IEEE STANDARDS ASSOCIATION

IEEE Standard for Software Quality Assurance Processes

IEEE Computer Society

Sponsored by the
Software & Systems Engineering Standards Committee

IEEE
3 Park Avenue
New York, NY 10016-5997
USA

IEEE Std 730™-2014
(Revision of
IEEE Std 730-2002)

IEEE Std 730™-2014
(Revision of
IEEE Std 730-2002)

IEEE Standard for Software Quality Assurance Processes

Sponsor

Software & Systems Engineering Standards Committee
of the
IEEE Computer Society

Approved 27 March 2014

IEEE-SA Standards Board

Abstract: Requirements for initiating, planning, controlling, and executing the Software Quality Assurance processes of a software development or maintenance project are established in this standard. This standard is harmonized with the software life cycle process of ISO/IEC/IEEE 12207:2008 and the information content requirements of ISO/IEC/IEEE 15289:2011.

Keywords: assurance, conformance, contract, cycle, IEEE 730™, management, project, quality, requirements, software, SQA, standard

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

PDF: ISBN 978-0-7381-9168-3 STD98691
Print: ISBN 978-0-7381-9169-0 STDPD98691

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading "Important Notice" or "Important Notices and Disclaimers Concerning IEEE Standards Documents."

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents (standards, recommended practices, and guides), both full-use and trial-use, are developed within IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association ("IEEE-SA") Standards Board. IEEE ("the Institute") develops its standards through a consensus development process, approved by the American National Standards Institute ("ANSI"), which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. IEEE standards documents are supplied "AS IS" and "WITH ALL FAULTS."

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854 USA

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

Photocopies

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every ten years. When a document is more than ten years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE-SA Website at http://ieeexplore.ieee.org/xpl/standards.jsp or contact IEEE at the address listed previously. For more information about the IEEE-SA or IEEE's standards development process, visit the IEEE-SA Website at http://standards.ieee.org.

Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE-SA Website at the following URL: http://standards.ieee.org/findstds/errata/index.html. Users are encouraged to check this URL for errata periodically.

Patents

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 by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE-SA Website at http://standards.ieee.org/about/sasb/patcom/patents.html. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this IEEE standard was completed, the Software Quality Assurance Plans (C/S2ESC/730_WG) Working Group had the following membership:

Sue McGrath Carroll, *Chair*
John Walz, *Vice Chair*
Steven Rakitin, *Vice Chair*
J. David Blaine, *Editor*
Byron Frank, *Secretary*

T. Scott Ankrum
Bakul Banerjee
Linda Binkley
Pieter Botman
Yegor Bugayenko
Emile Captain
Irene Chan
Shumin Cheng
Milton Concepcion
Ken Costello
Cynthia Didio
Rebecca Draxten
Tom Ehrhorn
Eva Freund
Mike Gayle
Gregg Giesler
Marilyn Ginsberg-Finner
Robin Goldsmith
Michelle Gross
David Heimann
Bernard Homes
John Janeri
Mike Kress
Claude Laporte
Carla Lienhard
H. Clark Leiphart
George Lipscomb
Linda McInnis
Denis Meredith
Keith Middleton
Prabhu Parthasarathy
Michelle Pierce
Mark Paulk
Nancy Phillips
Sheila Ray
Annette Reilly
Jeanne Ruggles
Robert Schaaf
Kandy Senthilmaran
Maud Schlich
Peter Schulz
Jeet Shangari
Maury Skibba
Lisa Smith
Malcolm Stiefel
Teri Stokes
John Suzuki
Bill Trest*
Vincent Tume
Linda Wharton
Dan Zrymiak

* Deceased

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Edward Addy
Johann Amsenga
T. Scott Ankrum
Angela Anuszewski
Chris Bagge
Bakul Banerjee
Pieter Botman
Bill Brown
Lyle Bullock
Juan Carreon
Sue McGrath Carroll
Lawrence Catchpole
Keith Chow
Paul Croll
Geoffrey Darnton
Ray Davis
Ronald Dean
Andrew Fieldsend
Andre Fournier
Byron Frank
Eva Freund
David Friscia
David Fuschi
Gregg Giesler
Robin Goldsmith
Randall Groves
Louis Gullo
John Harauz
David Heimann
Mark Henley
David Herrell
Rutger A. Heunks
Werner Hoelzl
Bernard Homes
Peter Hung
Noriyuki Ikeuchi
Atsushi Ito
Mark Jaeger
Cheryl Jones
Piotr Karocki
John Kay
Thomas Kurihara
Susan Land
Claude Laporte
Michael Lauxman
David Leciston
H. Clark Leiphart
Claire Lohr
Greg Luri
Edward McCall
James Moore
Michael Newman
Charles Ngethe
Mark Paulk
Robert Peterson
William Petit
Ulrich Pohl
Steven Rakitin
Annette Reilly
Robert Robinson
Terence Rout
Randall Safier
Bartien Sayogo
Robert Schaaf
Hans Schaefer
David Schultz
Stephen Schwarm
Raymond Senechal
Gil Shultz
Carl Singer
Kapil Sood
Luca Spotorno

Friedrich Stallinger
Thomas Starai
Eugene Stoudenmire
Walter Struppler
Gerald Stueve
Marcy Stutzman
Chandrasekaran Subramaniam
John Suzuki
Thomas Tullia
Vincent Tume
Charlene Walrad
John Walz
Jingxin Wang
Michael Waterman
Stephen Webb
M. Karen Woolf
Oren Yuen
Janusz Zalewski
Daidi Zhong

When the IEEE-SA Standards Board approved this standard on 27 March 2014, it had the following membership:

John Kulick, *Chair*
Jon Walter Rosdahl, *Vice-chair*
Richard H. Hulett, *Past Chair*
Konstantinos Karachalios, *Secretary*

Peter Balma
Farooq Bari
Ted Burse
Clint Chaplain
Stephen Dukes
Jean-Phillippe Faure
Gary Hoffman
Michael Janezic
Jeffrey Katz
Joseph L. Koepfinger*
David Law
Hung Ling
Oleg Logvinov
Ted Olsen
Glenn Parsons
Ron Peterson
Adrian Stephens
Peter Sutherland
Yatin Trivedi
Phil Winston
Don Wright
Yu Yuan

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Richard DeBlasio, *DOE Representative*
Michael Janezic, *NIST Representative*

Michelle Turner
IEEE Standards Program Manager, Document Development

Malia Zaman
IEEE Standards Program Manager, Technical Program Development

Introduction

This introduction is not part of IEEE Std 730™-2014, IEEE Standard for Software Quality Assurance Processes.

IEEE Std 730 has been a benchmark for Software Quality Assurance (SQA) professionals since it was first published in 1979. While previous versions of IEEE Std 730 provided an SQA plan outline this revision expands the scope of this standard to address the processes defined in software life cycle framework standard, ISO/IEC/IEEE 12207:2008. This change in emphasis is consistent with and elaborates the process requirements in ISO/IEC/IEEE 12207:2008.

Contents

IEEE Standard for Software Quality Assurance Processes

1. Overview

1.1 Scope

This standard establishes requirements for initiating, planning, controlling, and executing the Software Quality Assurance (SQA) processes of a software development or maintenance project. This standard is harmonized with the software life cycle process of ISO/IEC/IEEE 12207:2008[1] and the information content requirements of ISO/IEC/IEEE 15289:2011.

NOTE—Annex A presents detailed explanations and mappings between ISO/IEC/IEEE 12207:2008 and IEEE Std 730™-2014 subclauses.[2]

1.2 Purpose

The activities described in this standard are intended to enable the software project to use the SQA processes to produce and collect evidence that form the basis for giving a justified statement of confidence that the software product conforms to its established requirements. The purpose of this standard is to provide uniform, minimum acceptable requirements for SQA processes in support of a software project. In considering adoption of this standard, regulatory bodies should be aware that specific application of this standard may already be covered by one or more IEEE or ANSI (American National Standards Institute)

[1] Information on normative references can be found in Clause 2.
[2] Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

standards documents relating to quality assurance, definitions, or other matters. It is not the purpose of this document to supersede, revise, or amend existing standards directed to specific industries or applications.

1.3 Field of application

This standard guides SQA activities for software products or services. This standard is applicable to software as part of a system as well as standalone software products. This standard regards the field of software engineering to include projects to develop new or enhanced products, maintenance of existing products, and software integration projects.

This standard is aligned with the outcomes specified in ISO/IEC/IEEE 12207:2008 and ISO/IEC/IEEE 15289:2011. This standard is not restricted by size, complexity, criticality, or application of the software product.

1.4 Limitations

This standard specifies definitions, activities, tasks, and outcomes for SQA processes. This standard does not impose constraints on the implementation or performance of those activities and tasks.

1.5 Conformance

1.5.1 Conformance language conventions

The word *shall* is used to express a requirement, *should* to express a recommendation, and *may* to express alternative or optional methods of satisfying a requirement.

1.5.2 Conformance scope

Conformance to this standard is achieved by demonstrating that the requirements of Clause 5, indicated by the use of *shall*, are satisfied.

Conformance to this standard is a sufficient condition to meet the SQA outcomes enumerated in 7.2.3 of ISO/IEC/IEEE 12207:2008. The converse is not true—meeting all requirements of 7.2.3 of ISO/IEC/IEEE 12207:2008 is not sufficient for the SQA work and output to meet all requirements of this standard. Finally, conformance to this standard is not sufficient to meet other clauses of ISO/IEC/IEEE 12207:2008 or any other standard in whole or in part.

There are two ways that projects or organizations can claim conformance to the provisions of this standard: full conformance and tailored conformance as explained in 1.5.3 and 1.5.4 below.

1.5.3 Full conformance

Full conformance to this standard is achieved by demonstrating that all of the requirements of Clause 5 are satisfied, using the required outcomes as evidence.

1.5.4 Tailored conformance

When this standard is used as a basis for establishing a set of activities that do not qualify for full conformance, the clauses of this standard are selected or modified in accordance with the tailoring process prescribed below.

To conform to this standard, only the following modifications are allowed: a project team may limit the scope of the product and process assurance activities described in 5.4 and 5.5 in order to align with the project's activities, products, and services. Any SQA activity, task, or outcome required by this standard that is not performed is identified in the project SQA Plan as not applicable along with a justification for why it is not performed.

NOTE—When this standard is used to help develop an agreement between an acquirer and a supplier, clauses of this standard can be incorporated into the agreement with or without modification.

1.6 Intended usage of this standard

The requirements in this standard are contained in normative Clause 5. This standard specifies requirements for activities that may be executed during the life cycle of a software product or service. It is recognized that particular projects or organizations may not need to use all of the activities in this standard. Therefore, implementing this standard typically involves selecting a set of activities suitable to the organization or project as described in 1.5.

1.7 Organization of this standard

This standard is organized into clauses and annexes:

- Clause 1 contains scope, purpose, and introductory material.
- Clause 2 identifies the normative references used in this standard.
- Clause 3 defines terms and acronyms used in this standard.
- Clause 4 describes the context for the SQA processes and SQA activities, and covers expectations for how this standard will be applied.
- Clause 5 specifies the SQA processes, activities, and tasks. Sixteen activities are identified in this clause and are grouped into three major areas: process implementation, product assurance, and process assurance. These activities implement the required outcomes for SQA specified by 7.2.3 of ISO/IEC/IEEE 12207:2008.
- Refer to Annex A for the mapping of the four required outcomes to the subclauses of this standard.
- Refer to Annex B for information about mapping between the SQA plan outlines in IEEE Std 730-2002[3,4] and IEEE Std 730-2014.
- Refer to Annex C for information about guidance for creating a Software Quality Assurance Plan (SQAP).
- Refer to Annex D for information about mapping between IEEE Std 730-2014 and ISO/IEC 15504-1:2004 [B34].[5]

[3] IEEE publications are available from the Institute of Electrical and Electronics Engineers (http://standards.ieee.org/).
[4] The IEEE standards or products referred to in this clause are trademarks of the Institute of Electrical and Electronics Engineers, Inc.
[5] Information in brackets corresponds to those of the bibliography in Annex L.

— Refer to Annex E for information about applying IEEE Std 730-2014 to specific industries.

— Refer to Annex F for information about SQA's relationship to agile development methods.

— Refer to Annex G for information about mapping between IEEE Std 730-2014 and ISO/IEC 29110:2011 [B40].

— Refer to Annex H for information about validating software tools.

— Refer to Annex I for information about software integrity levels and assurance cases.

— Refer to Annex J for examples of corrective action, preventive action, and root cause analysis processes.

— Refer to Annex K for a cross-reference of IEEE Std 730-2014 SQA Activities and ISO/IEC/IEEE 12207:2008 Life Cycle Processes.

— Refer to Annex L for the bibliography of this standard.

Each of the activities in Clause 5 of this standard is described by the following information:

a) Reference to ISO/IEC/IEEE 12207:2008 clause. The text of the ISO/IEC/IEEE 12207:2008 subclause relevant to the activity is presented in a boxed figure, for example:

   This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

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

b) Purpose. Defines the activity's intention (e.g., "Determine the degree to which the software products and related documentation conform to established requirements.").

c) Outcomes. Specific observable results of successful achievement of activity's purpose, measurable and tangible business, or technical results. An outcome may be a software executable, a physical artifact, an information item (e.g., records, documents), a change in the state or an attribute of an information item, a change to a project constraint, or a change to an attribute (e.g., training, experience, capability) of a project team member. Information items are defined in ISO/IEC/IEEE 15289:2011. These outcomes are the basis for validated evidence to provide a justified statement of confidence in the quality of the software products. Outcomes are written as declarative sentences in the present tense (e.g., "Non-conformances are raised when software products do not conform to established software requirements.").

d) Tasks. Specific actions intended to contribute to the achievement of one or more of the stated outcomes of the activity. Using the definitions from ISO/IEC TR 24774:2010 [B41], a statement in the task description can be:

   1) A required action,

   2) A recommended action, or

   3) A permissible action.

Task statements identify the role performing the task. The role is either the subject of the sentence or is in a clause that introduces a set of tasks. Task statements for required actions begin with an action verb and are written as declarative sentences in the present tense. Recommended and permissible action statements start with a phrase or clause that qualifies the conditions under which the task may be performed. There is not necessarily a one-to-one relationship between tasks and outcomes. (e.g., Identify software products and related documentation required by the contract.)

The SQAP is the key document that defines the activities to be performed on a specific project. The SQA function's planning activities and the SQAP outline are described in 5.3.3.

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

ISO/IEC/IEEE 12207:2008 Systems and Software Engineering — Software Life Cycle Processes.

ISO/IEC/IEEE 15289:2011 Systems and Software Engineering — Content of Life-Cycle Information Products (documentation).

3. Definitions, acronyms, and abbreviations

For the purposes of this document, the following terms and definitions apply. ISO/IEC/IEEE 24765:2010 Systems and Software Engineering — Vocabulary [B42], the IEEE Standards Dictionary Online,[6] and IEEE Computer Society's Software and Systems Engineering Vocabulary[7] should be consulted for terms not defined in this clause.

3.1 Conventions

Throughout this standard, unless specifically noted otherwise, the term "SQA" refers to either the SQA function or the SQA process. The term "SQA function" is not to be interpreted as a particular person, tool, document, job title, or a specific group dedicated to SQA, regardless of how that function is staffed, organized, or executed. The term "SQA process" means a set of activities.

3.2 Definitions

Acquirer: A stakeholder that acquires or procures a product or service from a supplier [ISO/IEC/IEEE 12207:2008].

Activity: A set of cohesive tasks of a process, which transforms inputs into outputs [ISO/IEC/IEEE 12207:2008].

Assurance: Grounds for justified confidence that a claim has been or will be achieved (ISO/IEC 15026-1:2013 [B36]).

Assurance case: Representation of a claim or claims, and the support for these claims (ISO/IEC 15026-1:2013 [B36]).

NOTE—An assurance case is a reasoned, auditable artifact created to support the contention its claim or claims are satisfied. It contains the following and their relationships: one or more claims about properties; arguments that logically link the evidence and any assumptions to the claim(s); a body of evidence; and possibly assumptions supporting these arguments for the claim(s).

Assure: To promise or state with certainty by one person to another person or group. Contrast with **ensure**.

[6] *IEEE Standards Dictionary Online* subscription is available at: http://www.ieee.org/portal/innovate/products/standard/standards_dictionary.html.
[7] The current version of this dictionary is available at: http://www.computer.org/sevocab.

Audit: An independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria (ISO/IEC/IEEE 24765:2010 [B42]).

Compliance: Doing what has been asked or ordered; as required by rule or law (e.g., comply with a regulation).

Conformance: The fulfillment by a product, process, or service of specified requirements (adapted from ISO/IEC/IEEE 24765:2010 [B42]). For conformance to be meaningful, the specified requirements accurately represent stakeholder requirements.

Constraint: A limitation or implied requirement that constrains the design solution or implementation of the systems engineering process and is not changeable by the enterprise (ISO/IEC/IEEE 24765:2010 [B42]). A design constraint is an explicit and direct restriction regarding the choice of design ideas. It either declares a design idea to be compulsory or to be excluded.

Contract: A binding agreement between two parties, especially enforceable by law, or a similar internal agreement wholly within an organization [ISO/IEC/IEEE 12207:2008]. A contract is an agreement between two or more parties regarding a course of action. The formality of a contract can range from a simple informal oral description to a formal written instrument. This standard calls an agreement between software acquirer and supplier a contract.

Deliverable: Item to be provided to an acquirer or other designated recipient as specified in an agreement (ISO/IEC/IEEE 24765:2010 [B42]). This item can be a document, hardware item, software item, service, or any type of work product.

Document: A uniquely identified unit of information for human use, such as a report, specification, manual, or book, in printed or electronic form [ISO/IEC/IEEE 15289:2011].

Ensure: To make certain that things occur or events take place. Contrast with **assure**.

Established requirement: A requirement that the project has verified as satisfying project-specific criteria (such as clarity, suitability, and feasibility) and has validated to be an accurate representation of stakeholder needs, wants, and expectations. Established requirements are accepted by the project to form the basis of product development.

Function: In a software application, a module that performs a specific action. In an organization, a function is a set of resources and activities that achieve a particular purpose.

Functional requirement: A requirement that specifies a function that a system or system component must perform (ISO/IEC/IEEE 24765:2010 [B42]).

Independence [of SQA]: Situation in which SQA is free from technical, managerial, and financial influences (intentional or unintentional).

Independence, financial [of SQA]: Situation in which control of the SQA budget is vested in an organization independent of the development organization. (IEEE Std 1012™-2012 [B26]).

Independence, managerial [of SQA]: Situation in which the responsibility of the SQA effort is vested in an organization separate from the development and project management organizations. (IEEE Std 1012-2012 [B26]).

