the following tasks:

5.2.3.1 The supplier shall negotiate and enter into a contract with the acquirer organization to provide the software product or service.

5.2.3.2 The supplier may request modification to the contract as part of the change control mechanism.

5.2.4 Planning. This activity consists of the following tasks:

5.2.4.1 The supplier shall conduct a review of the acquisition requirements to define the framework for managing and assuring the project and for assuring the quality of the deliverable software product or service.

5.2.4.2 If not stipulated in the contract, the supplier shall define or select a software life cycle model appropriate to the scope, magnitude, and complexity of the project. The processes, activities, and tasks of this International Standard shall be selected and mapped onto the life cycle model.

5.2.4.3 The supplier shall establish requirements for the plans for managing and assuring the project and for assuring the quality of the deliverable software product or service. Requirements for the plans should include resource needs and acquirer involvement.

5.2.4.4 Once the planning requirements are established, the supplier shall consider the options for developing the software product or providing the software service, against an analysis of risks associated with each option. Options include:

a) Develop the software product or provide the software service using internal resources.
b) Develop the software product or provide the software service by subcontracting.

c) Obtain off-the-shelf software products from internal or external sources.
d) A combination of a, b, and c above.

5.2.4.5 The supplier shall develop and document project management plan(s) based upon the planning requirements and options selected in 5.2.4.4. Items to be considered in the plan include but are not limited to the following:

a) Project organizational structure and authority and responsibility of each organizational unit, including external organizations;

b) Engineering environment (for development, operation, or maintenance, as applicable), including test environment, library, equipment, facilities, standards, procedures, and tools;

c) Work breakdown structure of the life cycle processes and activities, including the software products, software services and non-deliverable items, to be performed together with budgets, staffing, physical resources, software size, and schedules associated with the tasks;

d) Management of the quality characteristics of the software products or services. Separate plans for quality may be developed.

e) Management of the safety, security, and other critical requirements of the software products or services. Separate plans for safety and security may be developed.

f) Subcontractor management, including subcontractor selection and involvement between the subcontractor and the acquirer, if any;

g) Quality assurance (see 6.3);

h) Verification (see 6.4) and validation (see 6.5); including the approach for interfacing with the verification and validation agent, if specified;

i) Acquirer involvement; that is, by such means as joint reviews (see 6.6), audits (see 6.7), informal meetings, reporting, modification and change; implementation, approval, acceptance, and access to facilities;

j) User involvement; by such means as requirements setting exercises, prototype demonstrations and evaluations;

k) Risk management; that is management of the areas of the project that involve potential technical, cost, and schedule risks;

l) Security policy; that is, the rules for need-to-know and access-to-information at each project organization level;

m) Approval required by such means as regulations, required certifications, proprietary, usage, ownership, warranty and licensing rights;

n) Means for scheduling, tracking, and reporting;

o) Training of personnel (see 7.4).

5.2.5 Execution and control. This activity consists of the following tasks:

5.2.5.1 The supplier shall implement and execute the project management plan(s) developed in 5.2.4.

5.2.5.2 The supplier shall:

a) Develop the software product in accordance with Development Process (5.3).
b) Operate the software product in accordance with Operation Process (5.4).
c) Maintain the software product in accordance with Maintenance Process (5.5).

5.2.5.3 The supplier shall monitor and control the progress and the quality of the software products or services of the project throughout the contracted life cycle. This shall be an ongoing, iterative task, which shall provide for:

a) Monitoring progress of technical performance, costs, and schedules and reporting of project status;

b) Problem identification, recording, analysis, and resolution.

5.2.5.4 The supplier shall manage and control the subcontractors in accordance with the Acquisition Process (5.1). The supplier shall pass down all contractual requirements necessary to ensure that the software product or service delivered to the acquirer is developed or performed in accordance with the prime-contract requirements.

5.2.5.5 The supplier shall interface with the independent verification, validation, or test agent as specified in the contract and project plans.

5.2.5.6 The supplier shall interface with other parties as specified in the contract and project plans.

5.2.6 Review and evaluation. This activity consists of the following tasks:

5.2.6.1 The supplier should coordinate contract review activities, interfaces, and communication with the acquirer's organization.

5.2.6.2 The supplier shall conduct or support the informal meetings, acceptance review, acceptance testing, joint reviews, and audits with the acquirer as specified in the contract and project plans. The joint reviews shall be conducted in accordance with 6.6, audits in accordance with 6.7.

5.2.6.3 The supplier shall perform verification and validation in accordance with 6.4 and 6.5 respectively to demonstrate that the software products or services and processes fully satisfy their respective requirements.

5.2.6.4 The supplier shall make available to the acquirer the reports of evaluation, reviews, audits, testing, and problem resolutions as specified in the contract.

5.2.6.5 The supplier shall provide the acquirer access to the supplier's and subcontractors' facilities for review of software products or services as specified in the contract and project plans.

5.2.6.6 The supplier shall perform quality assurance activities in accordance with 6.3.

5.2.7 Delivery and completion. This activity consists of the following tasks:

5.2.7.1 The supplier shall deliver the software product or service as specified in the contract.

5.2.7.2 The supplier shall provide assistance to the acquirer in support of the delivered software product or service as specified in the contract.

5.3 Development process

The Development Process contains the activities and tasks of the developer. The process contains the activities for requirements analysis, design, coding, integration, testing, and installation and acceptance related to software products. It may contain system related activities if stipulated in the contract. The developer performs or supports the activities in this process in accordance with the contract.

The developer manages the Development Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4). When the developer is the supplier of the developed software product, the developer performs the Supply Process (5.2).

List of activities: This process consists of the following activities:

1) Process implementation;
2) System requirements analysis;
3) System architectural design;
4) Software requirements analysis;
5) Software architectural design;
6) Software detailed design;
7) Software coding and testing;
8) Software integration;
9) Software qualification testing;
10) System integration;
11) System qualification testing;
12) Software installation;
13) Software acceptance support.

5.3.1 Process implementation. This activity consists of the following tasks:

5.3.1.1 If not stipulated in the contract, the developer shall define or select a software life cycle model appropriate to the scope, magnitude, and complexity of the project. The activities and tasks of the Development Process shall be selected and mapped onto the life cycle model.

NOTE—These activities and tasks may overlap or interact and may be performed iteratively or recursively.

5.3.1.2 The developer shall:

a) Document the outputs in accordance with the Documentation Process (6.1).

b) Place the outputs under the Configuration Management Process (6.2) and perform change control in accordance with it.

c) Document and resolve problems and nonconformances found in the software products and tasks in accordance with the Problem Resolution Process (6.8).

d) Perform the supporting processes (clause 6) as specified in the contract.

5.3.1.3 The developer shall select, tailor, and use those standards, methods, tools, and computer programming languages (if not stipulated in the contract) that are documented, appropriate, and established by the organization for performing the activities of the Development Process and supporting processes (clause 6).

5.3.1.4 The developer shall develop plans for conducting the activities of the development process. The plans should include specific standards, methods, tools, actions, and responsibility associated with the development and qualification of all requirements including safety and security. If necessary, separate plans may be developed. These plans shall be documented and executed.

5.3.1.5 Non-deliverable items may be employed in the development or maintenance of the software product. However, it shall be ensured that the operation and maintenance of the deliverable software product after its delivery to the acquirer are independent of such items, otherwise those items should be considered as deliverable.

5.3.2 System requirements analysis. This activity consists of the following tasks, which the developer shall perform or support as required by the contract:

5.3.2.1 The specific intended use of the system to be developed shall be analyzed to specify system requirements. The system requirements specification shall describe: functions and capabilities of the system; business, organizational and user requirements; safety, security, human-factors engineering (ergonomics), interface, operations, and maintenance requirements; design constraints and qualification requirements. The system requirements specification shall be documented.

5.3.2.2 The system requirements shall be evaluated considering the criteria listed below. The results of evaluations shall be documented.

a) Traceability to acquisition needs;
b) Consistency with acquisition needs;
c) Testability;
d) Feasibility of system architectural design;
e) Feasibility of operation and maintenance.

5.3.3 System architectural design. This activity consists of the following tasks, which the developer shall perform or support as required by the contract:

5.3.3.1 A top-level architecture of the system shall be established. The architecture shall identify items of hardware, software, and manual-operations. It shall be ensured that all the system requirements are allocated among the items. Hardware configuration items, software configuration items, and manual operations shall be subsequently identified from these items. The system architecture and the system requirements allocated to the items shall be documented.

5.3.3.2 The system architecture and the requirements for the items shall be evaluated considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the system requirements;
b) Consistency with the system requirements;
c) Appropriateness of design standards and methods used;
d) Feasibility of the software items fulfilling their allocated requirements;
e) Feasibility of operation and maintenance.

5.3.4 Software requirements analysis. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.4.1 The developer shall establish and document software requirements, including the quality characteristics specifications, described below. Guidance for specifying quality characteristics may be found in ISO/IEC 9126.

a) Functional and capability specifications, including performance, physical characteristics, and environmental conditions under which the software item is to perform;

b) Interfaces external to the software item;

c) Qualification requirements;

d) Safety specifications, including those related to methods of operation and maintenance, environmental influences, and personnel injury;

e) Security specifications, including those related to compromise of sensitive information;

f) Human-factors engineering (ergonomics), including those related to manual operations, human-equipment interactions, constraints on personnel, and areas needing concentrated human attention, that are sensitive to human errors and training;

g) Data definition and database requirements;

h) Installation and acceptance requirements of the delivered software product at the operation and maintenance site(s);

i) User documentation;

j) User operation and execution requirements;

k) User maintenance requirements.

5.3.4.2 The developer shall evaluate the software requirements considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to system requirements and system design;
b) External consistency with system requirements;
c) Internal consistency;
d) Testability;
e) Feasibility of software design;
f) Feasibility of operation and maintenance.

5.3.4.3 The developer shall conduct joint review(s) in accordance with 6.6. Upon successful completion of the review(s), a baseline for the requirements of the software item shall be established.

5.3.5 Software architectural design. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.5.1 The developer shall transform the requirements for the software item into an architecture that describes its top-level structure and identifies the software components. It shall be ensured that all the requirements for the software item are allocated to its software components and further refined to facilitate detailed design. The architecture of the software item shall be documented.

5.3.5.2 The developer shall develop and document a top-level design for the interfaces external to the software item and between the software components of the software item.

5.3.5.3 The developer shall develop and document a top-level design for the database.

5.3.5.4 The developer should develop and document preliminary versions of user documentation.

5.3.5.5 The developer shall define and document preliminary test requirements and the schedule for Software Integration.

5.3.5.6 The developer shall evaluate the architecture of the software item and the interface and database designs considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the requirements of the software item;
b) External consistency with the requirements of the software item;
c) Internal consistency between the software components;
d) Appropriateness of design methods and standards used;
e) Feasibility of detailed design;
f) Feasibility of operation and maintenance.

5.3.5.7 The developer shall conduct joint review(s) in accordance with 6.6.

5.3.6 Software detailed design. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.6.1 The developer shall develop a detailed design for each software component of the software item. The software components shall be refined into lower levels containing software units that can be coded, compiled, and tested. It shall be ensured that all the software requirements are allocated from the software components to software units. The detailed design shall be documented.

5.3.6.2 The developer shall develop and document a detailed design for the interfaces external to the software item, between the software components, and between the software units. The detailed design of the interfaces shall permit coding without the need for further information.

5.3.6.3 The developer shall develop and document a detailed design for the database.

5.3.6.4 The developer shall update user documentation as necessary.

5.3.6.5 The developer shall define and document test requirements and schedule for testing software units. The test requirements should include stressing the software unit at the limits of its requirements.

5.3.6.6 The developer shall update the test requirements and the schedule for Software Integration.

5.3.6.7 The developer shall evaluate the software detailed design and test requirements considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the requirements of the software item;
b) External consistency with architectural design;
c) Internal consistency between software components and software units;
d) Appropriateness of design methods and standards used;
e) Feasibility of testing;
f) Feasibility of operation and maintenance.

5.3.6.8 The developer shall conduct joint review(s) in accordance with 6.6.

5.3.7 Software coding and testing. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.7.1 The developer shall develop and document the following:

a) Each software unit and database;
b) Test procedures and data for testing each software unit and database;

5.3.7.2 The developer shall test each software unit and database ensuring that it satisfies its requirements. The test results shall be documented.

5.3.7.3 The developer shall update the user documentation as necessary.

5.3.7.4 The developer shall update the test requirements and the schedule for Software Integration.

5.3.7.5 The developer shall evaluate software code and test results considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the requirements and design of the software item;
b) External consistency with the requirements and design of the software item;
c) Internal consistency between unit requirements;
d) Test coverage of units;
e) Appropriateness of coding methods and standards used;
f) Feasibility of software integration and testing;
g) Feasibility of operation and maintenance.

5.3.8 Software integration. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.8.1 The developer shall develop an integration plan to integrate the software units and software components into the software item. The plan shall include test requirements, procedures, data, responsibilities, and schedule. The plan shall be documented.

5.3.8.2 The developer shall integrate the software units and software components and test as the aggregates are developed in accordance with the integration plan. It shall be ensured that each aggregate satisfies the requirements of the software item and that the software item is integrated at the conclusion of the integration activity. The integration and test results shall be documented.

5.3.8.3 The developer shall update the user documentation as necessary.

5.3.8.4 The developer shall develop and document, for each qualification requirement of the software item, a set of tests, test cases (inputs, outputs, test criteria), and test procedures for conducting Software Qualification Testing. The developer shall ensure that the integrated software item is ready for Software Qualification Testing.

5.3.8.5 The developer shall evaluate the integration plan, design, code, tests, test results, and user documentation considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the system requirements;
b) External consistency with the system requirements;
c) Internal consistency;
d) Test coverage of the requirements of the software item;
e) Appropriateness of test standards and methods used;
f) Conformance to expected results;
g) Feasibility of software qualification testing;
h) Feasibility of operation and maintenance.

5.3.8.6 The developer shall conduct joint review(s) in accordance with 6.6.

5.3.9 Software qualification testing. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.9.1 The developer shall conduct qualification testing in accordance with the qualification requirements for the software item. It shall be ensured that the implementation of each software requirement is tested for compliance. The qualification testing results shall be documented.

5.3.9.2 The developer shall update the user documentation as necessary.

5.3.9.3 The developer shall evaluate the design, code, tests, test results, and user documentation considering the criteria listed below. The results of the evaluations shall be documented.

a) Test coverage of the requirements of the software item;
b) Conformance to expected results;
c) Feasibility of system integration and testing, if conducted ;
d) Feasibility of operation and maintenance.