Independence, technical [of SQA]: Situation in which the SQA effort uses personnel who are not involved in the development of the system or its elements. (IEEE Std 1012-2012 [B26]).

Information item: A separately identifiable body of information that is produced, stored, and delivered for human use during a system or software life cycle [ISO/IEC/IEEE 15289:2011].

Integrity level: A value representing project-unique characteristics (e.g., complexity, criticality, risk, safety level, security level, desired performance, reliability) that define the importance of the system, software, or hardware to the user. (IEEE Std 1012-2012 [B26]).

Life cycle model: A framework of processes and activities concerned with the life cycle that may be organized into stages, which also acts as a common reference for communication and understanding [ISO/IEC/IEEE 12207:2008].

Measure: A variable to which a value is assigned as the result of measurement (IEEE Std 15939™-2008 [B28]).

Non-functional: As applied to requirements, the term "non-functional" is deprecated and is not used in this standard. The terms "performance requirement" or "performance attribute" are preferred.

Performance requirement: The measurable criterion that identifies a quality attribute of a function or how well a functional requirement must be accomplished (IEEE Std 1220™-2005 [B27]). A performance requirement is always an attribute of a functional requirement.

Plan: An information item that presents a systematic course of action for achieving a declared purpose, including when, how, and by whom specific activities are to be performed [ISO/IEC/IEEE 15289:2011].

Process: A set of interrelated or interacting activities which transforms inputs into outputs [ISO/IEC/IEEE 12207:2008]; a process can exist whether it is documented or not.

Product: The result of a process; an artifact that is produced, is quantifiable, and can be either an end item in itself or a component item (*PMBOK® Guide* [B48]).

NOTE—includes intermediate as well as end-item artifacts; complete set of software and documentation; output of software development activities.

Project: A temporary endeavor to develop a unique product, service, or result (*PMBOK® Guide* [B48]).

Quality: The degree to which a product or process meets established requirements; however, quality depends upon the degree to which those established requirements accurately represent stakeholder needs, wants, and expectations (adapted from ISO/IEC/IEEE 24765:2010 [B42]).

Record: A set of related data items treated as a unit [ISO/IEC/IEEE 15289:2011].

Requirement: A condition or capability that is to be met or possessed by a system, product, service, result, or component to satisfy a contract, standard, specification, or other formally imposed document. Requirements include the quantified and documented needs, wants, and expectations of the sponsor, customer, and other stakeholders (*PMBOK® Guide* [B48]). Requirements provide value when delivered, satisfied, or met.

Review: A process, which may include a meeting, in which work products are presented to some stakeholders for comment or approval (adapted from ISO/IEC/IEEE 24765:2010 [B42]).

Software engineering environment (SEE): The hardware, software, and firmware used to perform a software engineering effort (ISO/IEC/IEEE 24765:2010 [B42]).

Software integrity level: *See:* **Integrity level**.

Software life cycle (SLC): The project-specific sequence of activities that is created by mapping the activities of a standard onto a selected software life cycle model (ISO/IEC/IEEE 24765:2010 [B42]).

Software quality: The degree to which a software product meets established requirements; however, quality depends upon the degree to which those established requirements accurately represent stakeholder needs, wants, and expectations (adapted from ISO/IEC/IEEE 24765:2010 [B42]).

Software Quality Assurance: A set of activities that define and assess the adequacy of software processes to provide evidence that establishes confidence that the software processes are appropriate for and produce software products of suitable quality for their intended purposes. A key attribute of SQA is the objectivity of the SQA function with respect to the project. The SQA function may also be organizationally independent of the project; that is, free from technical, managerial, and financial pressures from the project.

Software quality management: Coordinated activities to direct and control an organization with regard to software quality.

Software testing: (1) An activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component. (2) To conduct an activity as in (1). (IEEE Std 829™-2008 [B25]).

Software testing environment: The facilities, hardware, software, firmware, procedures, and documentation needed to perform…testing of software (ISO/IEC/IEEE 24765:2010 [B42]).

Software validation: (1) The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements. (2) The process of providing evidence that the system, software, or hardware and its associated products satisfy requirements allocated to it at the end of each life cycle activity, solve the right problem (e.g., correctly model physical laws, implement business rules, and use the proper system assumptions), and satisfy intended use and user needs. (3) The confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application are fulfilled. (ISO/IEC/IEEE 12207:2008)

Software verification: (1) The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. (2) The process of providing objective evidence that the system, software, or hardware and its associated products conform to requirements (e.g., for correctness, completeness, consistency, and accuracy) for all life cycle activities during each life cycle process (acquisition, supply, development, operation, and maintenance); satisfy standards, practices, and conventions during life cycle processes; and successfully complete each life cycle activity and satisfy all the criteria for initiating succeeding life cycle activities. (3) The confirmation, through the provision of objective evidence, that specified requirements are fulfilled. (ISO/IEC/IEEE 12207:2008)

NOTE—"Verified" designates the corresponding status. In design and development, verification includes examining the result of a given activity to determine conformity with the stated requirement for that activity. A system may be verified to meet the stated requirements, yet be unsuitable for operation by the actual users (ISO 9000:2005 [B30]).

Specification: An information item that identifies, in a complete, precise, and verifiable manner, the requirements, design, behavior, or other expected characteristics of a system, service, or process [ISO/IEC/IEEE 15289:2011].

Supplier: An organization or individual that enters into an agreement with an acquirer for the supply of a product or service [ISO/IEC/IEEE 12207:2008].

Task: A required, recommended, or permissible action intended to contribute to the achievement of one or more outcomes of a process [ISO/IEC/IEEE 12207:2008].

Work product: An artifact resulting from the execution of a process (ISO/IEC 15504-1:2004 [B34]).

3.3 Acronyms and abbreviations

IEC International Electrotechnical Commission

ISO International Organization for Standardization

PMBOK Project Management Body of Knowledge

QA Quality assurance

QC Quality control

QE Quality engineering

SEE Software engineering environment

SLC Software life cycle

SLCM Software life cycle model

SQA Software Quality Assurance

SQAP Software Quality Assurance Plan

STE Software test environment

4. Key concepts of Software Quality Assurance

4.1 Organizational management responsibility

Management support of the SQA function is essential for SQA processes to be effective. This support minimally includes:

— Management is familiar with and understands the SQA function's purposes, concepts, practices, and needs.

— Management provides the SQA function with an appropriate level of skilled resources (people, equipment, knowledge, methods, facilities, and tools) in order to accomplish their project responsibilities.

— Management receives and acts upon information provided by the SQA function throughout the course of a project.

The organization's management supports the SQA function by providing the outcomes listed in 5.3.1.2.

4.2 Organization and project relationship

Projects are performed by people working within an organization. The scope of an organization can range from a very small group that has only one project to a multi-national conglomerate with many divisions and

many projects within each division. SQA can be described from the perspective of an acquirer or supplier. Except as specifically noted otherwise, this standard describes the SQA function from the supplier viewpoint.

Although this standard is focused on the project, this standard refers to some organizational-level activities. This standard presumes that some organizational-level activities establish the SQA processes when, or prior to, initiating the project. That initiating activity is the subject of 5.3.1. Additionally, if the organization expects to execute more than one project, then other organizational-level activities are concerned with capturing lessons from the current project and ensuring that those lessons are available for subsequent projects.

Figure 1 illustrates the time-oriented relationship of work performed at the organizational-level versus work performed at the project level. The figure emphasizes that the scope of this standard is a single project. This project is performed within an organization.

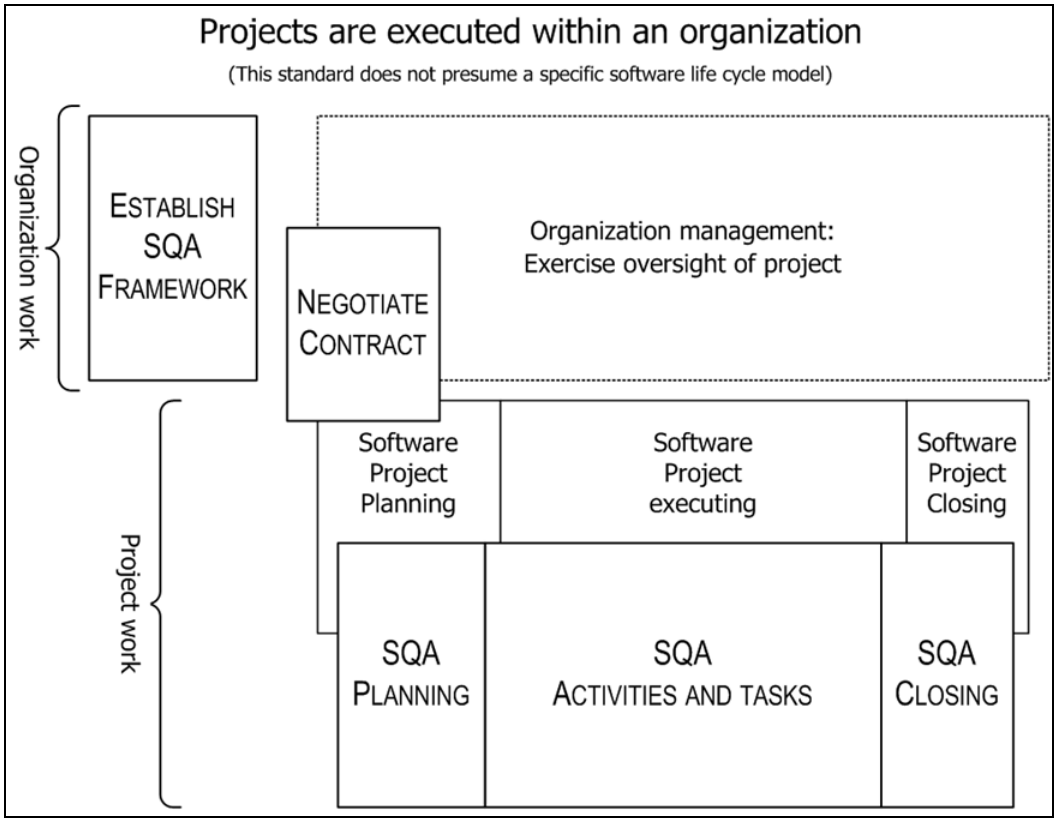

Figure 1—Relationship between the organization and the project

The project is performed to accomplish the business goals of the organization. The organization sponsors the project by providing people and resources. The organization initiates the project and, during project initiating, work at the organization level overlaps with work at the project level. During the course of the project, the organization exercises oversight of the project. The organizational-level work of capturing and applying lessons learned occurs during execution and continues through closing.

This standard does not presume or impose any particular project life cycle model. This standard is applicable to all software life cycle models. However, this standard does acknowledge and use generally recognized project management practices as described in the *PMBOK® Guide* [B48].

In addition to being executed by an organization, a project can be part of a product life cycle, as illustrated in Figure 2. The software project is completed upon the acceptance and release of the product. Subsequently, life cycle support activities are part of the product life cycle model. As post-release problems are reported, new software projects might be executed within the product life cycle support phase to produce new versions of the product, whose purpose could be to fix problems or introduce new features. Software projects can involve developing software, acquiring software, supporting software, or combinations of all three.

The project SQA role ends with project closing. The organization can provide post-project SQA support. The organization-level SQA can provide continuity and communications among many projects.

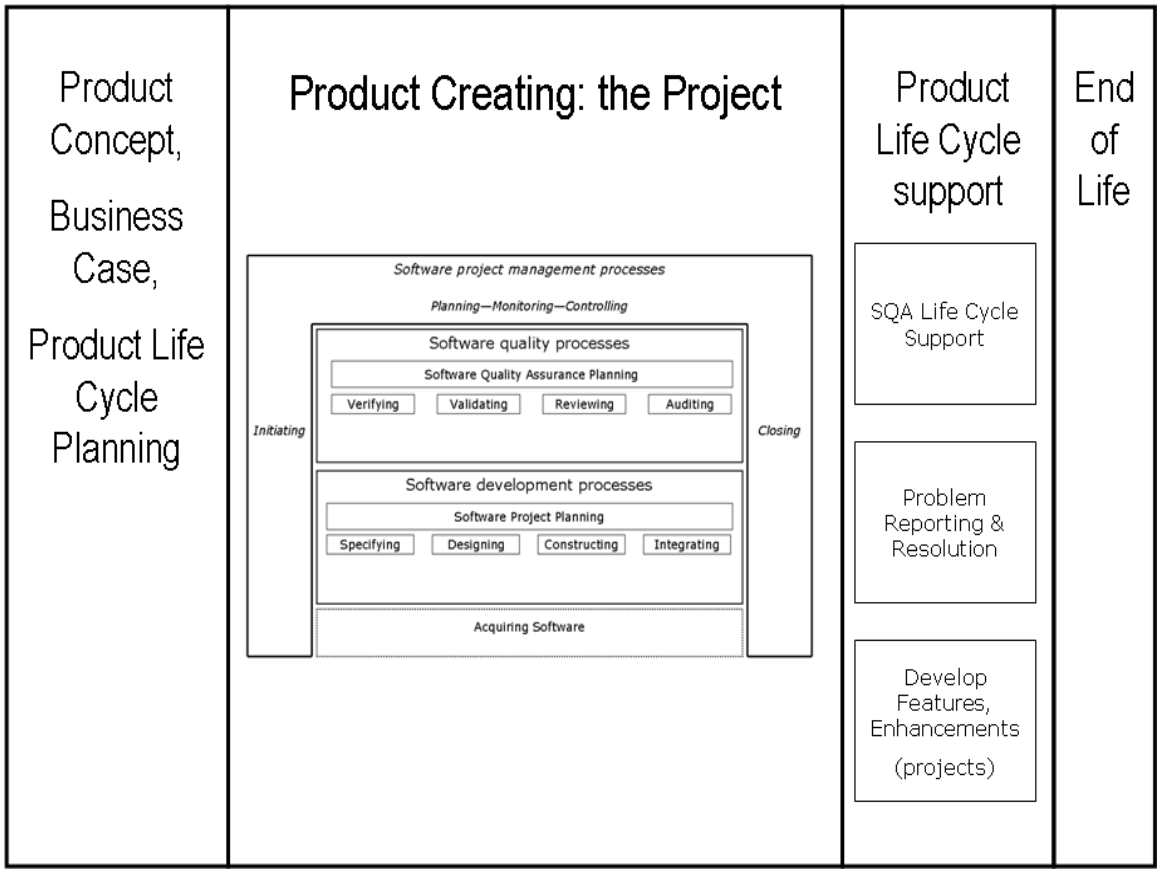

Figure 2—Example product life cycle, context for a software project

This standard may be applied at the organization or project level. Organizations that produce software strive to develop a set of software life cycle processes that are consistently applied to the organization's projects. An organization may define appropriate procedures, practices, and policies that conform to this standard. A software project would then conform to the organization's processes rather than directly to this standard. Organizations can create and structure their quality management system in various ways; the specific structure of this standard is not imposed.

A project may be executed in an organization that does not have an organizational-level set of software life cycle processes. Such a project may apply the provisions of this standard directly to the project.

4.3 Software quality and relationship to requirements

Quality is the entire set of attributes that gives a software product the ability to satisfy expressed or implied stakeholder requirements. These stakeholder requirements become refined into software requirements, including functional requirements and performance attributes that specify how well the software performs the functional requirements.

The project determines that requirements satisfy criteria such as clarity and testability and that those requirements accurately represent stakeholder needs, wants, and expectations. This standard defines software quality as conformance to requirements that have been established by the project.

This standard defines the scope of SQA as: 1. Assessing the software development process; 2. Evaluating the conformance to software processes; and 3. Evaluating the effectiveness of the software processes. These processes include those that identify and establish the software requirements, develop the software product, and maintain the software product.

Software quality is achieved by conforming to established requirements as described in 5.4.3. In some industries, conformance to requirements also includes the constraint that the product meets its intended use and is evaluated in the actual use setting. Because requirements are the basis for quality, the SQA function pays special attention to those software processes that evaluate and respond to the requirements as described in 5.5.2. Those project processes could utilize ISO/IEC/IEEE 29148:2011 [B43], which defines verification criteria for requirements including clarity, consistency, completeness, and verifiability. The ISO/IEC/IEEE 29148:2011 standard presents detailed attributes that projects can use for verifying requirements.

The SQA function confirms that the software processes can and do produce software that conforms to requirements. Confirming includes evaluating intermediate and final software products along with methods, practices, and workmanship. Evaluation further includes measurement and analysis of software process and product problems and related causes as well as recommendations about ways to correct current problems and prevent future problems.

Requirements can be categorized as either process or product requirements. Process requirements specify the processes the project will use to produce the project outcomes. Process requirements may include specific processes mandated to be in place, specific tasks the project or organization is mandated to perform, and the manner in which specific tasks are to be performed. Product requirements specify the functions that a product is mandated to perform and attributes that a product is mandated to possess. These attributes include performance attributes that specify how well the product should perform. Product requirements, also called system requirements, can be allocated to software or non-software aspects of the project. Both process and product requirements are derived from and are a response to stakeholder requirements.

Figure 3 shows the processes from ISO/IEC/IEEE 15289:2011 involved in requirements derivation. The figure illustrates the distinction among stakeholder, process, and product requirements, and shows the process flow leading to both process requirements and software requirements.

This standard presumes that the requirements derivation activities shown in Figure 3 are executed, but may not be fully completed prior to the start of the software project. The SQA activities and tasks in this standard presume that software requirements are established prior to being used by the SQA function to evaluate conformance of the software product to those requirements. There is no implication that all software requirements are established at any point in the project, only that the subset used for a particular conformance evaluation is established. This standard recognizes that some software projects include activities that create or refine requirements. Some projects are initiated with only an informal and incomplete set of requirements. Regardless of the starting point, requirements can be continually elicited, documented, validated, and implemented throughout the project.

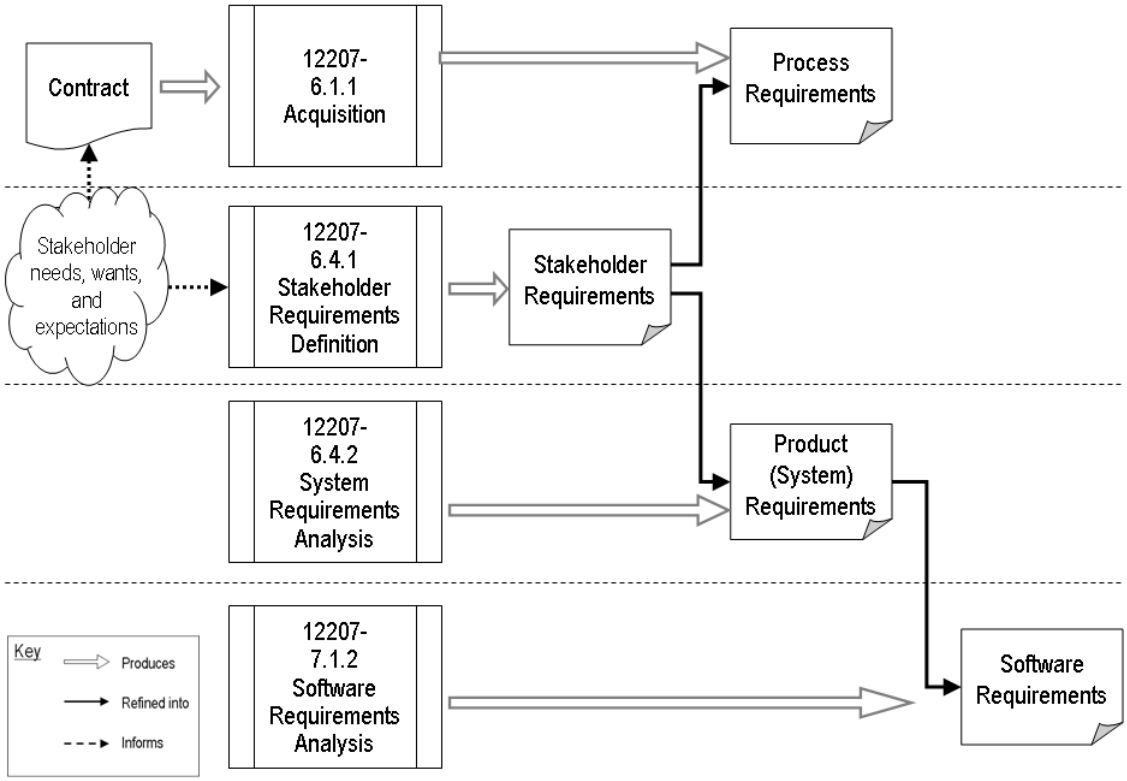

Figure 3—Derivation of requirements and relationships between levels of requirements

The SQA function's responsibility is to produce and collect evidence that forms the basis for giving a justified statement of confidence that the software product conforms to its established requirements. This responsibility is based on the definition of quality adopted by this standard: quality is conformance to established requirements. Assuring stakeholders that their expectations will be met is based on the conformance relationships illustrated in Figure 4 as well as additional SQA processes that determine the efficacy of the software processes.

The ISO/IEC/IEEE 12207:2008 standard, in 7.1.2, Stakeholder Requirements Definition Process, describes activities and tasks that produce requirements for the software project. Once reviewed and accepted by the project, these requirements become established requirements. The SQA function reviews established requirements as well as changes to established requirements.

4.4 Overview of SQA activities

Once the contract has been approved and the requirements established, the SQA function evaluates the compliance and conformance relationships shown in Figure 4. The scope of Figure 4 is the project.

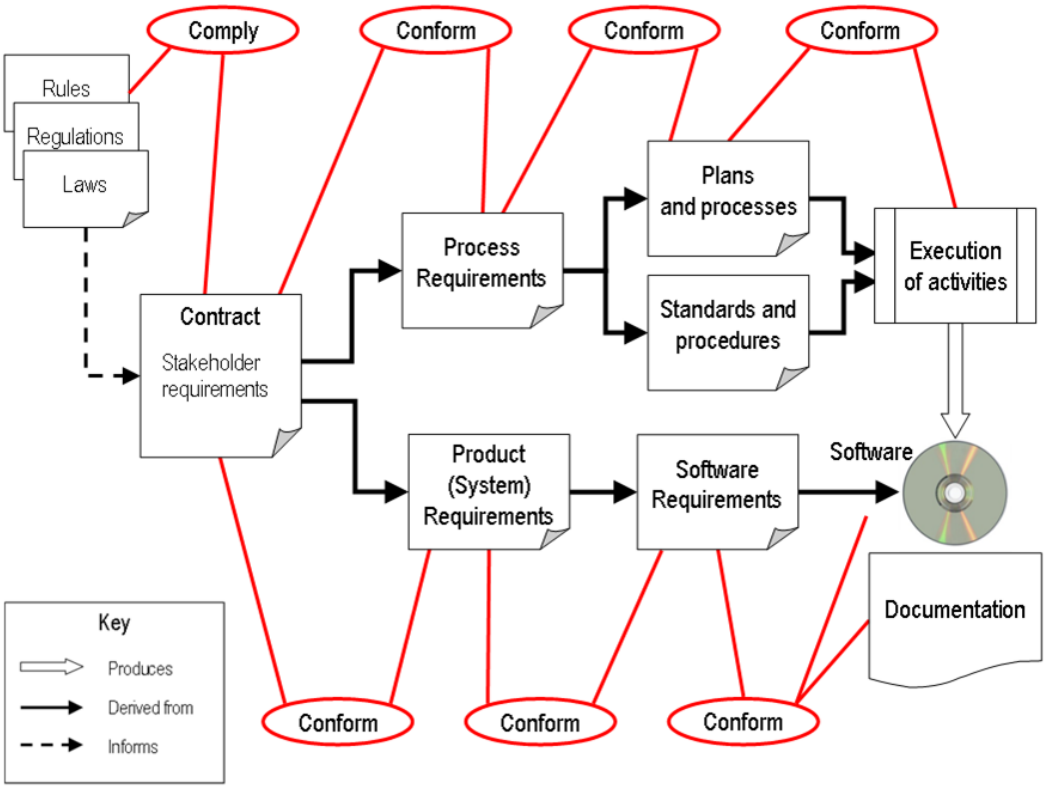

Figure 4—Relationships for determining conformance between project artifacts.

Figure 4 shows how project artifacts are derived from the contract. Process and product requirements are derived from the contract. Software requirements are derived from product requirements; software products are based on software requirements. Similarly, the project's processes and plans are derived from the established process requirements. The project's activity execution is based on the processes and plans. Figure 4 also shows that the contract is required to comply with some artifacts external to the contract: rules, regulations, and laws.

Figure 4 serves two purposes: 1. The figure illustrates the two fundamental conformance categories of product assurance and process assurance; and 2. The figure shows the transitivity of the conformance relationships. That is, if process requirements conform to the contract, and the project's processes and plans conform to process requirements, then the processes and plans conform to the contract. This simplifies SQA's job. Each project artifact need not be checked against the contract. Each artifact need only be checked against its immediate predecessor.

Software product assurance activities check that software products conform to software product requirements, and that those software product requirements conform to system product requirements that in turn conform to stakeholder requirements as shown in the contract.

Similarly, process assurance activities check that the execution of project activities conform to the project's processes and plans, that those processes and plans conform to process requirements, and that those process requirements conform to stakeholder requirements as shown in the contract. In addition, process assurance

includes checking that the process requirements comply with any laws, regulations, and rules imposed on the project.

In Clause 5, this standard describes the activities for these two categories. In particular, the project's Software Quality Assurance Plan (SQAP) defines the project's product and process assurance activities, tasks, and outcomes, as described in 5.3.3.

4.5 Acquirer and supplier perspectives

For the purpose of this standard, software development is considered to be carried out by a supplier and delivered to an acquirer according to the terms of a contract.

During the course of the software development life cycle, the contract may evolve, as long as both acquirer and supplier agree to the changes. This aspect is especially prevalent when iterative or incremental development strategies are used.

4.6 Key concepts of this standard

4.6.1 Defining the software quality assurance role

The roles and responsibilities of an organizational unit called Quality Assurance (QA), Software Quality Assurance (SQA), Quality Engineering (QE), or some other variant, differ across industry sectors and businesses. They might perform software testing, product assurance, process assurance, or some combination of these, or SQA as defined in this standard. Organizational units with the name SQA might or might not carry out the SQA role as defined in this standard.

While this standard does not require the prescribed activities be executed by any specific organizational unit, it does require that clear responsibility be assigned to resources to perform the SQA activities described.

The resources carrying out the SQA function determine whether project activities were performed in accordance with the project's processes and plans. Organizational Quality Management may provide oversight of the SQA function.

4.6.2 Software product risks

Software systems are increasingly developed to perform tasks that can cause harm to living things, physical structures, and the environment. A fundamental principle of this standard is to first understand software product risk and then to ensure that the planned SQA activities are commensurate with product risk. This means that the breadth and depth of SQA activities defined in the SQA Plan are determined by and derived from software product risk.

Two techniques for addressing software product risks are discussed in Annex I.

4.6.3 Relationship between systems and software

This standard recognizes that software is part of a system. It is based upon the general principles of systems engineering. The allocation of functions to hardware or software is a design decision that occurs within the overall system development process. The SQA function includes aspects of the process leading to such

allocation decisions. Software is treated as an integral part of the total system and performs certain functions in that system. Software is implemented by deriving the software requirements from the system requirements and design, producing the software, and integrating it into the system. A fundamental premise of this standard is that software always exists within the context of a system, even if the system consists of only the processor upon which the software is executed. Therefore, a software product is always treated as one item in a system. If a product is software-only, it would still be considered a system.

4.7 Software process improvement

This standard aids an organization's software process improvement efforts. The measurement activities and tasks can supply objective data to an organization's Life Cycle Management Process (6.2 of ISO/IEC/IEEE 12207:2008). Additionally, at all occurrences of evaluating the conformance and effectiveness of processes, improvement opportunities can be identified and reported to the organization's process improvement function. Similarly, evaluating software products for conformance can identify improvement opportunities.

An organization should base process improvement efforts on the results of in-process as well as completed projects, gathering lessons learned, and the results of ongoing SQA activities such as process assessments and reviews. Process documentation should be updated to reflect the identified improvements.

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 ongoing or subsequent projects, and to determine technology advancement needs.

Quality cost data should be collected, maintained, and used to improve the organization's processes as a management activity. This data serves the purpose of establishing the cost of both the prevention and resolution of problems in software products, documentation, and services.

4.8 Maintaining quality management system consistency

The SQA processes described in this standard are intended to provide evidence that serves as the basis for a justified statement of confidence that the quality of the organization's (or the project's) quality management system can produce software that conforms to established requirements.

The ISO 9001:2008 [B31] standard specifies requirements imposed on quality management systems. Guidance for the application of ISO 9001 to software can be found in IEEE Std 90003™-2008 [B29]. The ISO/IEC/IEEE 12207:2008 standard, in 6.2.5, provides a reference model for the activities of a quality management system.

5. Software Quality Assurance process

5.1 Purpose

This standard defines the purpose of SQA processes as: to specify the activities and tasks that enable software suppliers to produce, collect, and validate evidence that forms the basis for a justified statement of confidence that the software product conforms to its established requirements.

SQA ensures that processes are established, managed, maintained, and applied on projects by skilled and qualified staff and that the activities and tasks performed are commensurate with product risk.

5.2 Organization of SQA process outcomes

The outcomes produced by SQA processes are organized into three groups:

a) SQA Process Implementation. A strategy for conducting SQA is developed. SQA activities are planned and executed. Evidence of SQA is produced and maintained (see 5.3).

b) Product Assurance. Adherence of products to the established requirements is evaluated. Problems and non-conformance are identified and recorded (see 5.4).

c) Process Assurance. Adherence of processes and activities to the applicable standards and procedures is verified. Effectiveness of processes is evaluated and improvements are suggested. Problems and non-conformance are identified and recorded (see 5.5).

The 16 SQA activities are organized into three groups as shown in Table 1, below:

Table 1—Organization of 16 SQA activities

Subclause	Title
5.3	SQA process implementation activities
5.3.1	Establish the SQA processes
5.3.2	Coordinate with related software processes
5.3.3	Document SQA planning
5.3.4	Execute the SQA Plan
5.3.5	Manage SQA records
5.3.6	Evaluate organizational independence and objectivity
5.4	Product assurance activities
5.4.2	Evaluate plans for conformance to contracts, standards, and regulations
5.4.3	Evaluate product for conformance to established requirements
5.4.4	Evaluate product for acceptability

Table 1—Organization of 16 SQA activities (continued)

Subclause	Title
5.4.5	Evaluate product life cycle support for conformance
5.4.6	Measure products
5.5	Process assurance activities
5.5.2	Evaluate life cycle processes and plans for conformance
5.5.3	Evaluate environments for conformance
5.5.4	Evaluate subcontractor processes for conformance
5.5.5	Measure processes
5.5.6	Assess staff skill and knowledge

NOTE—5.4.1 and 5.5.1 contain introductory material concerning product assurance and process assurance, respectively, and are not numbered in the 16 SQA activities.

As specified in 1.7, each of the activities in Clause 5 of this standard is described by the following information:

a) Reference to ISO/IEC/IEEE 12207:2008 clause. The text of the ISO/IEC/IEEE 12207:2008 subclause relevant to the activity is presented in a boxed figure, for example:

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.2.2** It shall be assured that software products and related documentation comply with the contract and adhere to the plans.

b) Purpose. Defines the activity's intention (e.g., "Determine the degree to which the software products and related documentation conform to established requirements.").

c) Outcomes. Specific observable results of the successful achievement of activity's purpose. An outcome may an information item (e.g., records, documents), a change in the state or an attribute of an information item, a change to a project constraint, or a change to an attribute (e.g., training, experience, capability) of a project team member. Information items are defined in ISO/IEC/IEEE 15289:2011. These outcomes are the basis for validated evidence to provide a justified statement of confidence in the quality of the software products. Outcomes are written as declarative sentences in the present tense (e.g., "Non-conformances are raised when software products do not conform to established software requirements.")

d) Tasks. Specific actions that are intended to contribute to the achievement of one or more of the stated outcomes of the activity. Using the definitions from ISO/IEC TR 24774:2010 [B41], a statement in the task description can be:

1) A required action,
2) A recommended action, or
3) A permissible action.

Task statements identify the role performing the task. The role is either the subject of the sentence or is in a clause that introduces a set of tasks. Task statements for required actions begin with an action verb and are written as declarative sentences in the present tense. Recommended and permissible action statements start

with a phrase or clause that qualifies the conditions under which the task may be performed. There is not necessarily a one-to-one relationship between tasks and outcomes (e.g., identify software products and related documentation required by the contract).

5.3 SQA process implementation activities

5.3.1 Establish the SQA processes

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclauses:

> **7.2.3.3.1.1** A quality assurance process suited 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.

> **7.2.3.3.4.1** Additional quality management activities may be assured in accordance with the clauses of ISO 9001.

5.3.1.1 Purpose

Define and establish documented SQA processes that exist separately from projects. When these SQA processes are applied to a project they enable the development of software that conforms to established requirements. These processes also provide projects with software quality measurements to help make cost, schedule, quality, and risk tradeoffs. The SQA processes define the SQA function's role, concepts, methods, procedures, and practices.

Effective SQA processes identify what activities to do, how to confirm the activities are performed, how to measure and track the processes, how to learn from measures to manage and improve the processes, and how to encourage using the processes to produce software products that conform to established requirements. SQA processes are continually improved based on objective measures and actual project results.

5.3.1.2 Outcomes

This activity shall produce the following outcomes, prior to executing the project:

- Management has established an SQA function’s role within the organization.
- Management has established organizational SQA processes that are independent of SQA processes established for individual projects.
- An organizational policy is established that defines and governs SQA roles and responsibilities.
- A method is established for overseeing the execution of SQA activities, tasks, and outcomes along with a method for providing feedback to the SQA function.
- A method is established to enable the SQA function within projects to learn from the experiences of current and previous projects and share lessons learned with other projects.
- People are assigned responsibility for performing the SQA role both within the organization as a whole and for a specific project in a manner that is organizationally independent of both project management and software development.

5.3.1.3 Tasks

To accomplish this activity, the organization shall perform the following tasks:

1) Define an organizational quality policy statement that defines and governs SQA roles and responsibilities.
2) The organizational quality policy statement may be included in an organization Quality Manual for those organizations that have an existing quality management system.
3) Establish an organizational quality policy statement that defines the SQA process as an organization level process independent of SQA processes established for specific projects.
4) Create and assign tasks to those responsible for SQA activities and for implementing the organizational quality policy statement.
5) Define a mechanism for providing management oversight of SQA activities and tasks.
6) Assign responsibility for project SQA activities to individuals who are organizationally independent of project management and software development.
7) Review the organizational quality policy and identify gaps and inconsistencies between those organizational quality policies and proposed SQA roles and responsibilities.
8) The SQA process should be established prior to the acquisition phase, when contracts are typically negotiated and agreements reached, so that the SQA function can assist in forming effective contracts.

NOTE—Table A.1 in Annex A shows the correlation of four outcomes as described in ISO/IEC/IEEE 12207:2008 and the clauses of this standard.

5.3.2 Coordinate with related software processes

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.1.2** The quality assurance process should be coordinated with the related Software Verification (subclause 7.2.4), Software Validation (subclause 7.2.5), Software Review (subclause 7.2.6), and Software Audit (subclause 7.2.7) Processes.

5.3.2.1 Purpose

Coordinate with the verification, validation, review, audit and other ISO/IEC/IEEE 12207:2008 processes that are relevant to the project to achieve project objectives. The SQA function works with project management to determine which of the other 39 processes defined in ISO/IEC/IEEE 12207:2008 should be coordinated with SQA activities.

The SQA function identifies project and organizational roles and responsibilities with respect to all of the following ISO/IEC/IEEE 12207:2008 processes and clauses that apply to the project:

- 6.1 Agreement Processes
- 6.3 Project Processes
- 6.4 Technical Processes
- 7.1 Software Implementation Processes
- 7.2 Software Support Processes

  - 7.2.4 Software Verification
  - 7.2.5 Software Validation
  - 7.2.6 Software Review
  - 7.2.7 Software Audit
- 7.3 Software Reuse Processes

At the organizational-level, the SQA function identifies organizational roles and responsibilities with respect to the following organizational project-enabling processes:

- 6.2.1 Life Cycle Model Management Process
- 6.2.5 Quality Management Process

NOTE—Figure A.1 in Annex A shows the relation of the SQA function to all of the ISO/IEC/IEEE 12207:2008 processes.

5.3.2.2 Outcomes

This activity shall produce the following outcomes:

- Redundant tasks and duplication of effort are eliminated or reduced.
- Project and organizational-level roles and responsibilities for processes defined in ISO/IEC/IEEE 12207:2008 that apply to the project are defined and documented.
- SQA activities are coordinated with the project and organizational-level roles and responsibilities.

5.3.2.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Collaborate with other project and organizational elements to define roles and responsibilities with respect to all of the ISO/IEC/IEEE 12207:2008 processes listed above that are applicable to the project.
2) Document the defined roles and responsibilities based on this collaboration in the SQA Plan (SQAP) (see 5.3.3).

5.3.3 Document SQA planning

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

7.2.3.3.1.3 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 Software Verification (subclause 7.2.4), Software Validation (subclause 7.2.5), Software Review (subclause 7.2.6), Software Audit (subclause 7.2.7), and Software Problem Resolution (subclause 7.2.8).