5.3.9.4 The developer shall support audit(s) in accordance with 6.7. The results of the audits shall be documented. If both hardware and software are under development or integration, the audits may be postponed until the System Qualification Testing.

5.3.9.5 Upon successful completion of the audits, if conducted, the developer shall:

a) Update and prepare the deliverable software product for System Integration, System Qualification Testing, Software Installation, or Software Acceptance Support as applicable.
b) Establish a baseline for the design and code of the software item.

NOTE—The Software Qualification Testing may be used in the Verification Process (6.4) or the Validation Process (6.5).

5.3.10 System integration. This activity consists of the following tasks, which the developer shall perform or support as required by the contract.

5.3.10.1 The software configuration items shall be integrated, with hardware configuration items, manual operations, and other systems as necessary, into the system. The aggregates shall be tested, as they are developed, against their requirements. The integration and the test results shall be documented.

5.3.10.2 For each qualification requirement of the system, a set of tests, test cases (inputs, outputs, test criteria), and test procedures for conducting System Qualification Testing shall be developed and documented. The developer shall ensure that the integrated system is ready for System Qualification Testing.

5.3.10.3 The integrated system shall be evaluated considering the criteria listed below. The results of the evaluations shall be documented.

a) Test coverage of system requirements;
b) Appropriateness of test methods and standards used;
c) Conformance to expected results;
d) Feasibility of system qualification testing;
e) Feasibility of operation and maintenance.

5.3.11 System qualification testing. This activity consists of the following tasks, which the developer shall perform or support as required by the contract.

5.3.11.1 System qualification testing shall be conducted in accordance with the qualification requirements specified for the system. It shall be ensured that the implementation of each system requirement is tested for compliance and that the system is ready for delivery. The qualification testing results shall be documented.

5.3.11.2 The system shall be evaluated considering the criteria listed below. The results of the evaluations shall be documented.

a) Test coverage of system requirements;
b) Conformance to expected results;
c) Feasibility of operation and maintenance.

5.3.11.3 The developer shall support audit(s) in accordance with 6.7. The results of the audit(s) shall be documented.

NOTE—This subclause is not applicable to those software configuration items for which audits were conducted previously.

5.3.11.4 Upon successful completion of the audit(s), if conducted, the developer shall:

a) Update and prepare the deliverable software product for Software Installation and Software Acceptance Support.

b) Establish a baseline for the design and code of each software configuration item.

NOTE—The System Qualification Testing may be used in the Verification Process (6.4) or the Validation Process (6.5).

5.3.12 Software installation. This activity consists of the following tasks:

5.3.12.1 The developer shall develop a plan to install the software product in the target environment as designated in the contract. The resources and information necessary to install the software product shall be determined and be available. As specified in the contract, the developer shall assist the acquirer with the set-up activities. Where the installed software product is replacing an existing system, the developer shall support any parallel running activities that are required by contract. The installation plan shall be documented.

5.3.12.2 The developer shall install the software product in accordance with the installation plan. It shall be ensured that the software code and databases initialize, execute, and terminate as specified in the contract. The installation events and results shall be documented.

5.3.13 Software acceptance support. This activity consists of the following tasks:

5.3.13.1 The developer shall support the acquirer's acceptance review and testing of the software product. Acceptance review and testing shall consider the results of the Joint Reviews (6.6), Audits (6.7), Software Qualification Testing, and System Qualification Testing (if performed). The results of the acceptance review and testing shall be documented.

5.3.13.2 The developer shall complete and deliver the software product as specified in the contract.

5.3.13.3 The developer shall provide initial and continuing training and support to the acquirer as specified in the contract.

5.4 Operation process

The Operation Process contains the activities and tasks of the operator. The process covers the operation of the software product and operational support to users. Because operation of software product is integrated into the operation of the system, the activities and tasks of this process refer to the system.

The operator manages the Operation Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4). When the operator is the supplier of the operation service, the operator performs the Supply Process (5.2).

List of activities. This process consists of the following activities:

1) Process implementation;
2) Operational testing;
3) System operation;
4) User support.

5.4.1 Process implementation. This activity consists of the following tasks:

5.4.1.1 The operator shall develop a plan and set operational standards for performing the activities and tasks of this process. The plan shall be documented and executed.

5.4.1.2 The operator shall establish procedures for receiving, recording, resolving, tracking problems, and providing feedback. Whenever problems are encountered, they shall be recorded and entered into the Problem Resolution Process (6.8).

5.4.1.3 The operator shall establish procedures for testing the software product in its operation environment, for entering problem reports and modification requests to the Maintenance Process (5.5), and for releasing the software product for operational use.

5.4.2 Operational testing. This activity consists of the following tasks:

5.4.2.1 For each release of the software product, the operator shall perform operational testing, and, on satisfying the specified criteria, release the software product for operational use.

5.4.2.2 The operator shall ensure that the software code and databases initialize, execute, and terminate as described in the plan.

5.4.3 System operation. This activity consists of the following task:

5.4.3.1 The system shall be operated in its intended environment according to the user documentation.

5.4.4 User support. This activity consists of the following tasks:

5.4.4.1 The operator shall provide assistance and consultation to the users as requested. These requests and subsequent actions shall be recorded and monitored.

5.4.4.2 The operator shall forward user requests, as necessary, to the Maintenance Process (clause 5.5) for resolution. These requests shall be addressed and the actions that are planned and taken shall be reported to the originators of the requests. All resolutions shall be monitored to conclusion.

5.4.4.3 If a reported problem has a temporary work-around before a permanent solution can be released, the originator of the problem report shall be given the option to use it. Permanent corrections, releases that include previously omitted functions or features, and system improvements shall be applied to the operational software product using the Maintenance Process (5.5).

5.5 Maintenance process

The Maintenance Process contains the activities and tasks of the maintainer. This process is activated when the software product undergoes modifications to code and associated documentation due to a problem or the need for improvement or adaptation. The objective is to modify existing software product while preserving its integrity. This process includes the migration and retirement of the software product. The process ends with the retirement of the software product.

The activities provided in this clause are specific to the Maintenance Process; however, the process may utilize other processes in this International Standard. If the Development Process (5.3) is utilized, the term developer there is interpreted as maintainer.

The maintainer manages the Maintenance Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4). When the maintainer is the supplier of the maintenance service, the maintainer performs the Supply Process (5.2).

List of activities. This process consists of the following activities:

1) Process implementation;
2) Problem and modification analysis;
3) Modification implementation;
4) Maintenance review/acceptance;
5) Migration;
6) Software retirement.

5.5.1 Process implementation. This activity consists of the following tasks:

5.5.1.1 The maintainer shall develop, document, and execute plans and procedures for conducting the activities and tasks of the Maintenance Process.

5.5.1.2 The maintainer shall establish procedures for receiving, recording, and tracking problem reports and modification requests from the users and providing feedback to the users. Whenever problems are encountered, they shall be recorded and entered into the Problem Resolution Process (6.8).

5.5.1.3 The maintainer shall implement (or establish organizational interface with) the Configuration Management Process (6.2) for managing modifications to the existing system.

5.5.2 Problem and modification analysis. This activity consists of the following tasks:

5.5.2.1 The maintainer shall analyze the problem report or modification request for its impact on the organization, the existing system, and the interfacing systems for the following:

a) Type; for example, corrective, improvement, preventive, or adaptive to new environment;
b) Scope; for example, size of modification, cost involved, time to modify;
c) Criticality; for example, impact on performance, safety, or security.

5.5.2.2 The maintainer shall replicate or verify the problem.

5.5.2.3 Based upon the analysis, the maintainer shall consider options for implementing the modification.

5.5.2.4 The maintainer shall document the problem/modification request, the analysis results, and implementation options.

5.5.2.5 The maintainer shall obtain approval for the selected modification option as specified in the contract.

5.5.3 Modification implementation. This activity consists of the following tasks:

5.5.3.1 The maintainer shall conduct analysis and determine which documentation, software units, and versions thereof need to be modified. These shall be documented.

5.5.3.2 The maintainer shall enter the Development Process (5.3) to implement the modifications. The requirements of the Development Process shall be supplemented as follows:

a) Test and evaluation criteria for testing and evaluating the modified and the unmodified parts (software units, components, and configuration items) of the system shall be defined and documented.

b) The complete and correct implementation of the new and modified requirements shall be ensured. It also shall be ensured that the original, unmodified requirements were not affected. The test results shall be documented.

5.5.4 Maintenance review/acceptance. This activity consists of the following tasks:

5.5.4.1 The maintainer shall conduct review(s) with the organization authorizing the modification to determine the integrity of the modified system.

5.5.4.2 The maintainer shall obtain approval for the satisfactory completion of the modification as specified in the contract.

5.5.5 Migration. This activity consists of the following tasks:

5.5.5.1 If a system or software product (including data) is migrated from an old to a new operational environment, it shall be ensured that any software product or data produced or modified during migration are in accordance with this International Standard.

5.5.5.2 A migration plan shall be developed, documented, and executed. The planning activities shall include users. Items included in the plan shall include the following:

a) Requirements analysis and definition of migration;
b) Development of migration tools;
c) Conversion of software product and data;
d) Migration execution;
e) Migration verification;
f) Support for the old environment in the future.

5.5.5.3 Users shall be given notification of the migration plans and activities. Notifications shall include the following:

a) Statement of why the old environment is no longer to be supported;

b) Description of the new environment with its date of availability;

c) Description of other support options available, if any, once support for the old environment has been removed.

5.5.5.4 Parallel operations of the old and new environments may be conducted for smooth transition to the new environment. During this period, necessary training shall be provided as specified in the contract.

5.5.5.5 When the scheduled migration arrives, notification shall be sent to all concerned. All associated old environment's documentation, logs, and code should be placed in archives.

5.5.5.6 A post-operation review shall be performed to assess the impact of changing to the new environment. The results of the review shall be sent to the appropriate authorities for information, guidance, and action.

5.5.5.7 Data used by or associated with the old environment shall be accessible in accordance with the contract requirements for data protection and audit applicable to the data.

5.5.6 Software retirement. This activity consists of the following tasks:

NOTE—The software product will be retired on the request of the owner.

5.5.6.1 A retirement plan to remove active support by the operation and maintenance organizations shall be developed and documented. The planning activities shall include users. The plan shall address the items listed below. The plan shall be executed.

a) Cessation of full or partial support after a certain period of time;
b) Archiving of the software product and its associated documentation;
c) Responsibility for any future residual support issues;
d) Transition to new software product, if applicable;
e) Accessibility of archive copies of data.

5.5.6.2 Users shall be given notification of the retirement plans and activities. Notifications shall include the following:

a) Description of the replacement or upgrade with its date of availability;
b) Statement of why the software product is no longer to be supported;
c) Description of other support options available, once support has been removed.

5.5.6.3 Parallel operations of the retiring and the new software product should be conducted for smooth transition to the new system. During this period, user training shall be provided as specified in the contract.

5.5.6.4 When the scheduled retirement arrives, notification shall be sent to all concerned. All associated development documentation, logs, and code should be placed in archives, when appropriate.

5.5.6.5 Data used or associated by the retired software product shall be accessible in accordance with the contract requirements for data protection and audit applicable to the data.

6 Supporting life cycle processes

This clause defines the following supporting life cycle processes:

1) Documentation process;
2) Configuration management process;
3) Quality assurance process;
4) Verification process;
5) Validation process;
6) Joint review process;
7) Audit process;
8) Problem resolution process.

The activities and tasks in a supporting process are the responsibility of the organization performing that process. This organization ensures that the process is in existence and functional.

The organization employing and performing a supporting process manages it at the project level following the Management Process (7.1); establishes an infrastructure under it following the Infrastructure Process (7.2); tailors it for the project following the Tailoring Process (annex A); and manages it at the organizational level following the Improvement Process (7.3) and the Training Process (7.4). Joint Reviews, Audits, Verification, and Validation may be used as techniques of Quality Assurance.

6.1 Documentation process

The Documentation Process is a process for recording information produced by a life cycle process or activity. The process contains the set of activities, which plan, design, develop, produce, edit, distribute, and maintain those documents needed by all concerned such as managers, engineers, and users of the system or software product.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Design and development;
3) Production;
4) Maintenance.

6.1.1 Process implementation. This activity consists of the following task:

6.1.1.1 A plan, identifying the documents to be produced during the life cycle of the software product, shall be developed, documented, and implemented. For each identified document, the following shall be addressed:

a) Title or Name;

b) Purpose;

c) Intended audience;

d) Procedures and responsibilities for inputs, development, review, modification, approval, production, storage, distribution, maintenance, and configuration management;

e) Schedule for intermediate and final versions.

6.1.2 Design and development. This activity consists of the following tasks:

6.1.2.1 Each identified document shall be designed in accordance with applicable documentation standards for format, content description, page numbering, figure/table placement, proprietary/security marking, packaging, and other presentation items.

6.1.2.2 The source and appropriateness of input data for the documents shall be confirmed. Automated documentation tools may be used.

6.1.2.3 The prepared documents shall be reviewed and edited for format, technical content, and presentation style against their documentation standards. They shall be approved for adequacy by authorized personnel prior to issue.

6.1.3 Production. This activity consists of the following tasks:

6.1.3.1 The documents shall be produced and provided in accordance with the plan. Production and distribution of documents may use paper, electronic, or other media. Master materials shall be stored in accordance with requirements for record retention, security, maintenance, and backup.

6.1.3.2 Controls shall be established in accordance with the Configuration Management Process (6.2).

6.1.4 Maintenance. This activity consists of the following task:

6.1.4.1 The tasks, that are required to be performed when documentation is to be modified, shall be performed (see 5.5). For those documents that are under configuration management, modifications shall be managed in accordance with the Configuration Management Process (6.2).

6.2 Configuration management process

The Configuration Management Process is a process of applying administrative and technical procedures throughout the software life cycle to: identify, define, and baseline software items in a system; control modifications and releases of the items; record and report the status of the items and modification requests; ensure the completeness, consistency, and correctness of the items; and control storage, handling, and delivery of the items.

NOTE—When this process is employed on other software products or entities, the term "software item" below is interpreted accordingly.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Configuration identification;
3) Configuration control;
4) Configuration status accounting;
5) Configuration evaluation;
6) Release management and delivery.

6.2.1 Process implementation. This activity consists of the following task:

6.2.1.1 A configuration management plan shall be developed. The plan shall describe: the configuration management activities; procedures and schedule for performing these activities; the organization(s) responsible for performing these activities; and their relationship with other organizations, such as software development or maintenance. The plan shall be documented and implemented.

NOTE—The plan may be a part of the system configuration management plan.

6.2.2 Configuration identification. This activity consists of the following task:

6.2.2.1 A scheme shall be established for the identification of software items and their versions to be controlled for the project. For each software configuration item and its versions, the following shall be identified: the documentation that establishes the baseline; the version references; and other identification details.

6.2.3 Configuration control. This activity consists of the following task:

6.2.3.1 The following shall be performed: identification and recording of change requests; analysis and evaluation of the changes; approval or disapproval of the request; and implementation, verification, and release of the modified software item. An audit trail shall exist, whereby each modification, the reason for the modification, and authorization of the modification can be traced. Control and audit of all accesses to the controlled software items that handle safety or security critical functions shall be performed.

6.2.4 Configuration status accounting. This activity consists of the following task:

6.2.4.1 Management records and status reports that show the status and history of controlled software items including baseline shall be prepared. Status reports should include the number of changes for a project, latest software item versions, release identifiers, the number of releases, and comparisons of releases.

6.2.5 Configuration evaluation. This activity consists of the following task:

6.2.5.1 The following shall be determined and ensured: the functional completeness of the software items against their requirements and the physical completeness of the software items (whether their design and code reflect an up-to-date technical description).

6.2.6 Release management and delivery. This activity consists of the following task:

6.2.6.1 The release and delivery of software products and documentation shall be formally controlled. Master copies of code and documentation shall be maintained for the life of the software product. The code and documentation that contain safety or security critical functions shall be handled, stored, packaged, and delivered in accordance with the policies of the organizations involved.

6.3 Quality assurance process

The Quality Assurance Process is a process for providing adequate assurance that the software products and processes in the project life cycle conform to their specified requirements and adhere to their established plans. To be unbiased, quality assurance needs to have organizational freedom and authority from persons directly responsible for developing the software product or executing the process in the project. Quality assurance may be internal or external depending on whether evidence of product or process quality is demonstrated to the management of the supplier or the acquirer. Quality assurance may make use of the results of other supporting processes, such as Verification, Validation, Joint Reviews, Audits, and Problem Resolution.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Product assurance;
3) Process assurance;
4) Assurance of quality systems.

6.3.1 Process implementation. This activity consists of the following tasks:

6.3.1.1 A quality assurance process tailored to the project shall be established. The objectives of the quality assurance process shall be to assure that the software products and the processes employed for providing those software products comply with their established requirements and adhere to their established plans.

6.3.1.2 The quality assurance process should be coordinated with the related Verification (6.4), Validation (6.5), Joint Review (6.6), and Audit (6.7) Processes.

6.3.1.3 A plan for conducting the quality assurance process activities and tasks shall be developed, documented, implemented, and maintained for the life of the contract. The plan shall include the following:

a) Quality standards, methodologies, procedures, and tools for performing the quality assurance activities (or their references in organization's official documentation);

b) Procedures for contract review and coordination thereof;

c) Procedures for identification, collection, filing, maintenance, and disposition of quality records;

d) Resources, schedule, and responsibilities for conducting the quality assurance activities;

e) Selected activities and tasks from supporting processes, such as Verification (6.4), Validation (6.5), Joint Review (6.6), Audit (6.7), and Problem Resolution (6.8).

6.3.1.4 Scheduled and on-going quality assurance activities and tasks shall be executed. When problems or nonconformances with contract requirements are detected, they shall be documented and serve as input to the Problem Resolution Process (6.8). Records of these activities and tasks, their execution, problems, and problem resolutions shall be prepared and maintained.

6.3.1.5 Records of quality assurance activities and tasks shall be made available to the acquirer as specified in the contract.

6.3.1.6 It shall be assured that persons responsible for assuring compliance with the contract requirements have the organizational freedom, resources, and authority to permit objective evaluations and to initiate, effect, resolve, and verify problem resolutions.

6.3.2 Product assurance. This activity consists of the following tasks:

6.3.2.1 It shall be assured that all the plans required by the contract are documented, comply with the contract, are mutually consistent, and are being executed as required.

6.3.2.2 It shall be assured that software products and related documentation comply with the contract and adhere to the plans.

6.3.2.3 In preparation for the delivery of the software products, it shall be assured that they have fully satisfied their contractual requirements and are acceptable to the acquirer.

6.3.3 Process assurance. This activity consists of the following tasks:

6.3.3.1 It shall be assured that those software life cycle processes (supply, development, operation, maintenance, and supporting processes including quality assurance) employed for the project comply with the contract and adhere to the plans.

6.3.3.2 It shall be assured that the internal software engineering practices, development environment, test environment, and libraries comply with the contract.

6.3.3.3 It shall be assured that applicable prime-contract requirements are passed down to the subcontractor, and that the subcontractor's software products satisfy prime-contract requirements.

6.3.3.4 It shall be assured that the acquirer and other parties are provided the required support and cooperation in accordance with the contract, negotiations, and plans.

6.3.3.5 It should be assured that software product and process measurements are in accordance with established standards and procedures.

6.3.3.6 It shall be assured that the staff assigned have the skill and knowledge needed to meet the requirements of the project and receive any necessary training.

6.3.4 Assurance of quality systems. This activity consists of the following task:

6.3.4.1 Additional quality management activities shall be assured in accordance with the clauses of ISO 9001 as specified in the contract.

6.4 Verification process

The Verification Process is a process for determining whether the software products of an activity fulfill the requirements or conditions imposed on them in the previous activities. For cost and performance effectiveness, verification should be integrated, as early as possible, with the process (such as supply, development, operation, or maintenance) that employs it. This process may include analysis, review and test.

This process may be executed with varying degrees of independence. The degree of independence may range from the same person or different person in the same organization to a person in a different organization with varying degrees of separation. In the case where the process is executed by an organization independent of the supplier, developer, operator, or maintainer, it is called Independent Verification Process.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Verification.

6.4.1 Process implementation. This activity consists of the following tasks:

6.4.1.1 A determination shall be made if the project warrants a verification effort and the degree of organizational independence of that effort needed. The project requirements shall be analyzed for criticality. Criticality may be gauged in terms of:

a) The potential of an undetected error in a system or software requirement for causing death or personal injury, mission failure, or financial or catastrophic equipment loss or damage;

b) The maturity of and risks associated with the software technology to be used;

c) Availability of funds and resources.

6.4.1.2 If the project warrants a verification effort, a verification process shall be established to verify the software product.

6.4.1.3 If the project warrants an independent verification effort, a qualified organization responsible for conducting the verification shall be selected. This organization shall be assured of the independence and authority to perform the verification activities.

6.4.1.4 Based upon the scope, magnitude, complexity, and criticality analysis above, target life cycle activities and software products requiring verification shall be determined. Verification activities and tasks defined in 6.4.2, including associated methods, techniques, and tools for performing the tasks, shall be selected for the target life cycle activities and software products.

6.4.1.5 Based upon the verification tasks as determined, a verification plan shall be developed and documented. The plan shall address the life cycle activities and software products subject to verification, the required verification tasks for each life cycle activity and software product, and related resources, responsibilities, and schedule. The plan shall address procedures for forwarding verification reports to the acquirer and other involved organizations.

6.4.1.6 The verification plan shall be implemented. Problems and nonconformances detected by the verification effort shall be entered into the Problem Resolution Process (6.8). All problems and nonconformances shall be resolved. Results of the verification activities shall be made available to the acquirer and other involved organizations.

6.4.2 Verification. This activity consists of the following tasks:

6.4.2.1 Contract verification. The contract shall be verified considering the criteria listed below:

a) The supplier has the capability to satisfy the requirements.

b) The requirements are consistent and cover user needs.

c) Adequate procedures for handling changes to requirements and escalating problems are stipulated.

d) Procedures and their extent for interface and cooperation among the parties are stipulated, including ownership, warranty, copyright and confidentiality.

e) Acceptance criteria and procedures are stipulated in accordance with requirements.

NOTE—This activity may be used in the contract review (see 6.3.1.3 b).

6.4.2.2 Process verification. The process shall be verified considering the criteria listed below:

a) Project planning requirements are adequate and timely.

b) Processes selected for the project are adequate, implemented, being executed as planned, and compliant with the contract.

c) The standards, procedures, and environments for the project's processes are adequate.

d) The project is staffed and personnel trained as required by the contract.

6.4.2.3 Requirements verification. The requirements shall be verified considering the criteria listed below:

a) The system requirements are consistent, feasible, and testable.

b) The system requirements have been appropriately allocated to hardware items, software items, and manual operations according to design criteria.

c) The software requirements are consistent, feasible, testable, and accurately reflect system requirements.

d) The software requirements related to safety, security, and criticality are correct as shown by suitably rigorous methods.

6.4.2.4 Design verification. The design shall be verified considering the criteria listed below:

a) The design is correct and consistent with and traceable to requirements.

b) The design implements proper sequence of events, inputs, outputs, interfaces, logic flow, allocation of timing and sizing budgets, and error definition, isolation, and recovery.

c) Selected design can be derived from requirements.

d) The design implements safety, security, and other critical requirements correctly as shown by suitably rigorous methods.

6.4.2.5 Code verification. The code shall be verified considering the criteria listed below:

a) The code is traceable to design and requirements, testable, correct, and compliant with requirements and coding standards.

b) The code implements proper event sequence, consistent interfaces, correct data and control flow, completeness, appropriate allocation timing and sizing budgets, and error definition, isolation, and recovery.

c) Selected code can be derived from design or requirements.

d) The code implements safety, security, and other critical requirements correctly as shown by suitably rigorous methods.

6.4.2.6 Integration verification. The integration shall be verified considering the criteria listed below:

a) The software components and units of each software item have been completely and correctly integrated into the software item.

b) The hardware items, software items, and manual operations of the system have been completely and correctly integrated into the system.

c) The integration tasks have been performed in accordance with an integration plan.

6.4.2.7 Documentation verification. The documentation shall be verified considering the criteria listed below:

a) The documentation is adequate, complete, and consistent.
b) Documentation preparation is timely.
c) Configuration management of documents follows specified procedures.

6.5 Validation process

The Validation Process is a process for determining whether the requirements and the final, as-built system or software product fulfills its specific intended use. Validation may be conducted in earlier stages. This process may be conducted as a part of Software Acceptance Support (5.3.13).

This process may be executed with varying degrees of independence. The degree of independence may range from the same person or different person in the same organization to a person in a different organization with varying degrees of separation. In the case where the process is executed by an organization independent of the supplier, developer, operator, or maintainer, it is called Independent Validation Process.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Validation.

6.5.1 Process implementation. This activity consists of the following tasks:

6.5.1.1 A determination shall be made if the project warrants a validation effort and the degree of organizational independence of that effort needed.

6.5.1.2 If the project warrants a validation effort, a validation process shall be established to validate the system or software product. Validation tasks defined below, including associated methods, techniques, and tools for performing the tasks, shall be selected.

6.5.1.3 If the project warrants an independent effort, a qualified organization responsible for conducting the effort shall be selected. The conductor shall be assured of the independence and authority to perform the validation tasks.

6.5.1.4 A validation plan shall be developed and documented. The plan shall include, but is not limited to, the following:

a) Items subject to validation;
b) Validation tasks to be performed;
c) Resources, responsibilities, and schedule for validation;
d) Procedures for forwarding validation reports to the acquirer and other parties.

6.5.1.5 The validation plan shall be implemented. Problems and nonconformances detected by the validation effort shall be entered into the Problem Resolution Process (6.8). All problems and nonconformances shall be resolved. Results of the validation activities shall be made available to the acquirer and other involved organizations.

6.5.2 Validation. This activity shall consist of the following tasks:

6.5.2.1 Prepare selected test requirements, test cases, and test specifications for analyzing test results.

6.5.2.2 Ensure that these test requirements, test cases, and test specifications reflect the particular requirements for the specific intended use.

6.5.2.3 Conduct the tests in subclauses 6.5.2.1 and 6.5.2.2, including:

a) Testing with stress, boundary, and singular inputs;
b) Testing the software product for its ability to isolate and minimize the effect of errors; that is, graceful degradation upon failure, request for operator assistance upon stress, boundary, and singular conditions;
c) Testing that representative users can successfully achieve their intended tasks using the software product.

6.5.2.4 Validate that the software product satisfies its intended use.

6.5.2.5 Test the software product as appropriate in selected areas of the target environment.

6.6 Joint review process

The Joint Review Process is a process for evaluating the status and products of an activity of a project as appropriate. Joint reviews are at both project management and technical levels and are held throughout the life of the contract. This process may be employed by any two parties, where one party (reviewing party) reviews another party (reviewed party).

List of activities: This process consists of the following activities:

1) Process implementation;
2) Project management reviews;
3) Technical reviews.

6.6.1 Process implementation. This activity consists of the following tasks:

6.6.1.1 Periodic reviews shall be held at predetermined milestones as specified in the project plan(s). *Ad hoc* reviews should be called when deemed necessary by either party.

6.6.1.2 All resources required to conduct the reviews shall be agreed on by the parties. These resources include personnel, location, facilities, hardware, software, and tools.

6.6.1.3 The parties should agree on the following items at each review: meeting agenda, software products (results of an activity) and problems to be reviewed; scope and procedures; and entry and exit criteria for the review.

6.6.1.4 Problems detected during the reviews shall be recorded and entered into the Problem Resolution Process (6.8) as required.

6.6.1.5 The review results shall be documented and distributed. The reviewing party will acknowledge to the reviewed party the adequacy (for example, approval, disapproval, or contingent approval) of the review results.

6.6.1.6 The parties shall agree on the outcome of the review and any action item responsibilities and closure criteria.

6.6.2 Project management reviews. This activity consists of the following task:

6.6.2.1 Project status shall be evaluated relative to the applicable project plans, schedules, standards, and guidelines. The outcome of the review should be discussed between the two parties and should provide for the following:

a) Making activities progress according to plan, based on an evaluation of the activity or software product status;

b) Maintaining global control of the project through adequate allocation of resources;

c) Changing project direction or determining the need for alternate planning;

d) Evaluating and managing the risk issues that may jeopardize the success of the project.

6.6.3 Technical reviews. This activity consists of the following task:

6.6.3.1 Technical reviews shall be held to evaluate the software products or services under consideration and provide evidence that:

a) They are complete.

b) They comply with their standards and specifications.

c) Changes to them are properly implemented and affect only those areas identified by the Configuration Management Process (6.2).

d) They are adhering to applicable schedules.

e) They are ready for the next activity.

f) The development, operation, or maintenance is being conducted according to the plans, schedules, standards, and guidelines of the project.