5.3.3.1 Purpose

Document SQA activities, tasks, and outcomes that are commensurate with the software risk for a specific project. Project-specific SQA activities and tasks are derived from the "Coordinate with related software processes" activity described in 5.3.2 as well as from the organizational quality management plan. the SQA function's planning includes adapting the generic SQA processes to the specific needs of the project in a manner commensurate with product risk. The Software Quality Assurance Plan (SQAP) documents the results of SQA planning.

5.3.3.2 Outcomes

This activity shall produce the following outcomes:

— An SQAP is prepared that identifies SQA activities and tasks for the project commensurate with the software product risks established for the project.

— The SQAP identifies responsibilities that result from coordinating with other project and organizational units, as described in 5.3.2.

— The SQAP addresses all relevant clauses of this standard.

— The SQAP may reference organization-level SQA material or may describe any tailoring of organization level SQA material to establish the project's SQA processes.

— A method for periodically presenting the SQA function's status to project management and organization quality management is defined.

5.3.3.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Work with all stakeholders to determine which subclauses of this standard are relevant for the project.

2) When the applicable subclauses are identified, use this standard and Annex C to help prepare an SQAP that is appropriate for the project, addresses the needs of all stakeholders, and is commensurate with product risk.

3) Use outcomes of 5.3.2 to help prepare the SQAP in a manner that reflects the coordination with other processes areas.

4) Review and update the SQAP as the project evolves.

5) Present status information to Management in the agreed manner.

6) Estimate the SQA function's resources (including effort, schedule, people, required skills, tools, and equipment) needed to perform the SQA activities, tasks, and outcomes.

7) Analyze product risks, standards, and assumptions that could impact quality and identify specific SQA activities, tasks, and outcomes that could help determine whether those risks are effectively mitigated.

8) Analyze the project and adapt SQA activities accordingly so they are commensurate with risk.

9) Define specific measurements for assessing project, software quality, and the SQA function's performance against project and organization quality management objectives.

10) Identify and track project changes that require further SQA function planning, including changes to: requirements, resources, schedules, project scope, priorities, and product risk.

11) If process areas or activities are not adequately addressed by the organizational quality management function, or if there is no organizational quality management function, then these process areas may be in the SQAP.

12) Address all of the topics listed in the normative SQA plan outline shown in Figure 5. Every section in the plan is to be included. The topics of the SQAP are normative but the section names and order are informative. If a section in the outline is not applicable on a given project, a placeholder for that section may be included along with a justification for why the topic is not applicable.

13) If there is an organizational SQAP, or if organizational processes will be used directly, the SQAP may refer to those items. Additional sections and appendices beyond those listed below may be added.

NOTE—Refer to Annex C for additional guidance in preparing an SQAP.

```
1  Purpose and scope
2  Definitions and acronyms
3  Reference documents
4  SQA plan overview
      4.1  Organization and independence
      4.2  Software product risk
      4.3  Tools
      4.4  Standards, practices, and conventions
      4.5  Effort, resources, and schedule
5  Activities, outcomes, and tasks
      5.1  Product assurance
           5.1.1 Evaluate plans for conformance
           5.1.2 Evaluate product for conformance
           5.1.3 Evaluate product for acceptability
           5.1.4 Evaluate product life cycle support for conformance
           5.1.5 Measure products
      5.2  Process assurance
           5.2.1 Evaluate life cycle processes for conformance
           5.2.2 Evaluate environments for conformance
           5.2.3 Evaluate subcontractor processes for conformance
           5.2.4 Measure processes
           5.2.5 Assess staff skill and knowledge
6  Additional considerations
      6.1  Contract review
      6.2  Quality measurement
      6.3  Waivers and deviations
      6.4  Task repetition
      6.5  Risks to performing SQA
      6.6  Communications strategy
      6.7  Non-conformance process
7  SQA records
      7.1  Analyze, identify, collect, file, maintain and dispose
      7.2  Availability of records
```

Figure 5—SQA Plan Outline

NOTE—Specific content for quality records and reports are defined in ISO/IEC/IEEE 15289:2011.

5.3.4 Execute the SQA Plan

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.1.4** Scheduled and on-going quality assurance activities and tasks shall be executed. When problems or non-conformances with contract requirements are detected, they shall be documented and serve as input to the Problem Resolution Process (subclause 7.2.8). Records of these activities and tasks, their execution, problems, and problem resolutions shall be prepared and maintained.

5.3.4.1 Purpose

Execute the SQAP in coordination with the project manager, the project team, and organizational quality management.

5.3.4.2 Outcomes

This activity shall produce the following outcomes:

— The SQA activities and tasks defined in the SQAP are performed and repeated as needed.

— Product and process non-conformances are raised by the SQA function when actual outcomes do not agree with expectations.

— The SQAP is revised as needed to reflect project changes.

— SQA reports and records are created, maintained, and used to evaluate software quality.

— SQA reports and records are shared with other project stakeholders.

5.3.4.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Execute the activities and tasks defined in the SQAP, based on project schedules.
2) Create the outcomes identified in the SQA Plan.
3) Revise the SQAP in response to project changes.
4) Raise non-conformances when actual outcomes do not agree with expectations.

5.3.5 Manage SQA records

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.1.5** Records of quality assurance activities and tasks shall be made available to the acquirer as specified in the contract.

5.3.5.1 Purpose

Create records of SQA activities, outcomes, and tasks; manage and control these records; make these records available to project stakeholders.

5.3.5.2 Outcomes

This activity shall produce the following outcomes:

— Records related to SQA activities, outcomes, and tasks are created.

— Records are maintained and stored in accordance with appropriate organizational, regulatory, and project plan requirements.

— Records are made available to project stakeholders as specified by the contract and the SQAP.

5.3.5.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Create records as required by the SQAP. These records capture findings of SQA activities and tasks and provide evidence that SQA activities and tasks were performed.
2) Maintain records according to trustworthiness, security, and privacy requirements.
3) Identify records of quality assurance activities as accessible, deliverable, or internal use only to avoid contract non-compliance or inadvertent transfer of intellectual property.

4) Maintain the integrity of the SQA function's records through a document control system to prevent their modification or inadvertent removal and release.

5) Supply specific records to authorized stakeholders defined in the contract. Records are made available subject to confidentiality and other constraints. The contract specifies what the acquirer will receive from the supplier; the SQAP specifies which SQA function records the acquirer's internal organizations will receive.

5.3.6 Evaluate organizational independence and objectivity

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.1.6** 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.

5.3.6.1 Purpose

Determine whether those persons responsible for performing the SQA function have a position within the organization that provides an unimpeded communication mechanism with organizational management. Also, determine whether those persons have the resources and authority to make objective evaluations, initiate, effect, and verify problem resolutions.

5.3.6.2 Outcomes

This activity shall produce the following outcomes:

— The organizational objectivity of those responsible for the SQA function is defined and confirmed by the organization.

— Those responsible for the SQA function have adequate resources and authority to permit objective evaluations.

— Those responsible for the SQA function have adequate resources and authority to identify problems, ensure problem resolutions are implemented, and verify the effectiveness of problem resolutions.

— Those responsible for the SQA function are independent of the development organization. An independent SQA is defined by three parameters: technical independence, managerial independence, and financial independence.

— Those responsible for the SQA effort use personnel who are not involved in the development of the system or its elements (technical independence).

— Those responsible for the SQA effort are vested in an organization separate from the development and program management organizations (managerial independence).

— Those responsible for control of the SQA budget are vested in an organization independent of the development organization (financial independence).

5.3.6.3 Tasks

To accomplish this activity, the following tasks shall be performed:

1) The organization identifies the individuals and organizational entities responsible for the SQA function of the product and project

2) The organization, including SQA, determines whether those responsible for the SQA function have the organizational freedom to perform that activity in a manner that permits objective evaluations

5.4 Product assurance activities

5.4.1 Defining product assurance

An important aspect of SQA is the establishment of confidence in the quality of the software products produced by the project. The products are the software and related documentation. Related documentation may include all those documents associated with the development, operation, support, maintenance, and retirement of the software, including installation and administration. A product may also be a software service provided to the acquirer.

The outcomes of the product assurance activities provides evidence that the software services, products, and any related documentation are identified in and comply with the contract and any non-conformances are identified and addressed. Product Assurance comprises these activities:

— Evaluate plans for conformance

— Evaluate product for conformance

— Evaluate product for acceptability

— Evaluate product life cycle support for conformance

— Measure products

The SQA function confirms that software products are in conformance with established requirements. That is, SQA enables software suppliers to produce, collect, and validate evidence that forms the basis for a justified statement of confidence that the software product conforms to its established requirements. For example, Product Assurance activities may include SQA personnel participating in project technical reviews, software development document reviews, and software testing.

5.4.2 Evaluate plans for conformance to contracts, standards, and regulations

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.2.1** 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.

5.4.2.1 Purpose

Determine whether plans required by the contract are documented. Determine whether plans conform to the contract. Determine whether plans required by the contract comply with applicable laws, regulations, and rules. Determine whether all the plans taken as a whole are consistent with each other.

5.4.2.2 Outcomes

This activity shall produce the following outcomes:

— Plans required by the contract are identified and evaluated.

— Contractually imposed rules, regulations, and laws are identified and evaluated.

— Non-conformances are raised when project plans do not conform to the applicable, established process requirements.

5.4.2.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Analyze the contract to identify the plans required by the contract.
2) Evaluate (validate, verify, or review) project plans for conformance to the contract, determine whether plans conform to the established process requirements.
3) Raise non-conformances when actual results do not agree with expectations.
4) Evaluate (validate, verify, or review) plans for mutual consistency.

5.4.3 Evaluate product for conformance to established requirements

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.2.2** It shall be assured that software products and related documentation comply with the contract and adhere to the plans.

5.4.3.1 Purpose

Determine the degree to which the software products and related documentation conform to established requirements.

5.4.3.2 Outcomes

This activity shall produce the following outcomes:

— Software products and related documentation required by the contract are identified.

— Non-conformances are raised when software products do not conform to established software requirements.

— Non-conformances are raised when related documentation does not conform to established software requirements.

5.4.3.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Identify software products and related documentation required by the contract.
2) Identify the requirements allocated to the software products and related documentation. Evaluate (validate, verify, or review) the results of the allocation process.
3) Evaluate (validate, verify, or review) software products for conformance against the established software requirements.

4) Evaluate (validate, verify, or review) related documentation for conformance against the established software requirements.

5.4.4 Evaluate product for acceptability

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.2.3** 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.

5.4.4.1 Purpose

Prior to delivery, determine the degree of confidence the supplier has that the established requirements are satisfied and that the software products and related documentation will be acceptable to the acquirer. Collect measurement data sufficient to support these satisfaction and acceptability decisions. A contract may provide that the acquirer, prior to delivery, determine whether software products are acceptable.

5.4.4.2 Outcomes

This activity shall produce the following outcomes:

- The supplier's understanding of conditions for product acceptance is documented.
- Prior to delivery of the software products, the products' conformance to the supplier's understanding of conditions for product acceptance is confirmed.
- Non-conformances are raised when software products do not conform to established software requirements.
- Prior to delivery of the software products, the acquirer may acknowledge that the software products satisfy contractual obligations and are acceptable.

5.4.4.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Identify criteria for product acceptance. These conditions may be derived from the contract, project plans, documentation, SQA reports, and other sources.
2) Determine whether the product conforms to the contract using techniques that include: reviewing, auditing, testing, or evaluating the results of these techniques.
3) Determine whether the product conforms to the documented acceptance requirements.
4) Determine whether acquirer has the means to determine the criteria to accept the product.

5.4.5 Evaluate product life cycle support for conformance

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.3.4** 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.

5.4.5.1 Purpose

Determine whether the product support requirements identified in the project plans are consistent with the contract and clearly identify the responsibilities of both the product delivery organization and the acquirer.

5.4.5.2 Outcomes

This activity shall produce the following outcomes:

- The product life cycle support required by the contract is identified.
- Product life cycle support plans conform to the contract.
- Non-conformances are raised, by the SQA function, when support plans do not conform to the contract.
- The acquirer's contractual obligations for support and cooperation from the supplier are met.

5.4.5.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Analyze the contract to identify the required level of customer support.
2) Review customer support plans and determine whether the level of customer support is consistent with contractual support requirements.
3) Identify SQA requirements regarding software customer support and include them in the SQAP.
4) Monitor software customer support activities and raise non-conformances when these activities are not performed as defined in the plan.
5) Measure and evaluate, on a regular basis, the level of support and cooperation with respect to product support plan and identify issues.

5.4.6 Measure products

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.3.5** It should be assured that software product and process measurements are in accordance with established standards and procedures.

5.4.6.1 Purpose

Determine whether product measurements demonstrate the quality of the products and conform to standards and procedures established by the project.

5.4.6.2 Outcomes

This activity shall produce the following outcomes:

- Software product measurements conform to project's processes and plans, and conform to standards and procedures established by the project or organization.

— Software product measurements accurately represent software product quality.

— Software product measurements are shared with project stakeholders.

— Software product measurements are performed on software products developed by the supplier as well as all of the supplier's subcontractors.

— Software product measurements are presented to management for review and potential corrective and preventive action.

— Non-conformances are raised when required measurement activities are not performed as defined in project plans.

NOTE—Additional information about measurement can be found in IEEE 15939-2008 [B28].

5.4.6.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Identify the standards and procedures established by the project or organization.
2) Determine whether proposed product measurements are consistent with standards and procedures established by the project.
3) Determine whether the proposed product measurements are representative of product quality attributes.
4) Analyze product measurement results to identify gaps and recommend improvements to close gaps between measurements and expectations.
5) Evaluate product measurement results to determine whether improvements implemented as a result of product quality measurements are effective.
6) Analyze product measurement procedures to confirm they are sufficient to satisfy the measurement requirements defined in project's processes and plans.
7) Perform Task 1 through Task 6, above, for software products developed by all subcontractors.

5.5 Process assurance activities

5.5.1 Defining process assurance

The outcomes of SQA process assurance activities make certain that the processes used to develop, install, operate, and maintain software conform to the contract, comply with any imposed regulations, and are adequate, efficient, and effective. "Adequate, efficient, and effective" means that the processes can, and do, consistently produce software that conforms to established requirements and organizational considerations. Depending on life cycle model processes, these outcomes are reported to project management, quality management, and risk management.

Process Assurance comprises these activities:

— Evaluate life cycle processes for conformance

— Evaluate environments for conformance

— Evaluate subcontractor processes for conformance

— Measure processes

— Assess staff skill and knowledge

5.5.2 Evaluate life cycle processes and plans for conformance

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclauses:

> **7.2.3.3.3.1** It shall be assured that those software life cycle processes (supply, development, operation, maintenance, and support processes including quality assurance) employed for the project comply with the contract and adhere to the plans.

> **7.2.3.3.2.1** 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.

5.5.2.1 Purpose

Determine whether the project life cycle processes and plans conform to the established process requirements. Determine whether execution of software activities conform to the project's processes and plans.

5.5.2.2 Outcomes

This activity shall produce the following outcomes:

- Documented software life cycle processes and plans are evaluated for conformance to the established process requirements.
- Project life cycle processes and plans conform to the established process requirements.
- Non-conformances are raised when software life cycle processes and plans do not conform to the established process requirements.
- Non-conformances are raised when software life cycle processes and plans are not adequate, efficient, or effective.
- Non-conformances are raised when execution of project activities does not conform to software life cycle processes and plans.
- Subcontractor software life cycle processes and plans conform to the process requirements passed down from the acquirer.

5.5.2.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Identify applicable process requirements that may affect the selection of a software life cycle process.
2) Determine whether the defined software life cycle processes selected by the project team are appropriate, given the product risk.
3) Review project plans and determine whether plans are appropriate to meet the contract based on the chosen software life cycle processes and relevant contractual obligations.
4) Audit software development activities periodically to determine consistency with defined software life cycle processes.
5) Audit project team periodically to determine conformance to defined project plans.
6) Perform Task 1 through Task 5, above, for subcontractor's software development life cycle.

5.5.3 Evaluate environments for conformance

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.3.2** It shall be assured that the internal software engineering practices, development environment, test environment, and libraries comply with the contract.

5.5.3.1 Purpose

Determine whether software engineering environments (SEE) and software test environments (STE) conform to project processes and plans.

5.5.3.2 Outcomes

This activity shall produce the following outcomes:

- Software engineering environments are consistent with project plans.
- Software test environments are consistent with project plans.
- Non-conformances are raised when software engineering environments do not conform to project plans.
- Non-conformances are raised when software test environments do not conform to project plans.

5.5.3.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Review the software engineering environments used by the project team to determine whether they conform to the contract.
2) Review the software engineering libraries used by the project team to determine whether they conform to the contract and project plans.
3) Review the software test environments used by the project team to determine whether they conform to the contract and project plans.

NOTE—SEEs and STEs often include software development tools. Such tools may require validation or evaluation as determined by the contract or industry regulations.

5.5.4 Evaluate subcontractor processes for conformance

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.3.3** 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.

5.5.4.1 Purpose

Determine whether the subcontractor’s software processes conform to process requirements that have been allocated from the acquirer.

5.5.4.2 Outcomes

This activity shall produce the following outcomes:

- Process requirements passed down to subcontractors are identified.
- Non-conformances are raised when subcontractor software processes do not conform to process requirements passed down from the supplier.
- Subcontractor software processes conform to the established process requirements.
- Non-conformances are raised when a subcontractor's software processes do not conform to the established process requirements.

5.5.4.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Identify process requirements from the supplier and, if appropriate, from the acquirer that are allocated to the subcontractor.
2) Determine whether subcontractor software processes are defined.
3) Determine whether subcontractor software processes conform to project processes and plans.

5.5.5 Measure processes

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.3.5** It should be assured that software product and process measurements are in accordance with established standards and procedures.

5.5.5.1 Purpose

Determine whether the process measurements support effective process management and conform to the project's processes and plans and conform to established standards and procedures.

5.5.5.2 Outcomes

This activity shall produce the following outcomes:

- Software process measurement activities conform to project's processes and plans.
- Software process measurement activities conform to standards and procedures established for the project or organization.
- Reports documenting software measurement results are prepared and reviewed with project stakeholders and management.
- Non-conformances are raised when software measurement activities do not conform to project's processes and plans.
- Software process measurements accurately represent software process quality.
- Software process measurements are shared with project stakeholders.
- Software process measurements are performed on all of the project's and subcontractors' processes.

NOTE—Additional information about measurement can be found in IEEE 15939-2008 [B28].