6.7 Audit process

The Audit Process is a process for determining compliance with the requirements, plans, and contract as appropriate. This process may be employed by any two parties, where one party (auditing party) audits the software products or activities of another party (audited party).

List of activities. This process consists of the following activities:

1) Process implementation;
2) Audit.

6.7.1 Process implementation. This activity consists of the following tasks:

6.7.1.1 Audits shall be held at predetermined milestones as specified in the project plan(s).

6.7.1.2 Auditing personnel shall not have any direct responsibility for the software products and activities they audit.

6.7.1.3 All resources required to conduct the audits shall be agreed by the parties. These resources include supporting personnel, location, facilities, hardware, software, and tools.

6.7.1.4 The parties should agree on the following items at each audit: agenda; software products (and results of an activity) to be reviewed; audit scope and procedures; and entry and exit criteria for the audit.

6.7.1.5 Problems detected during the audits shall be recorded and entered into the Problem Resolution Process (6.8) as required.

6.7.1.6 After completing an audit, the audit results shall be documented and provided to the audited party. The audited party shall acknowledge to the auditing party any problems found in the audit and related problem resolutions planned.

6.7.1.7 The parties shall agree on the outcome of the audit and any action item responsibilities and closure criteria.

6.7.2 Audit. This activity consists of the following task:

6.7.2.1 Audits shall be conducted to ensure that:

a) As-coded software products (such as a software item) reflect the design documentation.

b) The acceptance review and testing requirements prescribed by the documentation are adequate for the acceptance of the software products.

c) Test data comply with the specification.

d) Software products were successfully tested and meet their specifications.

e) Test reports are correct and discrepancies between actual and expected results have been resolved.

f) User documentation complies with standards as specified.

g) Activities have been conducted according to applicable requirements, plans, and contract.

h) The costs and schedules adhere to the established plans.

6.8 Problem resolution process

The Problem Resolution Process is a process for analyzing and resolving the problems (including nonconformances), whatever their nature or source, that are discovered during the execution of development, operation, maintenance, or other processes. The objective is to provide a timely, responsible, and documented means to ensure that all discovered problems are analyzed and resolved and trends are recognized.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Problem resolution.

6.8.1 Process implementation. This activity consists of the following task:

6.8.1.1 A problem resolution process shall be established for handling all problems (including nonconformances) detected in the software products and activities. The process shall comply with the following requirements:

a) The process shall be closed-loop, ensuring that: all detected problems are promptly reported and entered into the Problem Resolution Process; action is initiated on them; relevant parties are advised of the existence of the problem as appropriate; causes are identified, analyzed, and, where possible, eliminated; resolution and disposition are achieved; status is tracked and reported; and records of the problems are maintained as stipulated in the contract.

b) The process should contain a scheme for categorizing and prioritizing the problems. Each problem should be classified by the category and priority to facilitate trend analysis and problem resolution.

c) Analysis shall be performed to detect trends in the problems reported.

d) Problem resolutions and dispositions shall be evaluated: to evaluate that problems have been resolved, adverse trends have been reversed, and changes have been correctly implemented in the appropriate software products and activities; and to determine whether additional problems have been introduced.

6.8.2 Problem resolution. This activity consists of the following task:

6.8.2.1 When problems (including nonconformances) have been detected in a software product or an activity, a problem report shall be prepared to describe each problem detected. The problem report shall be used as part of the closed-loop process described above: from detection of the problem, through investigation, analysis and resolution of the problem and its cause, and onto trend detection across problems.

7 Organizational life cycle processes

This clause defines the following organizational life cycle processes:

1) Management process;
2) Infrastructure process;
3) Improvement process;
4) Training process.

The activities and tasks in an organizational process are the responsibility of the organization using that process. The organization ensures that the process is in existence and functional.

7.1 Management process

The Management Process contains the generic activities and tasks, which may be employed by any party that has to manage its respective process(es). The manager is responsible for product management, project management, and task management of the applicable process(es), such as the acquisition, supply, development, operation, maintenance, or supporting process.

List of activities: This process consists of the following activities:

1) Initiation and scope definition;
2) Planning;
3) Execution and control;
4) Review and evaluation;
5) Closure.

7.1.1 Initiation and scope definition. This activity consists of the following tasks:

7.1.1.1 The management process shall be initiated by establishing the requirements of the process to be undertaken.

7.1.1.2 Once the requirements are established, the manager shall establish the feasibility of the process by checking that the resources (personnel, materials, technology, and environment) required to execute and manage the process are available, adequate, and appropriate and that the time-scales to completion are achievable.

7.1.1.3 As necessary, and by agreement of all parties concerned, the requirements of the process may be modified at this point to achieve the completion criteria.

7.1.2 Planning. This activity consists of the following task:

7.1.2.1 The manager shall prepare the plans for execution of the process. The plans associated with the execution of the process shall contain descriptions of the associated activities and tasks and identification of the software products that will be provided. These plans shall include, but are not limited to, the following:

a) Schedules for the timely completion of tasks;
b) Estimation of effort;
c) Adequate resources needed to execute the tasks;
d) Allocation of tasks;
e) Assignment of responsibilities;
f) Quantification of risks associated with the tasks or the process itself;
g) Quality control measures to be employed throughout the process;
h) Costs associated with the process execution;
i) Provision of environment and infrastructure.

7.1.3 Execution and control. This activity consists of the following tasks:

7.1.3.1 The manager shall initiate the implementation of the plan to satisfy the objectives and criteria set, exercising control over the process.

7.1.3.2 The manager shall monitor the execution of the process, providing both internal reporting of the process progress and external reporting to the acquirer as defined in the contract.

7.1.3.3 The manager shall investigate, analyze, and resolve the problems discovered during the execution of the process. The resolution of problems may result in changes to plans. It is the manager's responsibility to ensure the impact of any changes is determined, controlled, and monitored. Problems and their resolution shall be documented.

7.1.3.4 The manager shall report, at agreed points, the progress of the process, declaring adherence to the plans and resolving instances of the lack of progress. These include internal and external reporting as required by the organizational procedures and the contract.

7.1.4 Review and evaluation. This activity consists of the following tasks:

7.1.4.1 The manager shall ensure that the software products and plans are evaluated for satisfaction of requirements.

7.1.4.2 The manager shall assess the evaluation results of the software products, activities, and tasks completed during the execution of the process for achievement of the objectives and completion of the plans.

7.1.5 Closure. This activity consists of the following tasks:

7.1.5.1 When all software products, activities, and tasks are completed, the manager shall determine whether the process is complete taking into account the criteria as specified in the contract or as part of organization's procedure.

7.1.5.2 The manager shall check the results and records of the software products, activities, and tasks employed for completeness. These results and records shall be archived in a suitable environment as specified in the contract.

7.2 Infrastructure process

The Infrastructure Process is a process to establish and maintain the infrastructure needed for any other process. The infrastructure may include hardware, software, tools, techniques, standards, and facilities for development, operation, or maintenance.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Establishment of the infrastructure;
3) Maintenance of the infrastructure.

7.2.1 Process implementation. This activity consists of the following tasks:

7.2.1.1 The infrastructure should be defined and documented to meet the requirements of the process employing this process, considering the applicable procedures, standards, tools, and techniques.

7.2.1.2 The establishment of the infrastructure should be planned and documented.

7.2.2 Establishment of the infrastructure. This activity consists of the following tasks:

7.2.2.1 The configuration of the infrastructure should be planned and documented. Functionality, performance, safety, security, availability, space requirements, equipment, costs, and time constraints should be considered.

7.2.2.2 The infrastructure shall be installed in time for execution of the relevant process.

7.2.3 Maintenance of the infrastructure. This activity consists of the following task:

7.2.3.1 The infrastructure shall be maintained, monitored, and modified as necessary to ensure that it continues to satisfy the requirements of the process employing this process. As part of maintaining the infrastructure, the extent to which the infrastructure is under configuration management shall be defined.

7.3 Improvement process

The Improvement Process is a process for establishing, assessing, measuring, controlling, and improving a software life cycle process.

List of activities. This process consists of the following activities:

1) Process establishment;
2) Process assessment;
3) Process improvement.

7.3.1 Process establishment. This activity consists of the following task:

7.3.1.1 The organization shall establish a suite of organizational processes for all software life cycle processes as they apply to its business activities. The processes and their application to specific cases shall be documented in organization's publications. As appropriate, a process control mechanism should be established to develop, monitor, control, and improve the process(es).

7.3.2 Process assessment. This activity consists of the following tasks:

7.3.2.1 A process assessment procedure should be developed, documented, and applied. Assessment records should be kept and maintained.

7.3.2.2 The organization shall plan and carry out review of the processes at appropriate intervals to assure their continuing suitability and effectiveness in the light of assessment results.

7.3.3 Process improvement. This activity consists of the following tasks:

7.3.3.1 The organization shall effect such improvements to its processes as it determines to be necessary as a result of process assessment and review. Process documentation should be updated to reflect improvement in the organizational processes.

7.3.3.2 Historical, technical, and evaluation data should be collected and analyzed to gain an understanding of the strengths and weaknesses of the employed processes. These analyses should be used as feedback to improve these processes, to recommend changes in the direction of the projects (or subsequent projects), and to determine technology advancement needs.

7.3.3.3 Quality cost data should be collected, maintained, and used to improve the organization's processes as a management activity. These data shall serve the purpose of establishing the cost of both the prevention and resolution of problems and non-conformity in software products and services.

7.4 Training process

The Training Process is a process for providing and maintaining trained personnel. The acquisition, supply, development, operation, or maintenance of software products is largely dependent upon knowledgeable and skilled personnel. For example: developer personnel should have essential training in software management and software engineering. It is, therefore, imperative that personnel training be planned and implemented early so that trained personnel are available as the software product is acquired, supplied, developed, operated, or maintained.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Training material development;
3) Training plan implementation.

7.4.1 Process implementation. This activity consists of the following task:

7.4.1.1 A review of the project requirements shall be conducted to establish and make timely provision for acquiring or developing the resources and skills required by the management and technical staff. The types and levels of training and categories of personnel needing training shall be determined. A training plan, addressing implementation schedules, resource requirements, and training needs, should be developed and documented.

7.4.2 Training material development. This activity consists of the following task:

7.4.2.1 Training manuals, including presentation materials used in providing training, should be developed.

7.4.3 Training plan implementation. This activity consists of the following tasks:

7.4.3.1 The training plan shall be implemented to provide training to personnel. Training records should be maintained.

7.4.3.2 It should be ensured that the right mix and categories of appropriately trained personnel are available for the planned activities and tasks in a timely manner.

Annex A
(normative)
Tailoring process

The Tailoring Process is a process for performing basic tailoring of this International Standard for a software project. This annex provides requirements for tailoring this International Standard.

List of activities. This process consists of the following activities:

1) Identifying project environment;
2) Soliciting inputs;
3) Selecting processes, activities, and tasks;
4) Documenting tailoring decisions and rationale.

A.1 Identifying project environment. This activity consists of the following task:

A.1.1 Characteristics of the project environment that are going to influence tailoring shall be identified. Some of the characteristics may be: life cycle model; current system life cycle activity; system and software requirements; organizational policies, procedures and strategies; size, criticality and types of the system, software product or service; and number of personnel and parties involved.

A.2 Soliciting inputs. This activity consists of the following task:

A.2.1 Inputs from the organizations that are to be affected by the tailoring decisions shall be solicited. Users, support personnel, contracting officers, potential bidders should be involved in tailoring.

A.3 Selecting processes, activities, and tasks. This activity consists of the following tasks:

A.3.1 The processes, activities, and tasks that are to be performed shall be decided. These include the documentation to be developed and who are to be responsible for them. For this purpose, this International Standard should be evaluated against relevant data gathered in clauses A.1 and A.2.

A.3.2 The processes, activities, and tasks that were decided upon in A.3.1 but not provided in this International Standard shall be specified in the contract itself. Organizational life cycle processes (clause 7) should be evaluated to determine whether they could provide for these processes, activities, and tasks.

A.3.3 In this International Standard, requirements are indicated by tasks that contain "shall" or "will". These tasks should be carefully considered for whether they should be kept or deleted for a given project or a given business sector. Factors to be considered include but are not limited to: risk, cost, schedule, performance, size, criticality, and human interface.

A.4 Documenting tailoring decisions and rationale. This activity consists of the following task:

A.4.1 All tailoring decisions shall be documented together with the rationale for the decisions.

Annex B
(informative)
Guidance on tailoring

No two projects are the same. Variations in organizational policies and procedures, acquisition methods and strategies, project size and complexity, system requirements and development methods, among other things, influence how a system is acquired, developed, operated, or maintained. This International Standard is written for a general project to accommodate such variations as much as possible. Therefore, in the interest of cost reduction and quality improvement, this International Standard should be tailored for an individual project. All parties involved in the project should be involved in tailoring.

B.1 General tailoring guidance. This clause provides guidance on tailoring this International Standard and is not exhaustive. This clause may be used to perform first-level tailoring of this International Standard for a given business area; for example, aviation, nuclear, medical, military, country, or organization. The second-level tailoring should be performed for each specific project or contract.

B.2 Tailoring of the Development Process

The Development Process (5.3) needs special attention, because this process may be used by different parties with different objectives. As a first-level tailoring of this process, the following is recommended:

a) For software product that is embedded in or integral to the system: all the activities in the process should be considered; and it should be clarified whether the developer is required to *perform or support* the system activities.

b) For stand-alone software product, the system activities (5.3.2, 5.3.3, 5.3.10, and 5.3.11) may not be required but should be considered.

B.3 Tailoring of the evaluation-related activities

Persons who are involved in any activity of the life cycle of a project or a process, conduct evaluations either on their own or other's software products and activities. This International Standard groups these evaluations into five categories, which are listed below. The first four evaluation categories are at project level; the last one is at organizational level. These evaluations should be selected and tailored proportional to the scope, magnitude, complexity, and criticality of the project or of the organization. The problem, non-conformance, and improvement reports from these evaluations feed into the Problem Resolution Process (6.8).

a) Process-internal evaluations (evaluation tasks in 5.1 to 5.5). These are conducted by personnel performing the assigned tasks within the process during their day-to-day activities.

b) Verification (6.4) and Validation (6.5). Conducted by the acquirer, the supplier, or an independent party, to verify and validate the products in varying depth depending on the project. These evaluations do not duplicate or replace other evaluations, but supplement them.

c) Joint Reviews (6.6) and Audits (6.7). These are conducted in a joint forum by the reviewing and reviewed parties to evaluate status and compliance of products and activities on a pre-agreed to schedule.

d) Quality Assurance (6.3). Conducted by personnel independent of the personnel directly responsible for developing the software product or executing the process. The goal is to *independently assure* conformance of the software products and processes with the contract requirements and adherence to the established plans. This process may use the results from a, b, and c above as inputs. This process may coordinate its activities with those of a, b, and c.

e) Improvement (7.3). Conducted by an organization for efficient management and self-improvement of its process. This is conducted regardless of project or contract requirements.