5.5.5.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Identify the standards and procedures established for the project or organization.
2) Evaluate whether the process measurement activities are being executed in conformance to the project's processes and plans.
3) Evaluate whether the process measurement activities conform to standards and procedures established by the project and the organization.
4) Analyze process measurement procedures to confirm they are sufficient to satisfy the measurement requirements defined in project's processes and plans.
5) Review the process measurement plan to determine whether the SQA function's measurement needs are addressed.
6) Analyze process measurement procedures to confirm they are sufficient to satisfy the measurement requirements defined in project plans and the contract.
7) For each subcontractor, perform Task 1 through Task 6, above, for each of their software processes.

5.5.6 Assess staff skill and knowledge

This subclause addresses the following ISO/IEC/IEEE 12207:2008 subclause:

> **7.2.3.3.3.6** 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.

5.5.6.1 Purpose

Determine whether staff assigned to the project have the required knowledge, skill, competencies, and abilities to perform the tasks required for their roles.

5.5.6.2 Outcomes

This activity shall produce the following outcomes:

- Gaps in project staff education and training are identified.
- Education and training plans include knowledge transfer activities to propagate training, awareness, competence, and proficiency across resources.
- Education and training plans to close the identified gaps are documented.
- Non-conformances are raised when education and training results do not conform to plans.
- Education and training assessments of new project team members are conducted.
- Education and training plans are monitored and tracked.
- The required skill levels of every project role are defined in appropriate project plans.
- Skill development records or training records that demonstrate skill competence are complete and available.

5.5.6.3 Tasks

To accomplish this activity, the SQA function shall perform the following tasks:

1) Audit the skill and knowledge needs of the project and compare to skill and knowledge of the organization's staff and identify any gaps.
2) Determine whether skills and knowledge development plans are in place and are being executed to fill the identified gaps and to accomplish required knowledge transfer.
3) Determine whether the skills and knowledge needs of the project have changed.
4) Determine whether new team members are assessed and that all prepared individual skill and knowledge development plans are being monitored and tracked to completion.
5) Review personnel training records on a regular basis.

Annex A

(informative)

Mapping between 7.2.3 of ISO/IEC/IEEE 12207:2008 and IEEE Std 730-2014

There are four outcomes identified in 7.2.3 of ISO/IEC/IEEE 12207:2008 that the software quality assurance (SQA) function is to provide. The four outcomes are accomplished by activities in IEEE Std 730-2014. Table A.1 below shows how these four outcomes map to clauses of IEEE Std 730-2014.

Table A.1—Mapping between 7.2.3 of ISO/IEC/IEEE 12207:2008 and IEEE Std 730-2014

ISO/IEC/IEEE 12207:2008 Software Quality Assurance Outcomes	**IEEE Std 730-2014 subclauses that address each outcome**
A. A strategy for conducting quality assurance is developed.	5.3.1 Establish the SQA processes 5.3.2 Coordinate with related software processes 5.3.3 Document SQA planning 5.3.6 Evaluate organizational independence and objectivity
B. Evidence of software quality assurance is produced and maintained.	5.3.4 Execute the SQA Plan 5.3.5 Manage SQA records
C. Problems and/or non-conformance with requirements are identified and recorded.	5.4.2 Evaluate plans for conformance to contracts, standards, and regulations 5.4.3 Evaluate product for conformance to established requirements 5.4.4 Evaluate product for acceptability 5.4.5 Evaluate product life cycle support for conformance 5.4.6 Measure products 5.5.2 Evaluate life cycle processes and plans for conformance 5.5.3 Evaluate environments for conformance 5.5.4 Evaluate subcontractor processes for conformance 5.5.5 Measure processes 5.5.6 Assess staff skill and knowledge
D. Adherence of products, processes and activities to the applicable standards, procedures and requirements are verified.	5.4.6 Measure products 5.5.5 Measure processes 5.3.5 Manage SQA records

The 16 SQA activities identified in Clause 5 of this standard represent activities that span the entire system/software development life cycle. The role of SQA on a system or software development project is to be viewed as broadly as possible at first and then narrowed based on several factors, including but not limited to the following:

— Contractual agreement between acquirer and supplier.

— Existence of a supplier organizational Quality Management function.

— Product risk associated with the proposed system or software .

SQA adds value and plays an important role in all of the ISO/IEC/IEEE 12207:2008 process areas.

Figure A.1 (shown as a series of three images), below, illustrates the relationship between each of the ISO/IEC/IEEE 12207:2008 process areas and the SQA function as defined in this standard. It is expected that each software or system development project team will tailor the SQA role using the criteria above along with other relevant criteria.

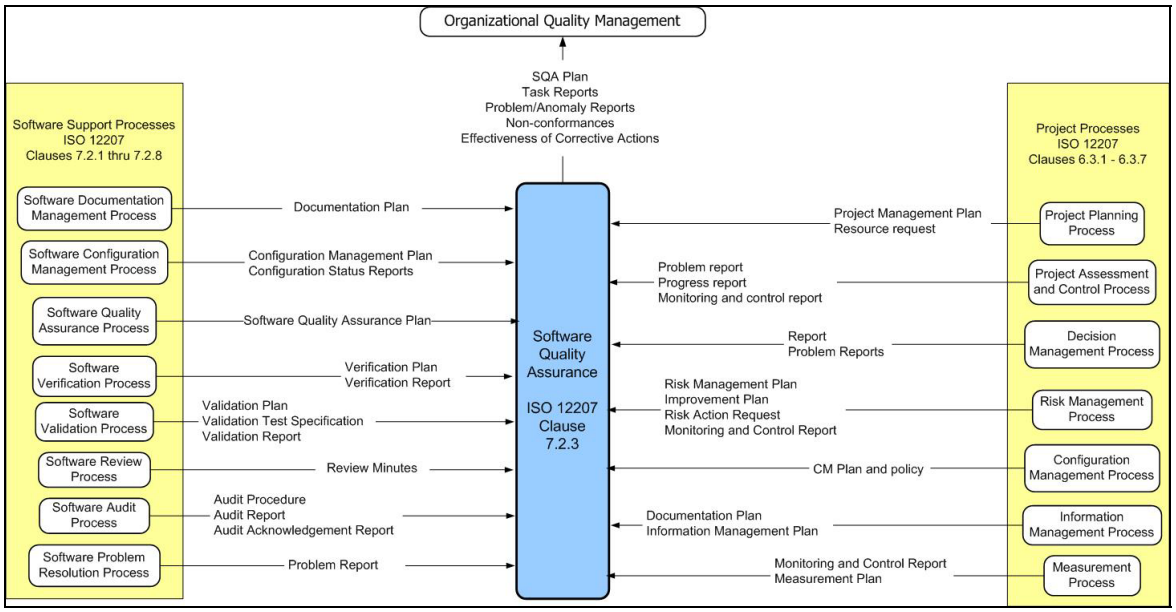

Figure A.1—Relationship of ISO/IEC/IEEE 12207:2008 process and the SQA function

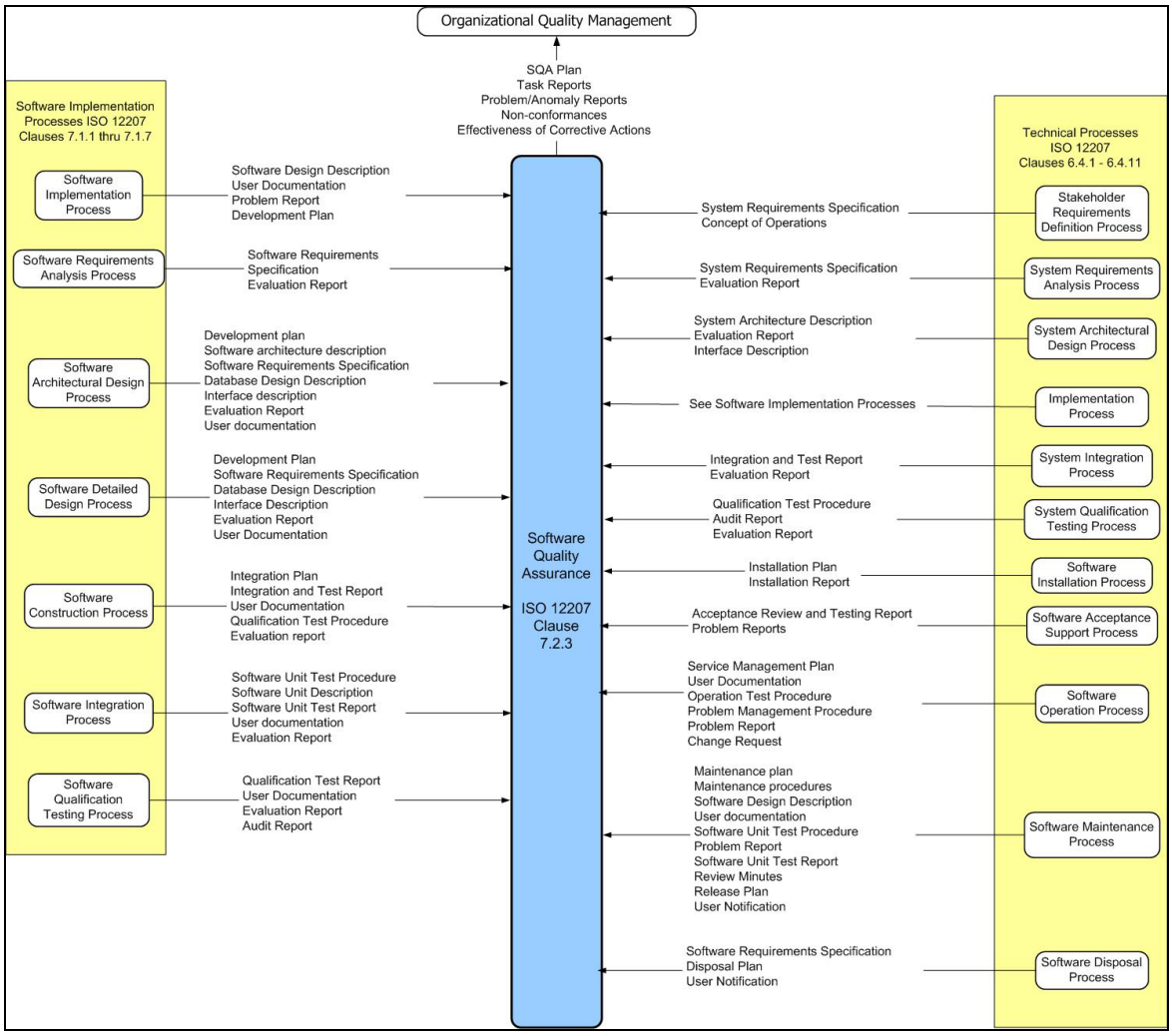

Figure A.1—Relationship of ISO/IEC/IEEE 12207:2008 process and the SQA function *(continued)*

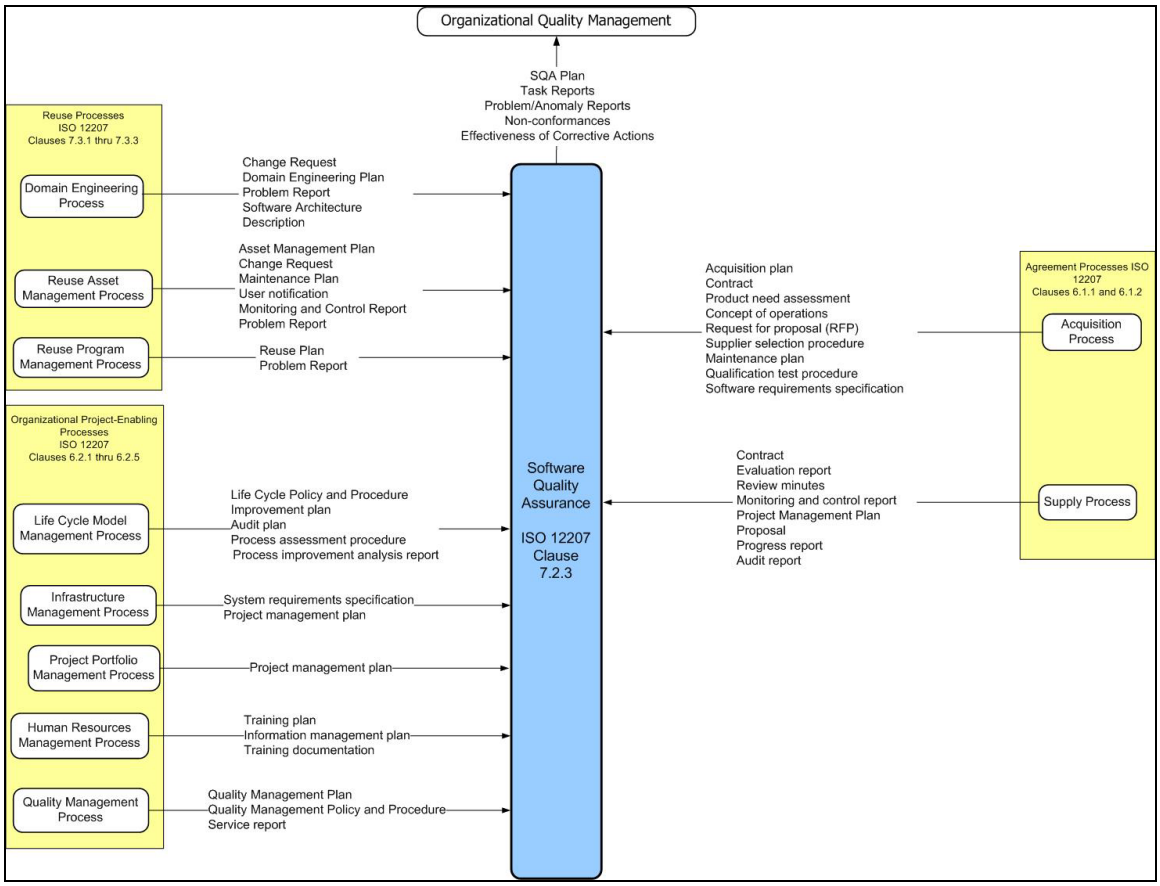

Figure A.1—Relationship of ISO/IEC/IEEE 12207:2008 process and the SQA function *(continued)*

Annex B

(informative)

Mapping between SQA Plan outlines contained in IEEE Std 730-2002 and IEEE Std 730-2014

This annex provides a cross reference between the Software Quality Assurance Plan (SQAP) outlines given in IEEE Std 730-2002 and IEEE Std 730-2014. Table B.1 presents the SQAP outline from IEEE Std 730-2002 and shows where the information would be found in the IEEE Std 730-2014 SQAP outline. Table B.2 presents the SQAP outline from IEEE Std 730-2014 and shows where the information would be found in the IEEE Std 730-2002 SQAP outline. This annex can be used as an aid to organizations transitioning from an earlier version of this standard to the current version.

Table B.1—SQA plan outline cross-reference IEEE Std 730-2002 and IEEE Std 730-2014

IEEE Std 730-2002 SQAP Outline **Section numbers refer to sections in the plan outline in Clause 4.**	**IEEE Std 730-2014 SQAP Outline** **Section numbers refer to sections in the plan outline in Figure 5.**
1. Purpose	1. Purpose and scope
2. Reference documents	3. Reference documents
3. Management	4. SQA plan overview
3.1 Organization	4.1 Organization and independence
3.2 Tasks	5. Activities, outcomes, and tasks
3.3 Roles and Responsibilities	4.1 Organization and independence
3.4 Quality Assurance Estimated Resources	4.5 Effort, resources, and schedule
4. Documentation 4.1 Purpose 4.2 Minimum Documentation Requirements 4.2.1 Software requirements description (SRD) 4.2.2 Software design description (SDD) 4.2.3 Verification and validation plans 4.2.4 Verification results report and validation results report 4.2.5 User documentation 4.2.6 Software configuration management plan (SCMP) 4.3 Other documentation	5. Activities, outcomes, and tasks
5. Standards, practices, conventions, and metrics 5.1 Purpose 5.2 Content	4.4 Standards, practices, and conventions

Table B.1—SQA plan outline cross-reference IEEE Std 730-2002 and IEEE Std 730-2014 (continued)

IEEE Std 730-2002 SQAP Outline **Section numbers refer to sections in the plan outline in Clause 4.**	**IEEE Std 730-2014 SQAP Outline** **Section numbers refer to sections in the plan outline in Figure 5.**
6. Software reviews 6.1 Purpose 6.2 Minimum requirements 6.2.1 Software specifications review (SSR) 6.2.2 Architecture design review (ADR) 6.2.3 Detailed design review (DDR) 6.2.4 Verification and validation plan review 6.2.5 Functional audit 6.2.6 Physical audit 6.2.7 In-process audits 6.2.8 Managerial reviews 6.2.9 Software configuration management plan review (SCMPR) 6.2.10 Post-implementation review 6.3 Other reviews and audits	5. Activities, outcomes, and tasks
7. Test	5. Activities, outcomes, and tasks
8. Problem reporting and corrective action	5. Activities, outcomes, and tasks
9. Tools, techniques, and methodologies	4.3 Tools
10. Media control	5.1 Product assurance
11. Supplier control	6.1 Contract review
12. Records collection, maintenance, and retention	7. SQA Records
13. Training	4.5 Effort, resources, and schedule
14. Risk management	4.2 Software product risk
15. Glossary	2. Definitions and acronyms
16. SQAP change procedure and history	(Intended to be included in the SQAP but not identified as a separate section in this standard.)

Table B.2—SQA plan outline cross-reference IEEE Std 730-2014 and IEEE Std 730-2002

IEEE Std 730-2014 SQAP Outline **Section numbers refer to sections in the plan outline in Figure 5.**	**IEEE Std 730-2002 SQAP Outline** **Section numbers refer to sections in the plan outline in Clause 4.**
1. Purpose and scope	1. Purpose
2. Definitions and acronyms	15. Glossary
3. Reference documents	2. Reference documents
4. SQA plan overview	3. Management
4.1 Organization and independence	3.1 Organization 3.3 Roles and Responsibilities
4.2 Software product risk	14. Risk management
4.3 Tools	9. Tools, techniques, and methodologies
4.4 Standards, practices, and conventions	5. Standards, practices, conventions, and metrics
4.5 Effort, resources, and schedule	3.4 Quality Assurance Estimated Resources
5. Activities, outcomes, and tasks	
5.1 Product assurance	10. Media control
5.2 Process assurance	
6 Additional considerations	
6.1 Contract review	11. Supplier control
6.2 Quality measurement	
6.3 Waivers and deviations	
6.4 Task repetition	
6.5 Risk to performing SQA	
6.6 Communications strategy	
6.7 Non-conformance process	8. Problem Reporting and Corrective Action
7 SQA records	12. Records collection, maintenance, and retention
7.1 Analyze, identify, collect, file, maintain and dispose	12. Records collection, maintenance, and retention
7.2 Availability of records	12. Records collection, maintenance, and retention

Annex C

(informative)

Guidance for creating Software Quality Assurance Plans

C.1 Introduction

This annex provides guidance in creating Software Quality Assurance Plans (SQAPs). This annex can help identify appropriate Software quality assurance (SQA) activities and tasks when preparing a project SQAP. This annex can also be used by the SQA team as the project progresses.

The information in this annex is presented in the form of questions. Some questions can be used in the planning phase to help create an appropriate SQAP, and other questions can be used to help the SQA team when executing the SQAP. Together, these questions are used to encourage discussion between the SQA team, the project team, and the organization as a whole.

As described in this standard, SQA activities are planned and executed in a manner that is commensurate with product risk—the higher the product risk, the greater the breadth and depth of SQA activities. The questions included in this annex are intended to prompt further questions and deeper probing for those projects where product risk is high. Similarly, these questions can also be used to tailor the SQA Plan for those projects where product risk is lower. Refer to 1.5 of this standard for more information on conformance and tailoring.

Asking appropriate questions both when planning the SQA project activities and when executing SQA project activities can help do the following:

- Promote constructive dialogue within the project team as well as between supplier and acquirer.
- Understand and interpret the requirements of this standard in a manner that is more consistent with the needs of the supplier and the acquirer.
- Prepare a more complete and appropriate SQAP as a result of constructive dialogue.
- Prepare an organization for a compliance audit in situations in which compliance with this standard is mandatory.

Organizations use these questions as a guide and supplement them to meet their own needs as well the needs of acquirers. When appropriate, additional questions are used to help ensure that the breadth and depth of SQA tasks and activities is in fact commensurate with product risk.

Table C.1 shows the correspondence between subclauses 5.3, 5.4, and 5.5 of this standard to the sections of the SQA Plan outline, shown in Figure 5 of this standard.

Table C.1—Mapping between ISO/IEC/IEEE 12207:2008 and IEEE Std 730-2014

SQA Activities defined in this standard	Section of the SQA Plan outline where these activities are addressed
5.3 SQA process implementation activities	
5.3.1 Establish the SQA processes	*Not covered in SQA Plan since this activity occurs before project starts*
5.3.2 Coordinate with related software processes	4.1 Organization and Independence
5.3.3 Document SQA planning	*Writing the SQA Plan*
5.3.4 Execute the SQA Plan	*Executing the SQA Plan*
5.3.5 Manage SQA records	7. SQA Records
5.3.6 Evaluate organizational independence and objectivity	4.1 Organization and Independence
5.4 Product assurance activities	
5.4.1 Defining product assurance	*Not an activity*
5.4.2 Evaluate plans for conformance to contracts, standards, and regulations	5.1.1 Evaluate Plans for Conformance
5.4.3 Evaluate product for conformance to established requirements	5.1.2 Evaluate Product for Conformance
5.4.4 Evaluate product for acceptability	5.1.3 Evaluate Plans for Acceptability
5.4.5 Evaluate product life cycle support for conformance	5.1.4 Evaluate Product Life Cycle Support for Conformance
5.4.6 Measure products	5.1.5 Measure Products
5.5 Process assurance activities	
5.5.1 Defining process assurance	*Not an activity*
5.5.2 Evaluate life cycle processes and plans for conformance	5.2.1 Evaluate Life cycle Processes for Conformance
5.5.3 Evaluate environments for conformance	5.2.2 Evaluate Environments for Conformance
5.5.4 Evaluate subcontractor processes for conformance	5.2.3 Evaluate Subcontractor Processes for Conformance
5.5.5 Measure processes	5.2.4 Measure Processes
5.5.6 Assess staff skill and knowledge	5.2.5 Assess Staff Skill and Knowledge

Table C.2 shows the correspondence between the sections of the SQA Plan (as shown in Figure 5 of this standard) and the activities identified in this standard.

Table C.2—Mapping between SQA Plan and IEEE Std 730-2014

SQA Plan Section	SQA Activities defined in this standard
1 Purpose and scope	*Introductory information – not covered in this standard.*
2 Definitions and acronyms	
3 Reference documents	
4 SQA plan overview	
4.1 Organization and independence	5.3.6 Evaluate organizational independence and objectivity
4.2 Software product risk	5.3.3 Document SQA planning
4.3 Tools	5.5.3 Evaluate environments for conformance
4.4 Standards, practices, and conventions	5.5.2 Evaluate life cycle processes and plans for conformance
4.5 Effort, resources, and schedule	5.3.3 Document SQA planning
5 Activities, outcomes and tasks	
5.1 Product assurance	
5.1.1 Evaluate plans for conformance	5.4.2 Evaluate plans for conformance to contracts, standards, and regulations
5.1.2 Evaluate product for conformance	5.4.3 Evaluate product for conformance to established requirements
5.1.3 Evaluate product for acceptability	5.4.4 Evaluate product for acceptability
5.1.4 Evaluate product life cycle support for conformance	5.4.5 Evaluate product life cycle support for conformance
5.1.5 Measure products	5.4.6 Measure products
5.2 Process assurance	
5.2.1 Evaluate life cycle processes for conformance	5.5.2 Evaluate life cycle processes and plans for conformance
5.2.2 Evaluate environments for conformance	5.5.3 Evaluate environments for conformance
5.2.3 Evaluate subcontractor processes for conformance	5.5.4 Evaluate subcontractor processes for conformance
5.2.4 Measure processes	5.5.5 Measure processes
5.2.5 Assess staff skill and knowledge	5.5.6 Assess staff skill and knowledge
6 Additional considerations	
6.1 Contract review	*Not explicitly defined in this standard*
6.2 Quality measurement	5.4.6 Measure products 5.5.5 Measure processes
6.3 Waivers and deviations	*Not explicitly defined in this standard*
6.4 Task repetition	*Not explicitly defined in this standard*
6.5 Risks to performing SQA	*Not explicitly defined in this standard*
6.6 Communications strategy	*Not explicitly defined in this standard*
6.7 Non-conformance process	*Not explicitly defined in this standard*

Table C.2—Mapping between SQA Plan and IEEE Std 730-2014 (continued)

SQA Plan Section	SQA Activities defined in this standard
7 SQA records	
7.1 Analyze, identify, collect, file, maintain and dispose	5.3.5 Manage SQA records
7.2 Availability of records	5.3.5 Manage SQA records

C.2 Organization of this annex

This annex presents the SQAP Outline as shown in Figure 5 of this standard. The text included in the outline below is intended to help users of this standard understand the intent and content of each section of the SQAP when preparing an SQAP for a project.

For each section of the SQA Plan Outline, one or two tables are included. One table contains questions intended to help when planning SQA activities, and the other table has questions that can be raised as the project progresses.

The information included in the SQA Outline shown in Figure 5 is normative and is to be included in all SQA plans. If a particular section is not applicable to a project, a placeholder is included with a justification for why the section is not applicable. Additional sections beyond those identified in Figure 5 can be included in the SQAP.

Required outcomes associated with performing SQA activities and tasks are identified in Clause 5 of this standard. These outcomes are addressed in the appropriate sections of the SQAP.

In most cases, suggested inputs (information items) are included that may be helpful when preparing and executing the SQAP.

C.3 Guidance

The following represents all required sections in the SQAP outline shown in Figure 5 of this standard.

C.3.1 SQAP sect. 1 Purpose and scope

This section of the SQAP identifies and describes the purpose and scope of the SQAP for the specific project. Refer to 5.1 of this standard for a description of the purpose of SQA. Table C.3 shows questions and suggested inputs related to Purpose and Scope to consider asking during the Project Planning Phase.

Table C.3— Questions and suggested inputs related to Purpose and Scope to consider asking during Project Planning Phase

Questions	Suggested inputs
— Is the project scope clearly defined and well-understood? — Is the SQA role on this project understood by the acquirer, the organization, the project team and the SQA team? — Are potential product risks known and well-documented? — Are potential product risks understood so that SQA activities can be planned in a manner commensurate with product risk?	— Acquisition Plan — Contract — Concept of operations

C.3.2 SQAP sect. 2 Definitions and acronyms

This section of the SQAP defines all relevant terms and acronyms used in the SQAP. If it is known, definitions include a source.

C.3.3 SQAP sect. 3 Reference documents

This section of the SQAP identifies all applicable standards, industry-specific regulations and compliance documents, documents referenced by the SQAP, and any relevant supporting documents. Supporting documents may include applicable professional, industry, government, corporate, organizational, and project-specific references. Table C.4 shows questions and suggested inputs related to Reference Documents to consider asking during the Project Planning Phase.

Table C.4—Questions and suggested inputs related to Reference Documents to consider asking during Project Planning Phase

Questions	Suggested inputs
— What government regulations are applicable to this project? — What specific standards are applicable to this project? — What organizational reference documents (such as standard operating procedures, coding standards, document templates, etc.) are applicable to this project? — What project-specific reference documents are applicable to this project? — Is SQA expected to assess compliance with applicable regulations, standards, organizational documents, and project reference documents? — What reference documents are appropriate to include in the SQAP?	— Contract — Acquisition Plan — Project Plans — Document Plan

C.3.4 SQAP sect. 4 SQA plan overview

This section of the SQAP provides an overview of SQA processes and an introduction to the following topics:

— Organization and independence
— Software product risk

— Tools
— Standards, practices, and conventions
— Effort, resources, and schedule

When writing the SQAP, remember that an effective SQA Process is one that:

— Identifies what to do,
— How to do it well,
— How to confirm it gets done right,
— How to measure and track it,
— How to learn from measures to manage and improve it, and
— How to encourage using it to improve software product quality.

C.3.4.1 SQAP sect. 4.1 Organization and independence

This section of the SQAP identifies the parties responsible for performing SQA for the project and defines reporting relationships between SQA and Project Management, Software Development, and organizational Quality Management. These relationships are often represented by a functional organization chart. This section also shows the general flow of information items between all relevant parties.

If subcontractors are involved, the relationships and information flow between SQA and those subcontractors are also be shown.

This section of the SQAP also identifies project-specific roles and responsibilities related to coordinating with related software processes as defined in 5.3.2 of this standard. Refer to 5.3.2 of this standard for specific outcomes included in this section of the SQAP.

This section of the SQAP also describes the level of organizational objectivity as described in subclause 5.3.6 of this standard. Outcomes are defined in 5.3.6 of this standard.

This section of the SQAP also defines the degree of independence of the organization performing the SQA function. There are three parameters that can be used to define independence: technical independence, managerial independence, and financial independence.

Technical independence requires that SQA utilize personnel who are not involved in the development of the system or its elements. SQA forms its own assessments of all project activities. Technical independence is an important method to detect subtle errors overlooked by those too close to the solution.

Managerial independence requires that responsibility for SQA be vested in an organization separate from the software development and program management organizations. Managerial independence also means that SQA independently selects segments of software to analyze and test, chooses techniques, defines the schedule of SQA activities, and selects the specific technical issues and problems to act upon. The SQA effort provides its findings in a timely fashion simultaneously to both the software development and program management organizations.

Financial independence requires that control of the SQA budget be vested in an organization independent of the software development organization. This independence prevents situations in which SQA cannot complete its activities because funds have been diverted or adverse financial pressures or influences have been exerted.

Table C.5 shows questions and suggested inputs related to Organization and independence to consider asking during the Project Planning Phase. Table C.6 shows questions and suggested inputs related to Organization and independence to consider asking during the Project Executing Phase.

Table C.5—Questions and suggested inputs related to Organization and independence to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have deficiencies in the organization's SQA policy been identified and documented? — Has Management established a method for monitoring the execution of SQA activities, tasks, and outcomes along with a method for providing feedback to the SQA function? — Has Management established an effective and appropriate policy defining and governing SQA roles and responsibilities? — Has Management established an SQA function with sufficient influence over software processes, including an effective reporting line independent of the software development group? — Has Management established responsibility for supervising a project's SQA function by an individual independent of both the project manager and software development manager? — Has Management established a method to enable projects to learn from the experiences of previous projects, if the SQA function is being established for more than one project? — Does the SQA process exist independently of SQA processes established for individual projects? — Have adequate resources, including sufficient numbers of suitably skilled and trained people as well as sufficiently capable tools and equipment, been identified for the project as well as for other projects if the SQA function is being established for multiple projects? — Has the degree of independence (technical, managerial, and financial) been defined? — Is the defined degree of independence appropriate given the potential product risk and the requirements of the contract?	— Organizational Quality Policy — Contract

Table C.6—Questions and suggested inputs related to Organization and independence to consider asking during Project Executing Phase

Questions	Suggested inputs
— Has an appropriate and effective strategy for conducting quality assurance been developed? — Has evidence of SQA been produced and maintained? — Have problems and non-conformances with requirements been identified and recorded? — Have identified problems been addressed appropriately and effectively? — Have non-conformances been appropriately and effectively resolved? — Have corrective actions been reviewed for effectiveness? — Have preventive actions been reviewed for effectiveness? — Has root cause analysis been routinely used to determine the root causes of product and process non-conformances? — Has adherence of products, processes, and activities to the applicable standards, procedures, and requirements been verified? — Does SQA have the necessary autonomy to provide truly objective technical assessments to project management as well as to organizational quality management without fear of recrimination?	— SQAP — SQA Records — Non-conformances — Corrective Actions — Preventive Actions

C.3.4.2 SQAP sect. 4.2 Software product risk

Software Product Risk refers to the inherent risks associated with use of the software product (e.g., safety risk, financial risk, security risk, etc.). Software product risk is distinguished from project management risk. Techniques for addressing software product risk are discussed in 4.6.2 of this standard and in Annex J of this standard.

This section of the SQAP states that activities and tasks identified in this plan are performed in a manner commensurate with the defined software product risk. Refer to Annex I of this standard for suggested tasks and outcomes included in this section of the SQAP. Table C.7 shows questions and suggested inputs related to Software Product Risk to consider asking during the Project Planning Phase. Table C.8 shows questions and suggested inputs related to Software Product Risk to consider asking during the Project Executing Phase.

Table C.7—Questions and suggested inputs related to Software Product Risk to consider asking during Project Planning Phase

Questions	Suggested inputs
— Are potential product risks known and well-documented? — Are potential product risks understood so that SQA activities can be planned in a manner commensurate with product risk? — Has the scope of product risk management to be performed been determined? — Have appropriate product risk management strategies been defined and implemented? — Will a software integrity level (see Annex I) be established, if appropriate? — Does the project team have adequate training in product risk management techniques? — Is the project team planning to adjust their activities and tasks in a manner commensurate with product risk? — Are the breadth and depth of planned SQA activities commensurate with product risk?	— Acquisition Plan — Contract — Concept of operations — Risk Management Plan

Table C.8—Questions and suggested inputs related to Software Product Risk to consider asking during Project Executing Phase

Questions	Suggested inputs
— Are risks identified and analyzed as they develop? — Has the priority in which to apply resources to treatment of these risks been determined? — Are risk measures appropriately defined, applied, and assessed to determine changes in the status of risk and the progress of the treatment activities? — Has appropriate treatment been taken to correct or avoid the impact of risk based on its priority, probability, and consequence or other defined risk threshold? — Has a software integrity level scheme been defined for the project? — Has the software integrity level scheme been reviewed and determined to be appropriate? — Has a software integrity level (see Annex I) been established, if appropriate? — Has a set of assurance cases been prepared? — Have the assurance cases been reviewed and determined to be appropriate and complete? — Has an appropriate risk assessment been performed and documented?	— Risk Management Plan — Improvement Plan — Monitoring and Control Report — Risk Action Request

C.3.4.3 SQAP sect. 4.3 Tools

This section of the SQAP describes tools to be used by SQA to perform specific tasks. These may include a variety of software tools to be used as part of the SQA process. Appropriate acquisition, documentation, training, support, validation, and qualification information for each tool is included in the SQAP. Table C.9 shows questions and suggested inputs related to Tools to consider asking during the Project Planning Phase. Table C.10 shows questions related to Tools to consider asking during the Project Executing Phase.

Table C.9—Questions and suggested inputs related to Tools to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have adequate resources, including sufficiently capable tools and equipment, been identified for the project as well as for other projects if the SQA function is being established for multiple projects? — Are all tools planned to be used by SQA on this project completely identified, including: supplier, version or release number, system platform requirements, tool description, numbers of concurrent users, etc.? — Based on product risk, do these tools require validation before they can be used on this project? — Is training in the effective use of SQA tools required and, if so, is training planned?	— Project Plans

Table C.10—Questions related to Tools to consider asking during Project Executing Phase

Questions
— Have any additional SQA tools been added during the project?
— Have additional tools been properly identified and validated, if appropriate?
— Have records of tool validation been created and filed properly, if appropriate?

C.3.4.4 SQAP sect. 4.4 Standards, practices and conventions

This section of the SQAP identifies standards, practices, and conventions to be used in performing activities and tasks and for creating outcomes. Table C.11 shows questions and suggested inputs related to Standards, Practices, and Conventions to consider asking during the Project Planning Phase. Table C.12 shows questions related to Standards, Practices, and Conventions to consider asking during the Project Executing Phase.

Table C.11—Questions and suggested inputs related to Standards, Practices, and Conventions to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have all laws, regulations, standards, practices, conventions, and rules been identified? — Have specific criteria and standards against which all project plans are to be evaluated been identified and shared within the project team? — Have specific criteria and standards against which software life cycle processes (supply, development, operation, maintenance, and support processes including quality assurance) are to be evaluated been identified and shared with the project team?	— Contract

Table C.12—Questions related to Standards, Practices, and Conventions to consider asking during Project Executing Phase

Questions
— Have all plans been reviewed against defined criteria and standards?
— Have all plans been reviewed to assess consistency with contract?
— Have software products been reviewed for consistency with plans and the contract?
— Have periodic reviews and audits been performed to determine if software products fully satisfy contractual requirements?
— Have software life cycle processes been reviewed against defined criteria and standards?
— Has the contract been reviewed to assess consistency with software products?
— Have any issues and non-conformances been reported?
— Have reported issues and non-conformances been tracked to closure?

C.3.4.5 SQAP sect. 4.5 Effort, resources and schedules

This section includes estimates of the effort required to complete the activities, tasks, and outcomes as defined in the SQAP.

This section identifies appropriately qualified SQA personnel and defines their specific responsibilities and authority within the context of the project.

This section also identifies additional SQA resources, including facilities, lab space, and special procedural requirements (e.g., security access rights and documentation control) that are required to perform SQA activities.