B.4 Tailoring and application considerations

The paragraphs in this clause outline broad tailoring and application considerations for key project characteristics. Neither the considerations nor the characteristics are exhaustive and represent only current thinking. Figure B.1 provides an example of the application of this International Standard.

Organizational policies. Determine which organizational policies are relevant and applicable, such as on computer languages, safety and security, hardware reserve requirements, and risk management. The clauses of this International Standard related to the organizational policies should be kept.

Acquisition strategy. Determine which acquisition strategies are relevant and applicable for the project, such as types of contract, more than one contractor, involvement of subcontractors and verification and validation agents, level of acquirer's involvement with contractors, and evaluation of contractors' capabilities. The clauses of this International Standard related to these strategies should be kept.

Support concept. Determine which support concepts are relevant and applicable, such as expected length of support, degree of change, and whether the acquirer or the supplier will support. If the software product will have a long support life or is expected to change significantly, all documentation requirements should be considered. It is advisable to have the documentation automated.

Life Cycle model(s). Determine which life cycle model(s) are relevant and applicable for the project, such as Waterfall, evolutionary, builds, pre-planned product improvement, Spiral. All such models prescribe certain processes and activities that may be performed sequentially, repeated, and combined; in these models, the life cycle activities in this International Standard should be mapped to the selected model(s). For evolutionary, build, and pre-planned product improvement models, the outputs of one project activity feed into the next. In these cases, the documentation should be complete at the end of an activity or a task.

Parties involved. Determine or identify which parties are involved in the project, such as acquirer, supplier, developer, subcontractor, verification agent, and validation agent, maintainer; and the number of personnel. All the requirements related to organizational interfaces between two parties are under consideration; for example, acquirer to developer and supplier to verification or validation agent. A large project involving many (tens or hundreds) persons needs significant management oversight and control. Tools such as internal and independent evaluations, reviews, audits and inspections, and data collection are important for a large project. For small projects, these controls may be excessive.

System life cycle activity. Determine which current system life cycle activities are relevant and applicable, such as acquirer's project initiation, supplier's development, and maintenance. Some scenarios:

Acquirer is initiating or defining system requirements. Feasibility studies and prototyping of requirements and design may be conducted. Software code for prototypes may be developed, which may or may not be used later in the development of software products performed under contract. System requirements and preliminary software requirements may be developed. In these cases, the Development Process (5.3) may be used as a guidance rather than requirement; the rigor of qualification and evaluation may not be needed; and joint reviews and audits may not be needed.

Developer is producing software products under contract. In this case all Development Process (5.3) requirements should be considered during tailoring.

Maintainer is modifying software products. The Maintenance Process (5.5) is under consideration. Parts of the Development Process (5.3) may be used as mini-processes.

System-Level characteristics. Determine which system-level characteristics are relevant and applicable, such as number of subsystems and configuration items. If the system has many subsystems or configuration items, the Development Process (5.3) should be carefully tailored for each subsystem and configuration item. All interface and integration requirements should be considered.

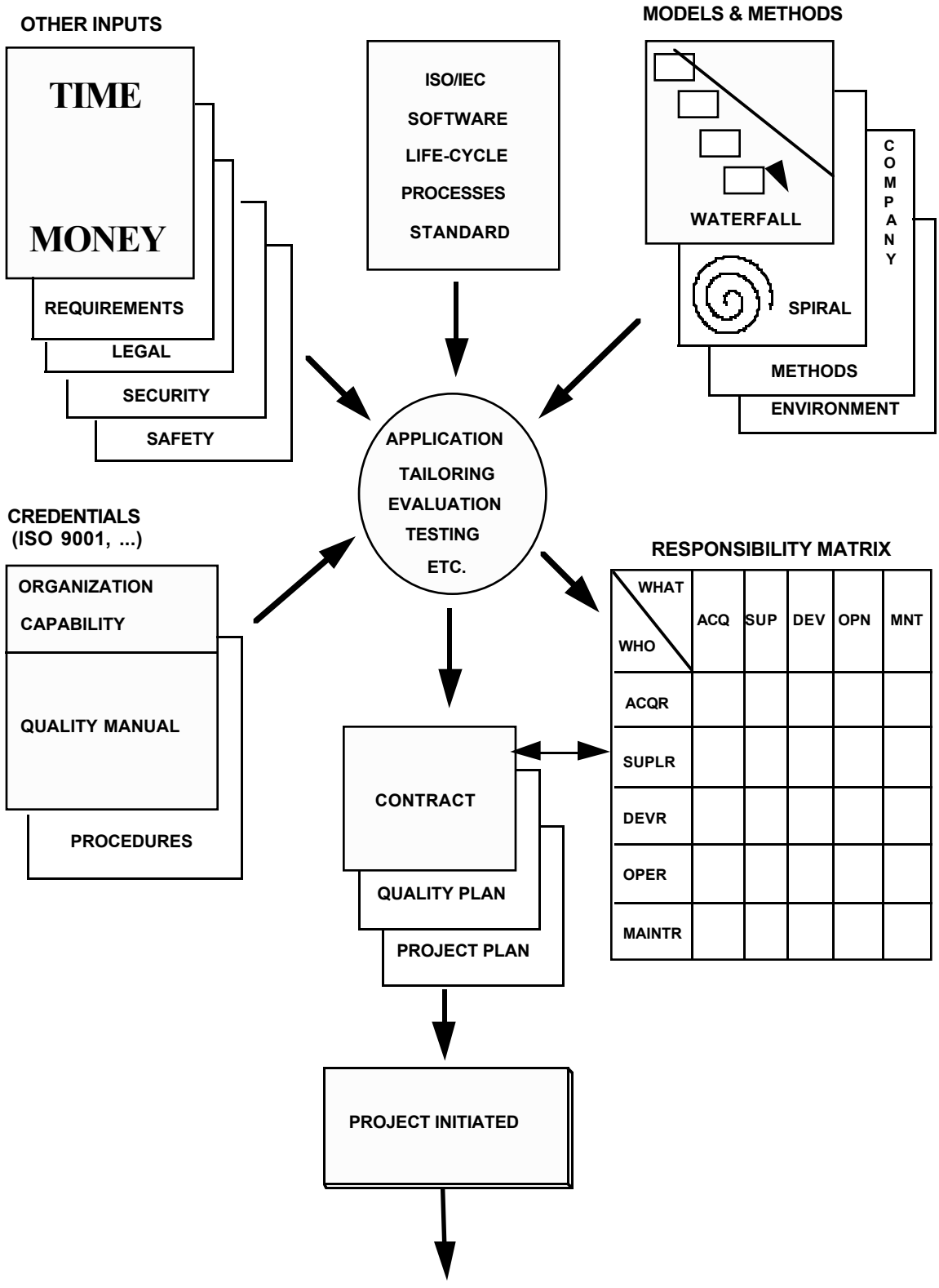


Figure B.1 An Example of the Application of the International Standard

Software-Level characteristics. Determine which software-level characteristics are relevant and applicable, such as number of software items, types, size and criticality of software products, and technical risks. If the software product has many software items, components, and units, the Development Process (5.3) should be carefully tailored for each software item. All interface and integration requirements should be considered.

Determine which types of software product are involved, as different types of software product may require different tailoring decisions. Some examples:

a) New development. All of the requirements, particularly the Development Process (5.3), should be under consideration.

b) Use of off-the-shelf software product "as is." The full Development Process (5.3) may be excessive. Performance, documentation, proprietary, usage, ownership, warranty and licensing rights, and future support related to the software product should be evaluated.

c) Modification of off-the-shelf software product. Documentation may not be available. Depending on the criticality and expected future changes, the Development Process (5.3) should be used via the Maintenance Process (clause 5.5). Performance, documentation, proprietary, ownership, usage, warranty and licensing rights, and future support related to the software product should be evaluated.

d) Software or firmware product embedded in or integral to a system. Since such a software product is a part of a larger system, the system-related activities in the Development Process (5.3) should be considered. In the system-related activities, only one verb "perform" or "support" needs to be selected. If the software or firmware product is not likely to be modified in the future, extent of documentation needs should be carefully examined.

e) Software product that is stand-alone. Since such a software product is not a part of a system, the system-related activities in the Development Process (5.3) need not be considered. Documentation needs, particularly for maintenance, should be carefully examined.

f) Non-deliverable software product. As no items are being acquired, supplied, or developed, no provision of this International Standard except 5.3.1.5 of the Development Process (5.3) should be considered. However, if the acquirer decides to acquire a piece of such a software product for future operation and maintenance, then this software product should be treated as in b or c above.

Other considerations.

The more dependent the system is upon the software product operating correctly and being finished on time, the more management control should be imposed via testing, reviews, audits, verification, validation, and so on. Conversely, much management control of non-critical or small software product may not be cost-effective.

Development of software product may have technical risks. If the software technology used is not mature, software product being developed is unprecedented or complex, or software product contains safety, security, or other critical requirements, then rigorous specification, design, testing, and evaluations may be needed. Independent verification and validation may be important.

Annex C
(informative)
Guidance on processes and organizations

This annex, to promote understandability, presents a discussion on the processes, organizations, and their relationships under key viewpoints.

C.1 Processes under key points of view

This International Standard contains the processes that are applicable throughout the life cycle of software. However, these processes may be used in different ways by different organizations and parties with different views and objectives. This clause presents the processes and their relationships under key points of view. See 4.1.1 for synopses of the processes.

Figure C.1 depicts the software life cycle processes and their relationships under different views of the usage of this International Standard. The basic views shown are: contract, management, operating, engineering, and supporting. Under the contract view, acquirer and supplier parties negotiate and enter into a contract and employ the Acquisition Process and Supply Process respectively. Under the management view, the acquirer, supplier, developer, operator, maintainer, or other party manages its respective process. Under the operating view, the operator provides software operation service for the users. Under the engineering view, the developer or maintainer conducts its respective engineering tasks to produce or modify software products. Under the supporting view, parties (such as configuration management, quality assurance) provide supporting services to others in fulfilling specific, unique tasks. Also shown (see the bottom box) are the organizational processes; these are employed by an organization at the corporate level to establish and implement an underlying structure made up of associated life cycle process(es) and personnel and continuously improve them.

Figure C.2 presents the primary (top, left box), supporting (top, right box), and organizational (bottom box) life cycle processes and their constituent activity names under different views. A numeral prefixed to a process refers to the section number in this International Standard.

The contract view has two life cycle processes (see the upper shaded box under the Primary Life Cycle Processes): an Acquisition Process for the acquirer and a Supply Process for the supplier. Each process shows its constituent activities. These processes define the tasks for the acquirer and the supplier respectively from the contractual viewpoint.

The engineering view has two life cycle processes (see the lower, left-bottom shaded box in the Primary Life Cycle Process): a Development Process and a Maintenance Process. Each process shows its constituent activities. The Development Process is employed by development engineers for producing software products. The Maintenance Process is employed by maintenance engineers for modifying the software and keeping it current.

The operating view has one life cycle process (see the lower, right shaded box in the Primary Life Cycle Process): an Operation Process and its constituent activities. The Operation Process is employed for operating the software for its users.

The quality management view has six life cycle processes (see the shaded box in Supporting Life Cycle Processes): Quality Assurance Process; Verification Process; Validation Process; Joint Review Process; and Audit Process. Their constituent activities are not shown. These quality related processes are employed for managing quality throughout the software life cycle. The Verification; Validation; Joint Review; and Audit processes may be employed by different parties separately and as techniques of the Quality Assurance Process as well.

The management view has one process (see the shaded box in Organizational Life Cycle Processes): a Management Process that is used by any organization for managing its respective process. Its constituent activities are shown.