This section includes a list of critical SQA project milestones and a schedule of planned SQA activities, tasks, and outcomes. Table C.13 shows questions and suggested inputs related to Effort, Resources, and Schedules to consider asking during the Project Planning Phase. Table C.14 shows questions related to Effort, Resources, and Schedules to consider asking during the Project Executing Phase

Table C.13—Questions and suggested inputs related to Effort, Resources, and Schedules to consider asking during Project Planning Phase

Questions	Suggested inputs
— Can estimated effort and schedules be based on past projects? — Can resource requirements for this project be determined based on past projects? — What resources (lab spaces, servers, software, databases, operating systems, security rights, document control, etc.) are required for this project? — Is effort based on factual information rather than "gut feel"? — What estimating and scheduling technique can be used for this project? — Are estimating and scheduling tools required?	— Schedules from past projects — SQA Staff skills matrix

Table C.14—Questions related to Effort, Resources, and Schedules to consider asking during Project Executing Phase

Questions
— Has the SQA schedule been updated to reflect changes to the project?
— Has the SQA schedule been updated to reflect activities and tasks that were not foreseen at the beginning of the project?
— Has the SQA schedule been updated to reflect staff changes and turnover?
— Does SQA need additional resources (lab spaces, servers, software, databases, operating systems, security rights, document control, etc.) to complete its work on schedule?
— Does SQA need additional staff to complete its work on schedule?

C.3.5 SQAP sect. 5 Activities, outcomes, and tasks

This section of the SQAP addresses product and process assurance activities, outcomes, and tasks.

C.3.5.1 SQAP sect. 5.1 Product assurance

This section of the SQAP defines specific activities, outcomes, and tasks associated with Product Assurance. Product assurance is defined in 5.4 of this standard and determines whether the software product conforms to the contract and established product requirements.

An important aspect of SQA is the establishment of confidence in the quality of the software products produced by the project. These products include not only the software and related documentation but also the plans associated with the development, operation, support, maintenance, and retirement of the software. A product may also be a software service provided to the acquirer. The outcome of the product assurance activities provides evidence that the software services, products, and any related documentation are identified in and comply with the contract and any non-conformances are identified and addressed.

Product Assurance comprises three activities:

— Evaluate plans for conformance
— Evaluate product for conformance

— Evaluate product for acceptability

Product Assurance activities provide confidence that software products are developed in conformance to established product requirements, project plans, and contractual requirements.

Product Assurance activities may include SQA personnel participating in project technical reviews, software development document reviews, and software validation testing. The activity produces reports that describe the adequacy of requirements defined in the contract and the degree of conformance of the software products or services performed to the requirements identified in the contract and applicable plans, regulations and standards. These reports are reviewed with appropriate stakeholders to determine acceptability.

C.3.5.1.1 SQAP sect. 5.1.1 Evaluate plans for conformance

This section of the SQAP includes activities and tasks for evaluating the degree to which all plans required by contract have been prepared and are consistent with the contract and with each other. Required outcomes for this activity are defined in 5.4.2 of this standard. Table C.15 shows questions and suggested inputs related to Evaluating Plans for Conformance to consider asking during the Project Planning Phase. Table C.16 shows questions and suggested inputs related to Evaluate Plans for Conformance to consider asking during the Project Executing Phase.

Table C.15—Questions and suggested inputs related to Evaluating Plans for Conformance to consider asking during Project Planning Phase

Questions	Suggested inputs
— Has the SQA function defined an activity to determine if all plans required by the contract have been identified in the Project Plan as being required? — Is the scope of the work for the project accurately defined in Project Plans? — Have the development and test environments been sized and appropriately planned? — Have project-specific roles and responsibilities with respect to all of the following processes that apply to the project been appropriately identified in Project Plans? — Software Implementation Processes (7.1 of ISO/IEC/IEEE 12207:2008) — Software Support Processes (7.2 of ISO/IEC/IEEE 12207:2008) — Software Reuse Processes (7.3 of ISO/IEC/IEEE 12207:2008) — Agreement Processes (6.1 of ISO/IEC/IEEE 12207:2008) — Project Processes (6.3 of ISO/IEC/IEEE 12207:2008) — Technical Processes (6.4 of ISO/IEC/IEEE 12207:2008) — Has the SQA function identified roles and responsibilities with respect to the Organizational Quality Management Process (6.2.5 of ISO/IEC/IEEE 12207:2008)? — Has the SQA function defined an activity related to periodically monitoring project progress and determining if progress is being accurately reported?	— Project Plan — Contract
• Has an appropriate and effective problem management strategy been developed?	

Table C.15—Questions and suggested inputs related to Evaluating Plans for Conformance to consider asking during Project Planning Phase (continued)

Questions	Suggested inputs
— Has an appropriate and effective software configuration management strategy, including change control processes, been developed? — Has the completeness and consistency of configuration items been ensured? — Has the storage, handling, and delivery of configuration items been controlled?	— Configuration Management Plan
— Has the documentation to be produced during the life cycle of the software product or service been identified? — Have standards, regulations, and corporate policies to be applied for the development of the software documentation been identified? — Are the documents to be produced in conformance with the Project Plan and the contract? — Has the content and purpose of all documentation been specified, reviewed, and approved?	— Documentation Plan
— Have the required characteristics and context of use of services been specified? — Have all relevant key stakeholders been identified? — Has traceability of stakeholder requirements to stakeholders and their needs been achieved? — Have relevant industry standards and regulatory requirements been identified? — Have specific requirements resulting from industry standards and regulatory requirements been identified and documented? — Are software development and SQA tools used on the project to be validated prior to use? — Is the basis for defining the system requirements described? — Is the basis for validating the conformance of the services defined? — Has a basis for negotiating and agreeing to supply a service or product been provided?	— System Requirements Specification
— Has a suitable strategy been developed to integrate the system according to the priorities of the system requirements? — Have suitable criteria been developed to verify compliance with the system requirements allocated to the system elements, including the interfaces between system elements?	— Integration and Test Report — Evaluation Report

Table C.16—Questions and suggested inputs related to Evaluate Plans for Conformance to consider asking during Project Executing Phase

Questions	Suggested inputs
— Does every plan required by contract appear on the Project Schedule and have a responsible author assigned? — Has the feasibility of achieving the goals of the project with available resources and constraints been evaluated? — Have the project tasks, resources, and infrastructure necessary to complete the work been sized and reliably and accurately estimated? — Have interfaces between elements in the project, and with other project and organizational units, been identified? — Have all required plans for the execution of the project been developed? — Have plans for the execution of the project been activated? — Have all tools that require validation been validated?	— Project Management Plan — Resource Request — Problem Reports — Progress Reports — Monitoring and Control Reports
— Has a project schedule with critical milestones been developed? — Is this schedule realistic and accurate given the information known? — Is project progress periodically monitored and accurately reported? — Are interfaces between elements in the project, and with other project and organizational units, monitored? — Have appropriate corrective and preventive actions been taken to correct deviations from the plan and to prevent recurrence of problems identified in the project been taken when project targets are not achieved? — What project objectives have been achieved and recorded? — What project objectives have not been achieved, why not, and what suitable actions have been taken with regard to them?	— Progress Report — Problem Report — Monitoring and Control Report
— Are risks identified as they develop? — Are risks identified and analyzed, and has the priority in which to apply resources to treatment of these risks been determined? — Are risk measures appropriately defined, applied, and assessed to determine changes in the status of risk and the progress of the treatment activities? — Has appropriate treatment been taken to correct or avoid the impact of risk based on its priority, probability, and consequence or other defined risk threshold? — Has a software integrity level scheme been defined? — Has a software integrity level been determined for the project? — Has a software integrity level (see Annex I) been established, if appropriate? — Has an appropriate risk assessment been performed?	— Risk Management Plan — Improvement Plan — Monitoring and Control Report — Risk Action Request

Table C.16—Questions and suggested inputs related to Evaluate Plans for Conformance to consider asking during Project Executing Phase (continued)

Questions	Suggested inputs
— Has an appropriate and effective problem management strategy been developed? — Are non-conformances recorded, identified, and classified? — Are non-conformances analyzed and assessed to identify acceptable solution(s)? — Are the corrective and preventive actions reviewed for effectiveness? — Are non-conformances tracked to closure? — Is the status of all non-conformances reported known?	— Problem Report — Corrective Actions — Preventive Actions — Root Cause Analysis
— Has an appropriate and effective software configuration management strategy, including change control processes, been implemented? — Have items generated by the process or project been identified, defined, and baselined? — Have modifications and releases of the items been controlled? — Have modifications and releases been made available to affected parties? — Has the status of the items and modifications been recorded and reported? — Has the completeness and consistency of configuration items been ensured? — Has the storage, handling, and delivery of configuration items been controlled?	— Configuration Management Plan — Configuration Status Report
— Has documentation been developed and made available in accordance with identified standards, regulations, and corporate policies? — Has documentation been maintained in accordance with defined criteria? — Has the documentation been verified for accuracy? — Has the information to be managed been identified? — Have the forms of the information representations been defined? — Is information transformed and disposed of as required? — Is the status of information recorded? — Is information current, complete, and valid? — Is information made available to designated parties?	— Documentation Plan

Table C.16—Questions and suggested inputs related to Evaluate Plans for Conformance to consider asking during Project Executing Phase (continued)

Questions	Suggested inputs
— Has traceability of stakeholder requirements to stakeholders and their needs been achieved? — Have relevant industry standards and regulatory requirements been identified? — Have a defined set of system functional and performance requirements describing the problem to be solved been established? — Have the appropriate techniques been performed to optimize the preferred project solution? — Have system requirements been analyzed for correctness and testability? — Have performance criteria been defined (e.g., expected response times, data load times)? — Have application security requirements been defined? — Is the impact of the system requirements on the operating environment understood? — Have the requirements been prioritized, approved, and updated as needed? — Has consistency and traceability been established between the system requirements and the customer's requirements baseline? — Have changes to the baseline been evaluated for cost, schedule, and technical impact? — Have the system requirements been communicated to all affected parties and have they been baselined?	— System Requirements Specification — Evaluation Report
— Has a system architecture design been defined that identifies the elements of the system and meets the defined requirements? — Have the system's functional and performance requirements been addressed? — Have the requirements been allocated to the elements of the system? — Have internal and external interfaces of each system element been defined? — Has verification between the system requirements and the system architecture been performed? — Are the requirements allocated to the system elements and their interfaces traceable to the customer's requirements baseline? — Has consistency and traceability between the system requirements and system architecture design been maintained? — Have the system requirements, the system architecture design, and their relationships been baselined and communicated to all affected parties? — Have human factors and ergonomic knowledge and techniques been incorporated in system design? — Have human-centered design activities been identified and performed?	— System Architecture Description — Interface Description — Evaluation Report

Table C.16—Questions and suggested inputs related to Evaluate Plans for Conformance to consider asking during Project Executing Phase (continued)

Questions	Suggested inputs
— Has the system integration been verified using the defined criteria? — Has a regression strategy been developed and applied for re-testing the system when changes are made? — Has consistency and traceability been established between the system design and the integrated system elements? — Has an integrated system been constructed that demonstrates compliance with the system design? — Has an integrated system been constructed that demonstrates a complete set of usable deliverable system elements exists?	— Integration and Test Report — Evaluation Report

C.3.5.1.2 SQAP sect. 5.1.2 Evaluate product for conformance

This section of the SQAP identifies activities and tasks related to evaluating the degree to which the software product and related documentation conform to established requirements, related plans, and the contract. Required outcomes for this activity are defined in 5.4.3 of this standard. Table C.17 shows questions and suggested inputs related to Evaluate Product for Conformance to consider asking during the Project Planning Phase. Table C.18 shows questions and suggested inputs related to Evaluate Product for Conformance to consider asking during the Project Executing Phase.

Table C.17—Questions and suggested inputs related to Evaluate Product for Conformance to consider asking during Project Planning Phase

Questions	Suggested inputs
— Has an appropriate and effective implementation strategy been defined? — Are implementation technology constraints on the design identified?	— Software Design Description — Development Plan
— Have verification criteria for software items been developed that ensure compliance with the software requirements allocated to the items? — Has an effective validation strategy been developed and implemented? — Have appropriate criteria for validation of all required work products been identified?	— Verification Plan — Validation Plan
— Has an appropriate and effective audit strategy been developed?	— Audit Plan

Table C.18—Questions and suggested inputs related to Evaluate Product for Conformance to consider asking during Project Executing Phase

Questions	Suggested inputs
— Have software items been realized? — Have software items been packaged and stored in accordance with an agreement for its supply?	— Software Design Description — Development Plan
— Are the requirements clearly defined? — Are the requirements allocated to the software elements of the system? — Are interfaces clearly defined? — Have software requirements been analyzed for correctness and testability? — Is the impact of software requirements on the operating environment understood? — Have consistency and traceability been established between the software requirements and system requirements? — Has an appropriate prioritization for implementing the software requirements been defined? — Have the software requirements been reviewed, approved, and updated as needed? — Have changes to the software requirements been evaluated for cost, schedule, and technical impact? — Have the software requirements been baselined and communicated to all affected parties? — Are requirements specific, measureable, attainable, realistic, and testable?	— Software Requirements Specification (SRS) — Evaluation Report
— Has an appropriate detailed design of each software component, describing the software units to be built, been developed? — Are external interfaces of each software unit defined? — Has consistency and traceability been established between the detailed design and the requirements and architectural design?	— Development plan — SRS — Database Design Description — Interface description — Evaluation Report — User documentation

Table C.18—Questions and suggested inputs related to Evaluate Product for Conformance to consider asking during Project Executing Phase (continued)

Questions	Suggested inputs
— Have verification criteria been defined for all software units against their requirements? — Have software units defined by the design been produced? — Have consistency and traceability been established between software units and requirements and design? — Has verification of the software units against the requirements and the design been accomplished? — Have appropriate coding standards and conventions been identified and applied? — Have adequate criteria for verification of all required software work products been identified? — Has an effective verification strategy been developed and implemented? — Have required verification activities been performed adequately? — Have defects been identified, recorded, and resolved? — Have defects been addressed appropriately? — Have results of the verification activities been made available to the customer and other involved parties?	— Software Unit Test Procedure — Software Unit Description — Software Unit Test Report — User documentation — Evaluation Report — Verification Report
— Has an integration strategy been developed for software units consistent with the software design and the prioritized software requirements? — Are software items verified using the defined criteria? — Have software items defined by the integration strategy been produced? — Have results of integration testing been recorded? — Have non-conformances been recorded and appropriately resolved? — Have consistency and traceability been established between software design and software items? — Has a regression strategy been developed and applied for re-verifying software items when a change in software units (including associated requirements, design, and code) occurs?	— Integration Plan — Integration and Test Report — User Documentation — Qualification Test Procedure — Evaluation report
— Have criteria for the integrated software been developed that demonstrate compliance with the software requirements? — Is integrated software verified using defined criteria? — Are test results recorded? — Have non-conformances been recorded and appropriately resolved? — Has a regression strategy been developed and applied for re-testing the integrated software when a change in software items is made? — Have specific completion criteria for qualification testing been defined?	— Qualification Test Report — User Documentation — Evaluation Report — Audit Report

Table C.18—Questions and suggested inputs related to Evaluate Product for Conformance to consider asking during Project Executing Phase (continued)

Questions	Suggested inputs
— Has an effective validation strategy been developed and implemented? — Have appropriate criteria for validation of all required work products been identified? — Have required validation activities been performed adequately? — Have problems been identified, recorded, and resolved? — Has evidence been provided that the software work products as developed are suitable for their intended use? — Have results of the validation activities been provided to the customer and other involved parties?	— Validation Test Specification — Validation Report
— Are stakeholder, steering committee, management, and technical reviews held based on the needs of the project? — Is the review process measured and is it effective? — Do review team members come to review meetings prepared? — Have review results been made known to all affected parties? — Have action items resulting from reviews been tracked to closure? — Have risks and problems been identified, recorded, and resolved?	— Review Minutes —
— Has an appropriate and effective audit strategy been implemented? — Has compliance of selected software work products or services or processes with requirements, plans, and agreement been determined according to the audit strategy? — Are audits conducted by an appropriate independent party? — Have the audit results been documented? — Have all issues detected during an audit been documented as non-conformances? — Have all non-conformances been considered for corrective action? — Have all corrective actions that were implemented proven to be effective as determined by effectiveness measures? — Has an appropriate justification been provided for each non-conformance not requiring corrective action?	— Audit Procedure — Audit Report — Audit Acknowledgement Report
— Has an appropriate and effective problem management strategy been developed? — Are problems recorded, identified, and classified? — Are problems analyzed and assessed to identify acceptable solution(s)? — Are problem resolutions implemented? — Are problems tracked to closure? — Is the status of all problems reported known?	— Problem Reports

C.3.5.1.3 SQAP sect. 5.1.3 Evaluate product for acceptability

This section of the SQAP identifies activities and tasks for evaluating the level of confidence that the software products and related documentation will be acceptable to the acquirer prior to delivery. Required outcomes for this activity are defined in 5.4.4 of this standard. Table C.19 shows questions and suggested inputs related to Evaluate Product for Acceptability to consider asking during the Project Planning Phase. Table C.20 shows questions and suggested inputs related to Evaluate Product for Acceptability to consider asking during the Project Executing Phase.

Table C.19—Questions and suggested inputs related to Evaluate Product for Acceptability to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have suitable criteria for evaluating compliance with system requirements been developed?	— Qualification Test Plan
— Has an appropriate software installation strategy been developed? — Have criteria for software installation been developed that demonstrate compliance with the software installation requirements?	— Installation Plan
— Has an appropriate and effective operation strategy been defined? — Have conditions for correct operation of the software in its intended environment been identified and evaluated?	— Service Management Plan — User Documentation
— Has an appropriate and effective maintenance strategy been developed to manage modification and migration of products according to the release strategy?	— Maintenance procedures

Table C.20—Questions and suggested inputs related to Evaluate Product for Acceptability to consider asking during Project Executing Phase

Questions	Suggested inputs
— Is the integrated system tested using the defined criteria? — Have test results been recorded? — Has readiness of the system for delivery been assured? — Have corrective actions plans been established for those items that did not meet the system requirements?	— Qualification Test Procedure — Audit Report — Evaluation Report
— Has the software product been installed in the target environment? — Is the readiness of the software product for use in its intended environment assured?	— Installation Plan — Installation Report
— Is the product completed and delivered to the acquirer? — Have acquirer acceptance tests and reviews been supported? — Has the product been put into operation in the customer's environment? — Have problems detected during acceptance been identified and communicated to those responsible for resolution? — Have identified problems been resolved appropriately?	— Acceptance Review and Testing Report — Problem Report

Table C.20—Questions and suggested inputs related to Evaluate Product for Acceptability to consider asking during Project Executing Phase (continued)

Questions	Suggested inputs
— Have conditions for correct operation of the software in its intended environment been identified and evaluated? — Has the software been tested and determined to operate in its intended environment? — Has the software been operated in its intended environment? — Have assistance and consultation been provided to the customers of the software product in accordance with the agreement?	— Service Management Plan — User Documentation — Operation Test Procedure — Problem Management Procedure — Problem Reports
— Has an appropriate and effective maintenance strategy been developed to manage modification and migration of products according to the release strategy? — Has the impact of changes to the existing system on organization, operations, or interfaces been identified? — Are affected system and software documentation updated as needed? — Are modified products developed with associated tests that demonstrate that requirements are not compromised? — Are product upgrades migrated to the customer's environment? — Has the system software modification been communicated to all affected parties?	— Maintenance procedures — Software Design Description — User documentation — Software Test Procedure — Problem Reports — Change Requests — Software Test Report — Review Minutes — Release Plan — User Notification

C.3.5.1.4 SQAP sect. 5.1.4 Evaluate product life cycle support for conformance

This section of the SQAP identifies activities and tasks for evaluating whether support requirements identified in the project plans are consistent with the contract and clearly identify the responsibilities of both the product delivery organization and the acquirer.