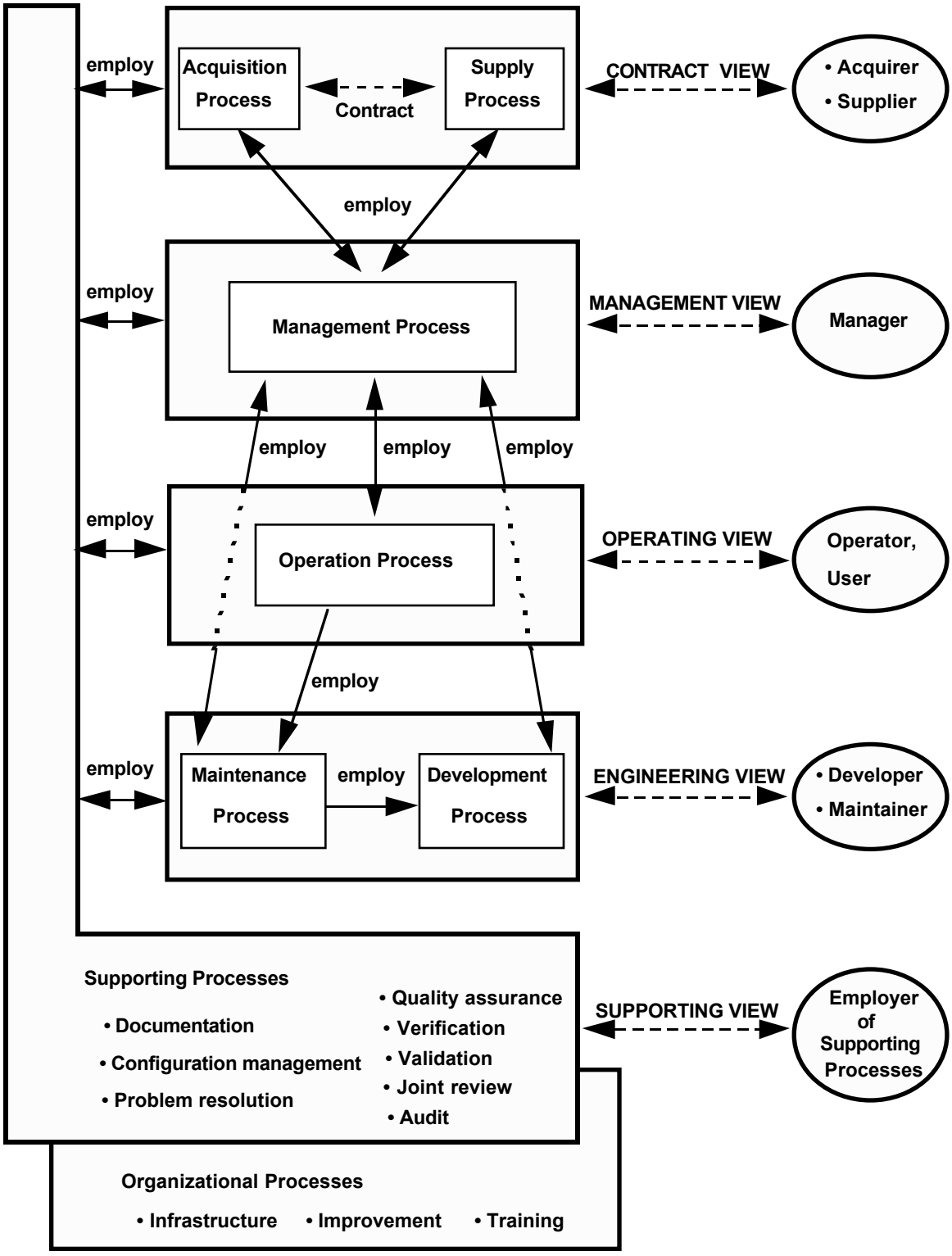
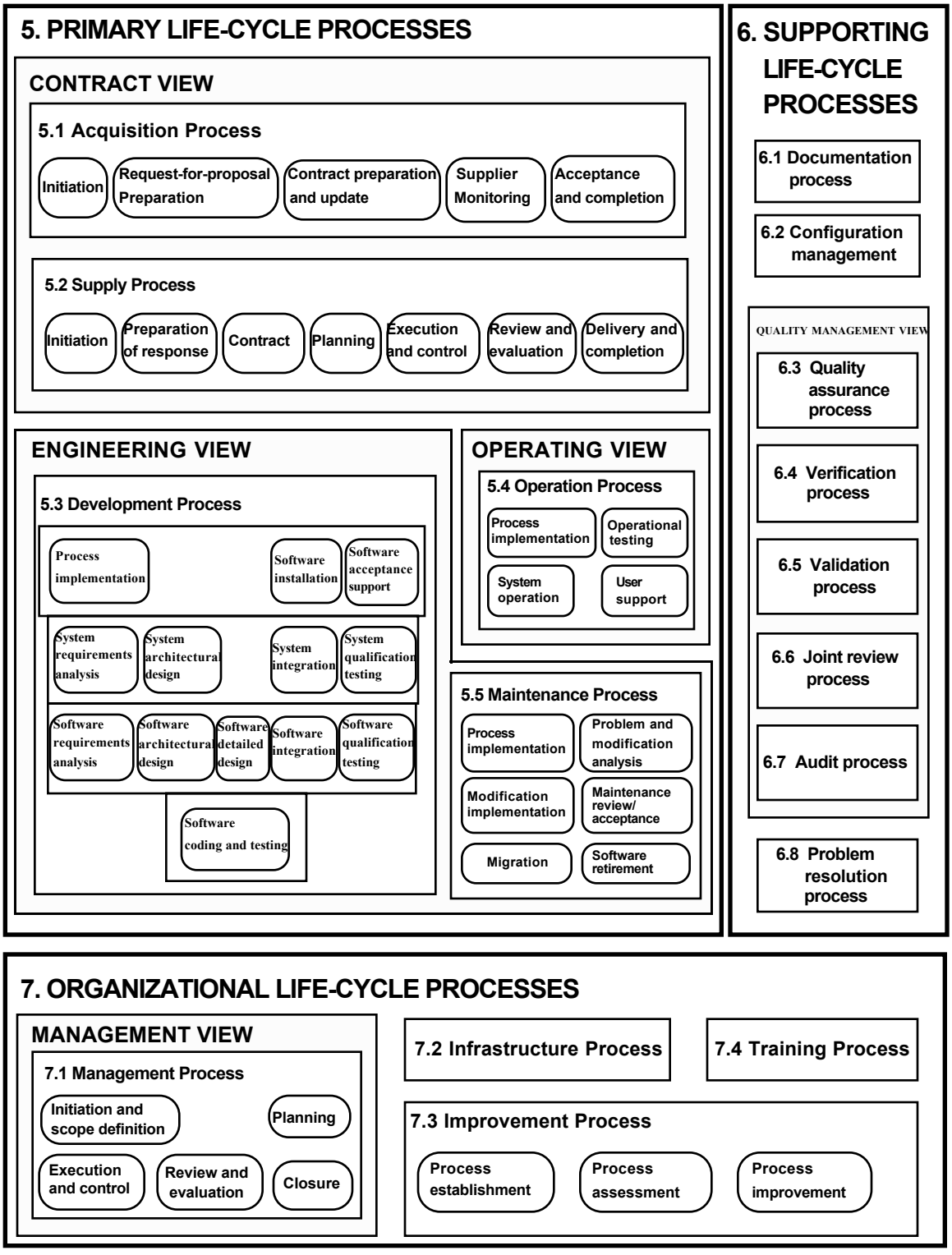


Figure C.1 Software Life-Cycle Processes -- Roles and Relationships



The position order of activities does not mean time order.

Names of activities in the Development Process are not names of development phases.

Figure C.2 Software Life-Cycle Processes, Views and Activities

C.2 Processes, organizations, and relationships

The processes and organizations (or parties) are only related functionally. They do not dictate a structure for an organization (or a party).

In this International Standard, the terms "organization" and "party" are *nearly* synonymous. An organization is a body of persons organized for some specific purpose, as a club, union, corporation, or society. When an organization, as a whole or a part, enters into a contract, it is a party. Organizations are separate bodies, but the parties may be from the same organization or from separate organizations.

An organization or a party gets its name from the process it performs; for example, it is called an acquirer when it performs the Acquisition Process.

An organization may perform one process or more than one process; a process may be performed by one organization or more than one organization. *Under one contract or application of this International Standard,* a given party should not perform both the Acquisition Process and the Supply Process, but it can perform other processes.

In this International Standard itself, the relationships between the processes are only static. The more important dynamic, real-life relationships between the processes, between the parties, and between the processes and the parties are automatically established when this International Standard is applied on software projects. Each process (and the party performing it) contributes to the software project in its own unique way. The Acquisition Process (and the acquirer) contributes by defining the system, which would contain software product. The Supply Process (and the supplier) contributes by providing the software product or service on which that system would depend. The Development Process (and the developer) contributes by "looking" to the system for correct derivation and definition of software product, by supporting proper integration of the software product back into the system, and by developing the software product in between. The Operation Process (and the operator) contributes by operating the software product in the system's environment for the benefit of the users, the business, and the mission. The Maintenance Process (and the maintainer) contributes by maintaining and sustaining the software product for operational fitness and by providing support and advice to the user community. Each supporting or organizational process contributes by providing unique, specialized functions to other processes as needed.

Annex D
(informative)
Bibliography

ISO/IEC 12119: 1994, *Information technology—Software packages—Quality requirements and testing*.

Annex E
(informative)
Basic concepts of ISO/IEC 12207

This annex explains the concepts upon which ISO/IEC 12207 was originally developed. The information in this annex is intended as an aid in understanding this standard.

E.1 Software life cycle architecture

This standard establishes a framework for the life cycle of software. The life cycle begins with an idea or a need that can be satisfied wholly or partly by software and ends with the retirement of the software. The architecture is built with a set of processes and interrelationships among these processes. The determination of the life cycle processes is based upon two basic principles: modularity and responsibility.

E.1.1 Modularity

The life cycle processes are cohesive and coupled to the optimum extent deemed practical and feasible.

E.1.2 Responsibility

A process is placed under the responsibility of an organization or a party in the software life cycle.

E.2 Life cycle processes

The life cycle processes are grouped into three broad classes: primary, supporting, and organizational. Primary processes are acquisition, supply, development, operation, and maintenance. Supporting processes are documentation, configuration management, quality assurance, joint review, audit, verification, validation, and problem resolution. Organizational processes are management, infrastructure, improvement, and training. An organization may employ an organizational process to establish, control, and improve a life cycle process. The life cycle processes are shown in figure E.1. The Tailoring process, even though not a life cycle process, is shown for completeness; it may be applied to any (or all) of the processes in this standard.

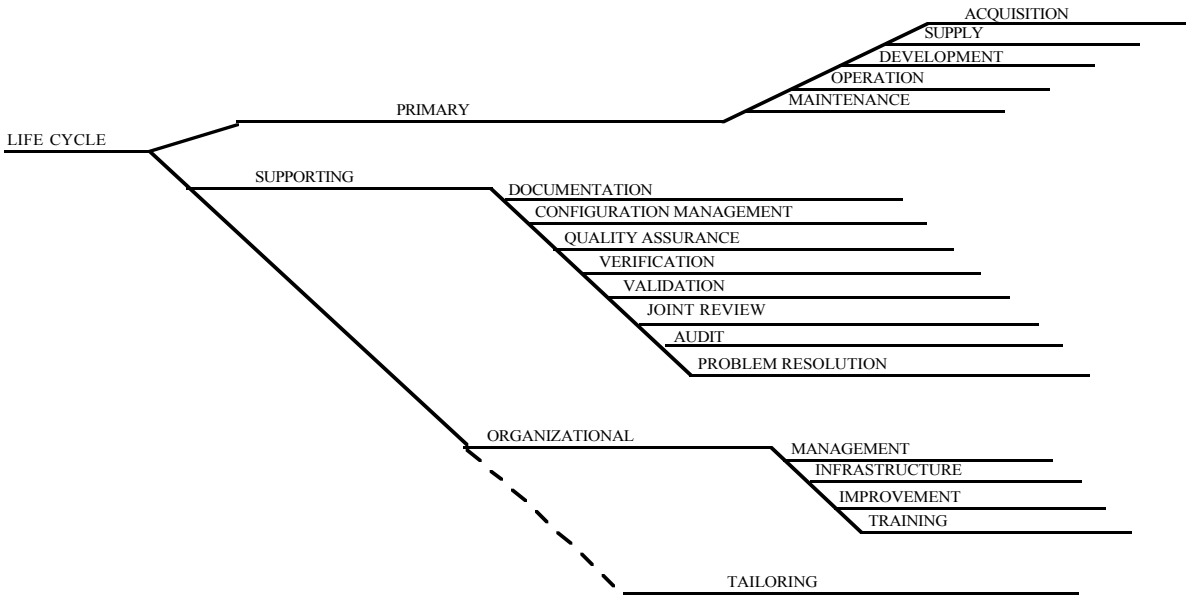


Figure E.1 The life cycle processes tree

E.3 Structure of a life cycle process

Each process is further described in terms of its own constituent activities, each of which is further described in terms of its constituent tasks. An activity under a process is a set of cohesive tasks.

E.4 Nature of a task

A task is expressed in the form of self-declaration, requirement, recommendation, or permissible action. For this purpose, this standard carefully employs certain auxiliary verbs (will, shall, should, and may) to differentiate between the distinct forms of a task. "Will" is used to express a self-declaration of purpose or intent by one party, "shall" to express a binding provision between two or more parties, "should" to express a recommendation among other possibilities, and "may" to indicate a course of action permissible within the limits of this standard.

E.5 Nature of evaluation

In this standard, evaluation is an elementary task that is used in many activities. Evaluations are conducted on entities with given purposes against given criteria. Examples of an entity include a process, an activity, a task, a plan, an agreement, a report, data, information, or a product. Examples of a purpose include review, audit, verify, validate, assure, or improve. Examples of a criterion include: traceability of design to requirements, or correctness of design.

E.6 Total quality management

This standard is consistent with total quality management principles.

This standard treats all activities related to quality as an integral part of the software life cycle. Thus, quality is considered from the outset.

The quality-related activities in the life cycle are appropriated to each process. Each primary process is equipped with a built-in "plan-do-check-act" (PDCA) cycle. Thus, each process and the personnel responsible for performing the process are assigned their germane process-internal quality-related activities, including evaluations.

The Quality Assurance process is dedicated to assuring conformity of products and services with their specified requirements. Those responsible for this process are provided with the organizational freedom and authority to effect the conformity. Organizational freedom requires independence from the one who has the direct management responsibility for producing the product or providing the service; while authority means having the authority to initiate evaluations and related corrective actions.

This standard provides an improvement process for managing, controlling, and improving the established processes and their performance.

E.7 Link between system and software

This standard establishes a strong link between a system and its software. It is based upon the general principles of system engineering. The basic components of system engineering (e.g., analysis, design, fabrication, evaluation, testing, integration, manufacturing, and storage/distribution) form the foundation for software engineering in the standard.

This standard provides the minimum system context for software. Software is treated as an integral part of the total system and performs certain functions in that system. This is implemented by extracting the software requirements from the system requirements and design, producing the software, and integrating it into the system.

E.8 Organization and party

In this standard, the terms "organization" and "party" are nearly synonymous. An organization is a body of persons organized for some specific purpose, such as a club, union, corporation, or society. When an organization, as a whole or a part, enters into an agreement, it is a party. Organizations are separate bodies, but the parties may be from the same organization or from separate organizations.

An organization or a party derives its name from the process for which it is responsible. For example, it is called an acquirer when it performs the Acquisition process.

E.9 Applicability to organizations

The processes in this standard form a comprehensive set to serve various organizations. An organization, small or large, depending on its business purpose or its acquisition strategy, can select an appropriate set of the processes (and associated activities and tasks) to fulfill that purpose. An organization may perform one process or more than one process. Under one contract or application of this standard, a given party should not perform both the Acquisition process and the Supply process, but, it can perform other processes. A process may be performed by one organization or more than one organization. An example of a process performed by more than one organization is the Joint Review process.

This standard is intended to be applied by an organization internally or contractually by two or more organizations. In order to facilitate application of this standard either internally or contractually, the tasks are expressed in contractual language. When applied internally, the contractual language is to be interpreted as a self-imposed task.

E.10 Applicability to projects

This standard is written for a general, large, and complex software project. This standard is designed to be tailored or adapted for a software project of lesser size or complexity. It is also designed to be used whether the software is a stand-alone entity, or an embedded or integral part of a parent system.

On the same project, this standard may be applied more than once. For example, in a given software development project, an acquirer tasks a supplier to perform software development; and the supplier tasks its subcontractor to perform all or parts of the software development. In the former, the acquirer and the supplier execute one application of the standard. In the latter, the supplier (as an acquirer) and its subcontractor (as a supplier) execute a separate application of this standard.

E.11 Responsiveness to evolving technologies

This standard is intended to be responsive to the rapidly evolving software engineering discipline. From a top-level viewpoint, the activities and tasks of a software life cycle process are "what-to-do" items, not "how-to-do" items. In other words, a task might be "develop an architectural design," but not "develop architectural design by using the top-down, functional-design method." This scheme provides an acquirer an avenue to specify an end product or service and, at the same time, allows the supplier to be creative and to employ appropriate methods, techniques, and tools to produce the product or provide the service.

This standard is flexible and usable with any life cycle model (e.g., Waterfall, Incremental, Evolutionary, Spiral); any software engineering method (e.g., object-oriented design, structured coding, top-down testing); or any programming language (e.g., Fourth Generation, Ada, Assembly). The models, methods, and languages are dependent upon the software project and state-of-the-art technology, and their selection is left to the user of this standard.

This standard is adaptable by a business sector (e.g., military, commercial, automotive, airline) or any national or organizational culture.

E.12 Non-prescription of events and milestones

In this standard, the processes and their activities and tasks are arranged in a sequence suitable for exposition. This positional sequence does not prescribe or dictate any time-dependent sequence. For lack of consensus on or use of a universal time-dependent sequence, the user of this standard may select and order the processes, activities, and tasks as appropriate and effective. This standard encourages iteration among the activities and recursion within an activity to offset the effects of any implied sequence of activities and tasks. The parties of this standard are responsible for selecting a life cycle model for the project and mapping the processes, activities, and tasks onto that model.

E.13 Documentation of outputs

This standard requires the documentation of certain outputs, but it does not specify format, content, or media of the documents, that is left for the agreement between the acquirer and the supplier. An organization may use its existing documentation sets or standards with this standard by establishing the correlation between the calls for documentation in this standard and the organization's documentation standards. This standard provides a Documentation process which allows the party or parties to plan, design, develop, produce, edit, distribute, and maintain needed documents. IEEE/EIA P12207.1 provides guidance on documentation of life cycle data.

E.14 Baselining

This standard differentiates between "configuration items" and "items of hardware and software." The intent of this standard is to appropriately control requirements, design, and code associated with both "items of software" and "software configuration items." The application of the Configuration Management process applies to both elements, but, for "software configuration items": (1) the time at which they are established may be much later in the project; (2) the formality of identification, control, status accounting, and evaluation may be more rigid; and (3) the approval process to modify may be more restrictive. According to this standard, software and hardware configuration items shall be established prior to system integration.

This standard also defines "baseline" as a formally approved version of a configuration item, regardless of media, formally designated and fixed at a specific time during the configuration item's life cycle. The intent is to establish and baseline configuration items at the appropriate time as part of the Development process (and the Maintenance process), not the Configuration Management process.

E.15 Software metrics

This standard is not a software metrics standard. This standard requires specification of management indicators (such as cost expenditure) and software attributes (e.g., reliability, maintainability), but it does not define or prescribe them. This standard references ISO/IEC 9126 for guidance on software quality characteristics. Detailed specifications of management indicators and software attributes are left to the users of this standard.

E.16 Certification to this standard

This standard does not address certification of an organization to the life cycle processes, nor does it define any certification criteria.

E.17 Processes and their interactions

Figure E.2 depicts the processes and their interactions.

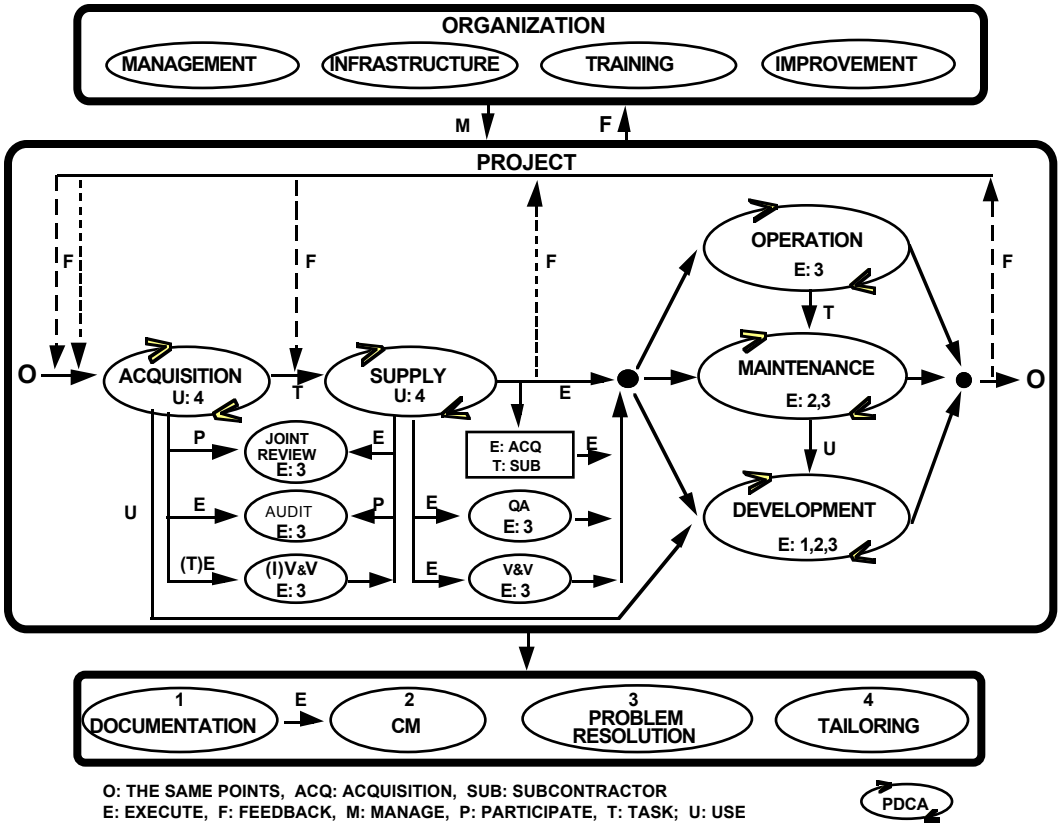


Figure E.2 The processes and their interactions

E.18 Limitations

This standard is not a substitute for systematic, disciplined management and engineering of software systems. This standard merely provides a framework where the processes, activities, and tasks related to software can be reasonably identified, planned, and acted upon.

This standard contains only a set of well-defined building blocks (processes); the user of this standard should select, tailor, and assemble those processes and their activities and tasks that are cost-effective for the organization and the project.

E.19 Prerequisites to using this standard

This standard covers the full life cycle of a software system and caters to diverse, independent parties in the life cycle. It integrates organizational and contractual tasks and responsibilities as related to the processes. This standard recognizes that the objectives of different life cycle phases and affected parties differ and sometimes conflict. Therefore, the intent of this standard is to require all affected parties to (1) perform identified activities and tasks, and (2) clearly articulate expectations from other parties (e.g., acquirer's expectations of supplier). When conflicts arise, the standard intends for the parties to resolve them. This standard is intended to be used within the context of an organization's objectives and each project's needs. For effective and productive use of this standard, the following prerequisites (in the given order) should be met:

a) Trained personnel;
b) Familiarity with the organization's policies;
c) Familiarity with the project's environment;
d) An understanding of the use of the standard.

Annex F
(normative)
Compliance

In accordance with clause 1.4, paragraph 2, of the International Standard, the U.S. defines compliance with this U.S. standard as explained in this annex.

F.1 Definition of compliance

Compliance with this U.S. standard is defined similarly to clause 1.4, paragraph 1, of the International Standard:

> Compliance with this Standard is defined as the performance of all the processes, activities, and tasks selected from this Standard in the Tailoring Process (annex A) for the software project. The performance of a process or an activity is complete when all its required tasks are performed in accordance with the pre-established criteria and the requirements specified in the contract as applicable.

The overall interpretation of compliance may vary in different situations, as explained in clause F.2. The set of tasks "selected ... in the Tailoring Process" may be constrained, as explained in clause F.3. The performance of the processes, activities, and tasks "in accordance with the pre-established criteria" may be achieved in a number of ways, as explained in clause F.4.

F.2 Compliance situations

Compliance may be interpreted differently for various situations. The relevant situation shall be identified in the claim of compliance:

a) When compliance is claimed for an organization, the organization shall make public a document declaring its tailoring of the processes, activities, and tasks and its interpretation of any clauses of the standard that reference "the contract."

   NOTE—One possible way for an organization to deal with clauses that cite "the contract" is to specify that they shall be interpreted in the project plans for any particular project.

b) When compliance is claimed for a project, the project plans or the contract shall document the tailoring of the processes, activities and tasks and the interpretation of any clauses of the standard that reference "the contract."

   NOTE—A project's claim of compliance is typically specified with respect to the organization's claim of compliance.

c) When compliance is claimed for a multi-supplier program, it may be the case that no individual project can claim compliance because no single contract calls for the performance of all required processes and activities. Nevertheless, the program, as a whole, may claim compliance if each of the required processes and activities is performed by an identified party. The program plans shall document the tailoring of the processes, activities and tasks, and their assignment to the various parties, as well as the interpretation of any clauses of the standard that reference "the contract."

d) When the standard is used as a basis for regulatory decisions, supplemental standards, legal regulations or sector-specific requirements may constrain the tailoring and the interpretation of the clauses of the standard that reference "the contract." In this case, the project plans shall specify the tailoring and the interpretation of the clauses of the standard that reference "the contract."

F.3 Level of compliance

One of the following levels of compliance shall be asserted. The selected level shall be identified in the claim of compliance:

a) Tailored: The minimum set of required processes, activities, and tasks is determined by tailoring in accordance with annex A.

b) Absolute: The minimum set of required processes, activities, and tasks are all of those specified as mandatory (i.e., clauses containing "shall" or "will") in the text of the standard.

NOTE—Absolute compliance may be claimed for selected processes even if absolute compliance with the entire standard is not claimed.

F.4 Compliance criteria

Performance of any process of the standard shall be completed using either of two sets of criteria. For each process, the claim of compliance shall cite which of the sets has been chosen:

a) Accomplishment of all of the requirements of the activities and tasks of the process as specified in clauses 5, 6, and 7 of IEEE/EIA 12207, in accordance with the selected level of compliance and the chosen tailoring.

b) Accomplishment of the process objectives via an "alternative method": To be acceptable, alternative methods must accomplish the process objectives specified in annex G of IEEE/EIA 12207, must accomplish the life cycle data objectives specified in annex H of IEEE/EIA 12207, must not hamper compliance of any other process, and must be specified by plans, standards or other documents. The alternative method and the means of compliance shall be specified or cited in the claim of compliance.

NOTE—Possible "alternative methods" for some processes are specified in IEEE/EIA P12207.2.

Annex G
(normative)
Life cycle processes objectives

The purpose of this annex is to describe the basic objectives to be considered in meeting the intent of each life cycle process defined in ISO/IEC 12207. For this annex, “customer” is defined to be the recipient of a product provided by a suppler. In the contractual situation, the customer is called the “acquirer.” The “customer” may be the ultimate consumer, user, beneficiary, or purchaser. The “customer” can be external or internal to the organization. Data engineering activities are assumed to be covered under the heading of the analogous software engineering activities (e.g., requirements and design).

The objectives presented below are intended to be complete. For a particular project, only the subset of those objectives that conform to the tailoring of the standard would be applicable.

G.1 Acquisition process

The use of the Acquisition process should achieve the following objectives:

a) Develop a contract, including tailoring of the standard, that clearly expresses the expectation, responsibilities, and liabilities of both the acquirer and the supplier;

b) Obtain products and/or services that satisfy the customer need;

c) Manage the acquisition so that specified constraints (e.g., cost, schedule and quality) and goals (e.g., degree of software reuse) are met;

d) Establish a statement of work to be performed under contract;

e) Qualify potential suppliers through an assessment of their capability to perform the required software;

f) Select qualified suppliers to perform defined portions of the contract;

g) Establish and manage commitments to and from the supplier;

h) Regularly exchange progress information with the supplier;

i) Assess compliance of the supplier against the agreed upon plans, standards and procedures;

j) Assess the quality of the supplier’s delivered products and services;

k) Establish and execute acceptance strategy and conditions (criteria) for the software product or service being acquired;

l) Establish a means by which the acquirer will assume responsibility for the acquired software product or service.

G.2 Audit process

The use of the Audit process should achieve the following objectives:

a) Determine compliance with requirements, plans, and contract, as appropriate;

b) Arrange the conduct of audits of work products or process performance by a qualified independent party, as specified in the plans;

c) Conduct follow-up audits to assess corrective action(s), closure, and root cause actions.

G.3 Configuration Management process

The use of the Configuration Management process should achieve the following objectives:

a) Identify, define, and control all relevant items of the project;

b) Control modifications of the items;

c) Record and report the status of items and modification requests;

d) Ensure the completeness of the items;

e) Control storage, handling, release, and delivery of the items.

G.4 Development process

The use of the Development process should achieve the following objectives:

a) Develop requirements of the system that match the customer's stated and implied needs;

b) Propose an effective solution that identifies the main elements of the system;

c) Allocate the defined requirements to each of those main elements;

d) Develop a system release strategy;

e) Communicate the requirements, proposed solution and their relationships to all affected parties;

f) Define the requirements allocated to software components of the system and their interfaces to match the customer's stated and implied needs;

g) Develop software requirements that are analyzed, correct, and testable;

h) Understand the impact of software requirements on the operating environment;

i) Develop a software release strategy;

j) Approve and update the software requirements, as needed;

k) Communicate the software requirements to all affected parties;

l) Develop an architectural design;

m) Define internal and external interfaces of each software component;

n) Establish traceability between system requirements and design and software requirements, between software requirements and software design, and between software requirements and tests;

o) Define verification criteria for all software units against the software requirements;

p) Produce software units defined by the design;

q) Accomplish verification of the software units against the design;

r) Develop an integration strategy for software units consistent with the release strategy;

s) Develop acceptance criteria for software unit aggregates that verify compliance with the software requirements allocated to the units;

t) Verify software aggregates using the defined acceptance criteria;

u) Verify integrated software using the defined acceptance criteria;

v) Record the results of the software tests;

w) Develop a regression strategy for retesting aggregates, or the integrated software, should a change in components be made;

x) Develop an integration plan to build system unit aggregates according to the release strategy;

y) Define acceptance criteria for each aggregate to verify compliance with the system requirements allocated to the units;

z) Verify system aggregates using the defined acceptance criteria;

aa) Construct an integrated system demonstrating compliance with the system requirements (functional, nonfunctional, operations and maintenance);

ab) Record the results of the system tests;

ac) Develop a regression strategy for retesting aggregates or the integrated system should a change in components be made;

ad) Identify transition concerns, such as availability or work products, availability of system resources to resolve problems and adequately test before fielding corrections, maintainability, and assessment of transitioned work products.

G.5 Documentation process

The use of the Documentation process should achieve the following objectives:

a) Identify all documents to be produced by the process or project;

b) Specify the content and purpose of all documents and plan and schedule their production;

c) Identify the standards to be applied for development of documents;

d) Develop and publish all documents in accordance with identified standards and in accordance with nominated plans;

e) Maintain all documents in accordance with specified criteria.

NOTE—Refer to clause H.5 of this standard for presentation form of software life cycle data.

G.6 Improvement process

The use of the Improvement process should achieve the following objectives:

a) Establish a well-defined and maintained standard set of processes, along with a description of the applicability of each process;

b) Identify the detailed tasks, activities, and associated work products for each standard process, together with expected criteria;

c) Establish a deployed specific process for each project, tailored from the standard process in accordance with the needs of the project;

d) Establish and maintain information and data related to the use of the standard process for specific projects;

e) Understand the relative strengths and weaknesses of the organization's standard software processes;

f) Make changes to standard and defined processes in a controlled way;

g) Implement planned and monitored software process improvement activities in a coordinated manner across the organization.

G.7 Infrastructure process

The use of the Infrastructure process should achieve the following objectives:

a) Establish and maintain a well-defined software engineering environment, consistent with and supportive of the set of standard processes and organizational methods and techniques;

b) Tailor the software engineering environment to the needs of the project and the project team;

c) Develop a software engineering environment that supports project team members regardless of the performance location of process activities;

d) Implement a defined and deployed strategy for reuse.

G.8 Joint Review process

The use of the Joint Review process should achieve the following objectives:

a) Evaluate the status and products of an activity of a process through joint review activities between the parties to a contract;

b) Establish mechanisms to ensure that action items raised are recorded for action.

G.9 Maintenance process

The use of the Maintenance process should achieve the following objectives:

a) Define the impact of organization, operations, and interfaces on the existing system in operation;

b) Identify and update appropriate life cycle data;

c) Develop modified system components with associated documentation and tests that demonstrate that the system requirements are not compromised;

d) Migrate system and software upgrades to the user's environment;

e) Ensure fielding of new systems or versions does not adversely affect ongoing operations;

f) Maintain the capability to resume processing with prior versions.

G.10 Management process

The use of the Management process should achieve the following objectives:

a) Define the scope of the work for the project;

b) Identify, size, estimate, plan, track, and measure the tasks and resources necessary to complete the work;

c) Identify and manage interfaces between elements in the project and with other projects and organizational units;

d) Take corrective action when project targets are not achieved;

e) Establish quality goals, based on the customer's quality requirements, for various checkpoints within the project's software life cycle;

f) Establish product performance (memory, processing, communications) goals, based on the customer's requirements, for various checkpoints within the project's software life cycle;

g) Define and use metrics that measure the results of project activities or tasks, at checkpoints within the project's life cycle, to assess whether the technical, quality, and product performance goals have been achieved;

h) Establish criteria, metrics, and procedures for identifying software engineering practices and integrate improved practices into the appropriate software life cycle processes and methods;

i) Perform the identified quality activities and confirm their performance;

j) Take corrective action when technical, quality, and product performance goals are not achieved;

k) Determine the scope of risk management to be performed for the project;

l) Identify risks to the project as they develop;

m) Analyze risks and determine the priority in which to apply resources to mitigate those risks;

n) Define, implement, and assess appropriate risk mitigation strategies;

o) Define, apply, and assess risk metrics to measure the change in the risk state and the progress of the mitigation activities;

p) Establish an environment that supports effective interaction between individuals and groups;

q) Take corrective action when expected progress is not achieved.

G.11 Operation process

The use of the Operation process should achieve the following objectives:

a) Identify and mitigate operational risks for the software introduction and operation;

b) Operate the software in its intended environment according to documented procedures;

c) Provide operational support by resolving operational problems and handling user inquires and requests;

d) Provide assurance that software (and host system) capacities are adequate to meet user needs.

e) Identify customer support service needs on an ongoing basis;

f) Assess customer satisfaction with both the support services being provided and the product itself on an ongoing basis;

g) Deliver needed customer services.

G.12 Problem Resolution process

The use of the Problem Resolution process should achieve the following objectives:

a) Provide a timely, responsive, and documented means to ensure that all discovered problems are analyzed and resolved;

b) Provide a mechanism for recognizing and acting on trends in problems identified.

G.13 Quality Assurance process

The use of the Quality Assurance process should achieve the following objectives:

a) Identify, plan, and schedule quality assurance activities for the process or product;

b) Identify quality standards, methodologies, procedures, and tools for performing quality assurance activities and tailor to the project;

c) Identify resources and responsibilities for the performance of quality assurance activities;

d) Establish and guarantee the independence of those responsible for performing quality assurance activities;

e) Perform the identified quality assurance activities in line with the relevant plans, procedures, and schedules;

f) Apply organizational quality management systems to the project.

G.14 Supply process

The use of the Supply process should achieve the following objectives:

a) Establish clear and ongoing communication with the customer;

b) Define documented and agreed customer requirements, with managed changes;

c) Establish a mechanism for ongoing monitoring of customer needs;

d) Establish a mechanism for ensuring that customers can easily determine the status and disposition of their requests;

e) Determine requirements for replication, distribution, installation, and testing of the system containing software or stand-alone software product;

f) Package the system containing software or the stand-alone software product in a way that facilitates its efficient and effective replication, distribution, installation, testing, and operation;

g) Deliver a quality system containing software or stand-alone software product to the customer, as defined by the requirements, and install in accordance with the identified requirements.

G.15 Training process

The use of the Training process should achieve the following objectives:

a) Identify the roles and skills required for the operations of the organization and the project;

b) Establish formal procedures by which talent is recruited, selected, and transitioned into assignments in the organization;

c) Design and conduct training to ensure that all individuals have the skills required to perform their assignments;

d) Identify and recruit or train, as appropriate, individuals with the required skills and competencies to perform the organizational and project roles;

e) Establish a work force with the skills to share information and coordinate their activities efficiently;

f) Define objective criteria against which unit and individual training performance can be measured, to provide performance feedback, and to enhance performance continuously;

G.16 Validation process

The use of the Validation process should achieve the following objectives:

a) Identify criteria for validation of all required work products;

b) Perform required validation activities;

c) Provide evidence that the work products, as developed, are suitable for their intended use.

G.17 Verification process

The use of the Verification process should achieve the following objectives:

a) Identify criteria for verification of all required work products;

b) Perform requirements verification activities;

c) Find and remove defects from products produced by the project.

Annex H
(Normative)
Life cycle data objectives

The purpose of this annex is to describe the basic principles to be considered in preparing data during the execution of the software life cycle processes of IEEE/EIA 12207.

H.1 Purpose of software life cycle data

The life cycle data should support the following actions:

a) Describe and record information about a software product during its life cycle;

b) Assist usability and maintainability of a software product;

c) Define and control life cycle processes;

d) Communicate information about the system, software product or service, and project to those who need it;

e) Provide a history of what happened during development and maintenance to support management and process improvement;

f) Provide evidence that the processes were followed.

H.2 Operations on software life cycle data

The life cycle data should be supported by the following operations:

a) Create;

b) Read;

c) Update;

d) Delete.

H.3 Characteristics of software life cycle data

The life cycle data should adhere to the following characteristics:

a) Unambiguous: Information is unambiguous if it is described in terms that only allow a single interpretation, aided, if necessary, by a definition.

b) Complete: Information is complete if it includes necessary, relevant requirements and/or descriptive material, responses are defined for the range of valid input data, and terms and units of measure are defined.

c) Verifiable: Information is verifiable if it can be checked for correctness by a person or tool.

d) Consistent: Information is consistent if there are no conflicts within it.

e) Modifiable: Information is modifiable if it is structured and has a style such that changes can be made completely, consistently, and correctly while retaining the structure.

f) Traceable: Information is traceable if the origin of its components can be determined.

g) Presentable: Information is presentable if it can be retrieved and viewed.

H.4 Basic types of software life cycle data

The life cycle data should contain content in the following areas:

a) Requirements data: Expected functionality, operational context, performance constraints and expectations, basis for qualification testing, and key decision rationale.

b) Design data: Architecture, algorithms, design constraints, mapping to requirements, and key decision rationale.

c) Test data: Test strategy and criteria, cases (what to test), procedures (how to carry out tests), test results, and key decision rationale.

d) Configuration data: Configuration description, build instructions, reference to source code, reference to object code, data integrity approach, description of development environment, and key decision rationale.

e) User data: Software overview, system access information, commands and responses, error messages, operational environment, and key decision rationale.

f) Management data: Management plans, status reports, management indicators, criteria and key decision rationale, and contract and other procurement information.

g) Quality data: Quality plans and procedures, corrective action status, root cause analysis, product quality characteristics and process measurement data, and criteria and key decision rationale.

H.5 Presentation form of software life cycle data

The presentation form of life cycle data should

a) Be appropriate to support the purpose of the life cycle data;

b) Support the retrieval and review of data of a software item during its life cycle;

c) Support the basic operations on data of a software item during its life cycle;

d) Be selected subject to concurrence of the users of the data.

NOTE—In preparing or finalizing the contract, the acquirer should specify the requirements for data delivery, taking into account the maintenance strategy. Some of the choices are

1) Raw data—Repositories of the development tools such as CASE tools, databases, file systems, and other tool repositories.

2) On-line publishing systems—Data assembled and formatted for presentation by systems such as: word processors, World Wide Web publishing and display systems, SGML viewers.

3) Hard copy print—Traditional paper document form.

Annex I
(informative)
Relationships

This annex briefly describes the roles of IEEE Std 1074, ISO/IEC 12207, IEEE Std 1498, and ISO 9001. For more information on the relationships of these standards, refer to SESC-97-001, Harmonization Action Plan, which is available from the IEEE Computer Society, Washington, D.C.

I.1 IEEE Std 1074

IEEE Std 1074 is primarily written for an organizational process architect; the individual responsible for establishing the software life cycle to be followed on a particular project or set of projects. The standard requires that the process architect identify a set of available software life cycle models (e.g., prototyping, spiral, waterfall). The process architect then selects one of these models to be used on the project. The activities of the standard are then mapped in time order onto the model. Assigning owners and schedule to the activities completes what the standard defines as the project software life cycle. The standard is additionally intended to provide definition of each of the activities to its respective owner. IEEE Std 1074 would be particularly useful to a process architect in developing organizational processes complying with the requirements of ISO/IEC 12207.

I.2 ISO/IEC 12207

The International Standard ISO/IEC 12207 establishes a common framework for the life cycle of software in terms of the processes that can be employed to (1) acquire, supply, develop, operate, and maintain software; (2) manage, control, and improve the processes; and (3) provide the basis for world trade in software. ISO/IEC 12207 places requirements upon the characteristics of a designated set of life cycle processes, but does not specify the detailed implementation of those processes; a process architect may find IEEE Std 1074 to be useful in developing organizational processes complying with the requirements of ISO/IEC 12207.

I.3 IEEE Std 1498

IEEE Std 1498 was the first step in the adoption of ISO/IEC 12207. IEEE Std 1498 defines a set of software development activities and resulting software products. It provides a framework for the software development planning and engineering. The standard can be used as a basis for an acquirer-supplier agreement or for internal software engineering guidance.

I.4 ISO 9001

ISO 9001 establishes the generic requirements for a quality management system for the design, development, production, installation, and servicing of systems that include hardware and/or software. ISO 9000-3 should be used for guidance on the application of ISO 9001 to the development, supply and maintenance of software.

NOTE—ISO/IEC 12207 includes ISO 9001 as a normative reference.

Annex J
(normative)
Errata

This annex identifies those clauses in ISO/IEC 12207 that have ambiguities that could not be removed prior to its publication. They are presented here to minimize any possible misinterpretation or misapplication of this standard. Any suggested change is indicated in *italics*. All clauses and paragraphs refer to ISO/IEC 12207.

These identified clauses will be forwarded to the ISO/IEC JTC 1 for appropriate action.

1. Clause 1.2, paragraph 4, is clarified below.

   This clause does not imply that the suppliers or developers of off-the-shelf software should not use ISO/IEC 12207 when developing, operating, or maintaining such software.

2. Clause 1.5, paragraph 6 should read as follows:

   In this International Standard, there are a number of lists for tasks; none of these is presumed to be exhaustive—they are intended as examples *unless introduced by a clause containing a "shall" or a "will."*

3. Clause 5.1.3.5, sentence 2. "Shall" should be changed to "will."

4. Add the following clause to 5.3.1.2:

   e) Establish baselines for each configuration item at appropriate times, as determined by the acquirer and the supplier.

5. Delete sentence 2 of clause 5.3.4.3.

6. Delete clause 5.3.9.5.b.

7. Delete clause 5.3.11.4.b.

8. Clause 6.1, Preamble. The following should be added as a second paragraph to the preamble:

   Execution of this process by an organization results in the establishment of internal documentation standards (such as standards for program management plan and software design document) in a suitable media—paper, electronic, or other. The terms used in this process need to be interpreted accordingly for a given media or domain.

9. Clause 6.2, preamble, line 2. "Baseline" should be deleted. The resulting sentence should read as follows:

   The Configuration Management Process is a process of applying administrative and technical procedures throughout the software life cycle to: *identify and define* software items in a system; control modifications and releases of the items; record and report the status of the items and modification requests; ensure the completeness, consistency, and correctness of the items, and control storage, handling, and delivery of the items.

10. Clause 6.3.4.1. This clause should read as follows:

    Additional quality management activities shall be assured in accordance with the clauses of ISO 9001.

11. Clause 6.5.2. This clause depends heavily on testing (real-time executions) for validation. To allow flexibility, the following note should be added at the end of clause 6.5.2.5:

    NOTE—Other means besides testing (such as, analysis, modeling, simulation, etc.) may be employed for validation.

12. Clause 6.6.3.1.e. This clause should read as follows:

    e) They are ready for the next *planned* activity.