Required outcomes for this activity are defined in 5.4.5 of this standard. Table C.21 shows questions and suggested inputs related to Evaluate Product Life cycle Support for Conformance to consider asking during the Project Planning Phase. Table C.22 shows questions and suggested inputs related to Evaluate Product Life cycle Support for Conformance to consider asking during the Project Executing Phase.

Table C.21—Questions and suggested inputs related to Evaluate Product Life cycle Support for Conformance to consider asking during Project Planning Phase

Questions	Suggested inputs
— Has an appropriate and effective maintenance strategy been developed to manage modification and migration of products according to the release strategy?	— Maintenance procedures
— Has an appropriate software disposal strategy been defined?	— Disposal Plan

Table C.22—Questions and suggested inputs related to Evaluate Product Life cycle Support for Conformance to consider asking during Project Executing Phase

Questions	Suggested inputs
— Is the product completed and delivered to the acquirer? — Have acquirer acceptance tests and reviews been supported? — Has the product been put into operation in the customer's environment? — Have problems detected during acceptance been identified and communicated to those responsible for resolution? — Have identified problems been resolved appropriately?	— Acceptance Review and Testing Report — Problem Report
— Has an appropriate and effective maintenance strategy been developed to manage modification and migration of products according to the release strategy? — Has the impact of changes to the existing system on organization, operations, or interfaces been identified? — Are affected system and software documentation updated as needed? — Are modified products developed with associated tests that demonstrate that requirements are not compromised? — Are product upgrades migrated to the customer's environment? — Has the system software modification been communicated to all affected parties?	— Maintenance procedures — Software Design Description — User documentation — Software Unit Test Procedure — Problem Report — Software Unit Test Report — Review Minutes — Release Plan — User Notification
— Has an appropriate software disposal strategy been defined? — Have disposal constraints been provided as inputs to requirements? — Are the system's software elements destroyed or stored appropriately in light of safety and security requirements? — Is the environment left in an agreed-upon state? — Are records allowing knowledge retention of disposal actions and any analysis of long-term impacts available?	— SRS — User Notification — Disposal Plan

C.3.5.1.5 SQAP sect. 5.1.5 Measure products

This section of the SQAP identifies activities and tasks for evaluating whether the measurements objectively demonstrate the quality of the products in accordance with established standards and processes.

Required outcomes for this activity are defined in 5.4.6 of this standard. Table C.23 shows questions and suggested inputs related to Measure Products to consider asking during the Project Planning Phase. Table C.24 shows questions and suggested inputs related to Measure Products to consider asking during the Project Executing Phase.

Table C.23—Questions and suggested inputs related to Measure Products to consider asking during Project Planning Phase

Questions	Suggested inputs
— Are proposed metrics and measurements consistent with product risk and overall organization quality goals? — Are proposed data collection and analysis processes consistent with product risk and overall organization quality goals?	— Measurement Plan
— Have the information needs required to measure the effectiveness of technical and management processes been identified? — Has an appropriate set of measures, driven by the information needs, been identified and developed? — Have appropriate measurement activities been identified and planned?	— Monitoring and Control Report — Measurement Plan

Table C.24—Questions and suggested inputs related to Measure Products to consider asking during Project Executing Phase

Questions	Suggested inputs
— Has the required data been collected, stored, analyzed, and the results interpreted? — Have information items been used to support decisions and provide an objective basis for communication? — Have the measurement process and specific measures been evaluated? — Have improvements been communicated to the Measurement Process owner?	— Monitoring and Control Report — Measurement Plan

C.3.5.2 SQAP sect. 5.2 Process assurance

This section of the SQAP defines specific activities, tasks, and outcomes associated with Process Assurance. Process Assurance is defined in 5.5 of this standard and determines whether software processes defined in project plans are appropriate based on software product risk and whether these software processes are followed.

This SQA process activity makes certain that life cycle model processes used to develop, install, operate, and maintain software are adequate, efficient, and effective. This SQA process provides evidence that the processes that create software products comply with the contract and any non-conformances are identified and recorded. Depending on Life Cycle Model processes, these task outcomes are reported to organizations responsible for Project Planning, Quality Management, and Risk Management.

SQA activity tasks determine the degree to which software processes are capable of producing software products with quality suitable for its purposes and in fact do produce software products with needed quality and adheres to the contract and to the plans in the areas of software life cycle processes, software infrastructure, subcontractor products, measurements, and SQA staff competence.

Process Assurance comprises six activities:

— Evaluate life cycle processes for conformance
— Evaluate environments for conformance

— Evaluate subcontractor processes for conformance
— Evaluate product life cycle support for conformance
— Measure products and processes
— Assess staff skill and knowledge

C.3.5.2.1 SQAP sect. 5.2.1 Evaluate life cycle processes for conformance

This section of the SQAP identifies activities and tasks for determining the degree to which project life cycle processes and plans conform to the contract and the degree to which the execution of project activities conforms to project plans. Required outcomes for this activity are defined in 5.5.2 of this standard. Table C.25 shows questions and suggested inputs related to Life cycle Processes for Conformance to consider asking during the Project Planning Phase. Table C.26 shows questions and suggested inputs related to Life cycle Processes for Conformance to consider asking during the Project Executing Phase.

Table C.25—Questions and suggested inputs related to Life cycle Processes for Conformance to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have life cycle processes, models, and procedures for use by the organization been defined, maintained, and improved? — Have prioritized process improvements been implemented? — Have appropriate policies and procedures for the management and deployment of life cycle models and processes been provided? — Has the responsibility, accountability, and authority for life cycle management been defined? — Has an appropriate and effective decision-making strategy been defined? — At the organizational level, are organizational roles and responsibilities with respect to the Lifecycle Model Management Process (as defined in 6.2.1 of ISO/IEC/IEEE 12207:2008) identified?	— Life cycle policy and procedures — Improvement Plan — Audit Plan — Process Assessment Procedure — Process Improvement Analysis Report
— Has an effective configuration management strategy been defined? — Have roles and responsibilities for configuration management been defined? — Have configuration management tools, techniques, and methods been defined?	— Configuration Management Plan and Policy

Table C.26—Questions and suggested inputs related to Life cycle Processes for Conformance to consider asking during Project Executing Phase

Questions	Suggested inputs
— Has an effective configuration management strategy been implemented? — Have configuration management tools been validated, if appropriate? — Are items requiring configuration management defined? — Have configuration baselines been established? — Are changes to items under configuration management controlled? — Is the configuration of released items controlled? — Is the status of items under configuration management made available throughout the life cycle?	— Configuration Management Plan and Policy

C.3.5.2.2 SQAP sect. 5.2.2 Evaluate environments for conformance

This section of the SQAP identifies activities and tasks for evaluating whether software development environments and test environments conform to project plans. Required outcomes for this activity are defined in 5.5.3 of this standard. Table C.27 shows questions and suggested inputs related to Evaluate Environments for Conformance to consider asking during the Project Planning Phase. Table C.28 shows questions and suggested inputs related to Evaluate Environments for Conformance to consider asking during the Project Executing Phase.

Table C.27—Questions and suggested inputs related to Evaluate Environments for Conformance to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have the criteria, standards, and contractual requirements against which software engineering practices, development environment, and libraries are to be reviewed been identified and documented? — Is the proposed software development environment suitable based on the contract and project needs? — Is the proposed test environment suitable based on the contract and project needs? — Do the tools included in the software development environment require validation? — Do the tools included in the test environment require validation?	— Project Plan

Table C.28—Questions and suggested inputs related to Evaluate Environments for Conformance to consider asking during Project Execution Phase

Questions	Suggested inputs
— Have the development and test environments been sized and appropriately planned? — Have the test environment been sized and appropriately planned? — Have appropriate security and access controls been applied to the test environment? — Have tools used on the project been validated as required by contract or regulatory requirements?	— Project Plan

C.3.5.2.3 SQAP sect. 5.2.3 Evaluate subcontractor processes for conformance

This section of the SQAP identifies activities and tasks for evaluating whether subcontractor software processes conform to requirements passed down from the acquirer. Required outcomes for this activity are defined in 5.5.4 of this standard. Table C.29 shows questions and suggested inputs related to Evaluate Subcontractor Processes for Conformance to consider asking during the Project Planning Phase. Table C.30 shows questions and suggested inputs related to Evaluate Subcontractor Processes for Conformance to consider asking during the Project Executing Phase.

Table C.29—Questions and suggested inputs related to Evaluate Subcontractor Processes for Conformance to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have all subcontracts been reviewed and assessed from a software process perspective? — Is a subcontractor audit required for subcontractors? — Are the project responsibilities of all subcontractors clearly defined? — Are subcontract requirements traced to prime contract requirements? — Are prime contract requirements traced to subcontract requirements?	— Subcontracts — Project Plan

Table C.30—Questions and suggested inputs related to Evaluate Subcontractor Processes for Conformance to consider asking during Project Executing Phase

Questions	Suggested inputs
— Have subcontract requirements been traced to prime contract requirements? — Have prime contract requirements been traced to subcontract requirements?	— Subcontracts — Project Plan — Reports

C.3.5.2.4 SQAP sect. 5.2.4 Measure processes

This section of the SQAP identifies activities and tasks for evaluating whether the measurements support effective management of the processes in accordance with established standards and processes. Required outcomes for this activity are defined in 5.5.5 of this standard. Table C.31 shows questions and suggested inputs related to Measure Processes to consider asking during the Project Planning Phase. Table C.32 shows questions and suggested inputs related to Measure Processes to consider asking during the Project Executing Phase.

Table C.31—Questions and suggested inputs related to Measure Processes to consider asking during Project Planning Phase

Questions	Suggested inputs
— Are proposed metrics and measurements consistent with product risk and overall organization quality goals? — Are proposed data collection and analysis processes consistent with product risk and overall organization quality goals?	— Measurement Plan
— Have the information needs required to measure the effectiveness of technical and management processes been identified? — Has an appropriate set of measures, driven by the information needs, been identified or developed? — Have appropriate measurement activities been identified and planned?	— Monitoring and Control Report — Measurement Plan

Table C.32—Questions and suggested inputs related to Measure Processes to consider asking during Project Executing Phase

Questions	Suggested inputs
— Has the required data been collected, stored, analyzed, and the results interpreted? — Have information items been used to support decisions and provide an objective basis for communication? — Have the measurement process and specific measures been evaluated? — Have improvements been communicated to the measurement process owner?	— Monitoring and Control Report — Measurement Plan

C.3.5.2.5 SQAP sect. 5.2.5 Assess staff skill and knowledge

This section of the SQAP identifies activities and tasks for evaluating whether the staff assigned to the project has the required knowledge, skill, and abilities to perform the tasks required for their roles. Required outcomes for this activity are defined in 5.5.6 of this standard. Table C.33 shows questions and suggested inputs related to Assess Staff Skill and Knowledge to consider asking during the Project Planning Phase. Table C.34 shows questions and suggested inputs related to Assess Staff Skill and Knowledge to consider asking during the Project Executing Phase.

Table C.33—Questions and suggested inputs related to Assess Staff Skill and Knowledge to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have adequate resources, including sufficient numbers of suitably skilled and trained people as well as sufficiently capable tools and equipment, been identified for the project as well as for other projects if the SQA function is being established for multiple projects? — Have required project skills for staff and subcontractors been identified? — Have staff and subcontractor training records been reviewed against required skills?	— Project Plan — Training Plan

Table C.34—Questions and suggested inputs related to Assess Staff Skill and Knowledge to consider asking during Project Executing Phase

Questions	Suggested inputs
— Has a Training Gap Analysis been identified? — Have training plans that were created to address the Training Gap Analysis been prepared and reviewed?	— Project Plan — Training Plan

C.3.6 SQAP sect. 6 Additional considerations

This section of the SQAP identifies any additional SQA processes that support both Project Management and Organizational Quality Management that are not described elsewhere in this standard. The following topics are included in this section of the plan:

- Contract review
- Quality measurements
- Waivers and deviations
- Task repetition
- Risks to performing SQA
- Communications strategy
- Non-conformance process

An example of an additional process is a process for tracking and managing non-conformances and problems that are identified during the course of performing SQA tasks. This process is either defined in the SQAP or referenced if it is defined elsewhere, such as the Organizational Quality Management plan.

Any additional topics beyond those included in the SQA Plan outline in Figure 5 of this standard are included in SQAP Section 6.0.

C.3.6.1 SQAP sect. 6.1 Contract review

This section of the SQAP identifies or references the contract review process and describes SQA roles and responsibilities with regard to contract reviews. Table C.35 shows questions and suggested inputs related to Contract Review to consider asking during the Project Planning Phase. Table C.36 shows questions and suggested inputs related to Contract Review to consider asking during the Project Executing Phase.

Table C.35—Questions and suggested inputs related to Contract Review to consider asking during Project Planning Phase

Questions	Suggested inputs
— Are acquisition needs, goals, product, and service acceptance criteria, and acquisition strategies clearly defined, suitable, and complete? — Are product and service acceptance criteria defined in a manner that is measureable, suitable, and sufficient? — Does the contract clearly express the expectation, responsibilities, and liabilities of both the acquirer and the supplier? — Are the expectations within the capability of the supplier? — Do both parties have the capability to meet expectations identified in the contract? — Does the contract require ongoing monitoring so that specified constraints such as cost, schedule, and quality are met? — Have all identified contractual issues been resolved in a satisfactory manner and agreed to by the acquirer and the supplier? — Are Supplier Audits required for any critical software sub-contractors?	— Acquisition Plan — Contract — Product need assessment — Concept of operations — Request for proposal (RFP) — Supplier selection procedure

Table C.36—Questions and suggested inputs related to Contract Review to consider asking during Project Executing Phase

Questions	Suggested inputs
— Does the response to an acquirer's request address all of the acquirer's constraints and requirements, including requirements for quality? — Is the response appropriate for the anticipated software integrity level of the product or service? — Does the agreement (initial and any subsequent agreements) between the acquirer and the supplier address development, maintenance, operation, packaging, delivery, and installation of the product or service? — Is the agreement appropriate for the anticipated software integrity level of the product or service? — Are required contractual constraints such as cost, schedule, and quality being monitored on a regular basis? — Are deviations from specific contractual constraints reported and addressed? — Have required Supplier Audits for critical software sub-contractors been performed? — Have any issues raised as part of these Supplier Audits been reviewed and assessed for impact? — Have corrective/preventive action plans been developed for any non-conformances identified during the Supplier Audit?	— Contract — Review Minutes — Monitoring and Control Report — Project Management Plan — Proposal — Problem Reports — Progress Reports — Audit Reports

C.3.6.2 SQAP sect. 6.2 Quality measurement

This section of the SQAP identifies quality measurements that are appropriate for the project. This section of the SQAP identifies specific data collection requirements associated with identified measurements as well as responsibilities for data collection and reporting. Table C.37 shows questions and suggested inputs related to Quality measurement to consider asking during the Project Planning Phase. Table C.38 shows questions and suggested inputs related to Quality measurement to consider asking during the Project Executing Phase.

Table C.37—Questions and suggested inputs related to Quality measurement to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have appropriate product quality measures been identified in Project Plans? — Have appropriate process quality measures been identified in Project Plans?	— Project Plans

Table C.38—Questions and suggested inputs related to Quality measurement to consider asking during Project Executing Phase

Questions	Suggested inputs
— Has data required supporting the identified product and processing quality measures been identified and collected? — Have quality measurements been presented to the appropriate stakeholders for their review? — Have any corrective or preventive actions resulting from the review of the quality measures been initiated?	— Measurement Plan

C.3.6.3 SQAP sect. 6.3 Waivers and deviations

This section of the SQAP defines or references the criteria used to review and approve waivers and deviations to the contract and project management controls. The SQAP describes SQA roles and responsibilities with regard to reviewing and approving waivers and deviations. Table C.39 shows questions and suggested inputs related to Waivers and Deviations to consider asking during the Project Planning Phase. Table C.40 shows questions and suggested inputs related to Waivers and Deviations to consider asking during the Project Executing Phase.

Table C.39—Questions and suggested inputs related to Waivers and Deviations to consider asking during Project Planning Phase

Questions	Suggested inputs
— What deviations from the SQA Plan outline shown in Figure 5 of this standard are being considered? — What deviations from contract or project plans or controls are being considered? — What justification is provided for each proposed waiver or deviation?	— Acquisition plan — Contract — Project Plan

Table C.40—Questions and suggested inputs related to Waivers and Deviations to consider asking during Project Executing Phase

Questions	Suggested inputs
— What additional deviations from contract or project controls are being considered? — What justification is provided for each proposed waiver or deviation?	— Acquisition plan — Contract — Project Plan

C.3.6.4 SQAP sect. 6.4 Task repetition

This section of the SQAP defines or references the criteria used to determine when and under what conditions SQA tasks previously completed need to be repeated. Table C.41 shows questions and suggested inputs related to Task Repetition to consider asking during the Project Planning Phase. Table C.42 shows questions and suggested inputs related to Task Repetition to consider asking during the Project Executing Phase.

Table C.41—Questions and suggested inputs related to Task Repetition to consider asking during Project Planning Phase

Questions	Suggested inputs
— What SQA tasks are likely to need to be repeated based on the nature of the software product and the software development process to be used by the project? — For each SQA task, what criteria need to be defined to help determine the conditions that would require the task to be repeated?	— Project Plans

Table C.42—Questions and suggested inputs related to Task Repetition to consider asking during Project Executing Phase

Questions	Suggested inputs
— Have additional SQA tasks requiring repetition been identified that were not initially identified?	— SQA Plan

C.3.6.5 SQAP sect. 6.5 Risks to performing SQA

This section of the SQAP identifies potential projects risks that could prevent SQA from accomplishing its defined purposes, activities, and tasks. Examples include inadequate staffing levels, insufficient resources, and lack of training. Also included in this section are actions taken to mitigate identified project risks. Table C.43 shows questions and suggested inputs related to Risks to performing SQA to consider asking during the Project Planning Phase. Table C.44 shows questions and suggested inputs related to Risks to performing SQA to consider asking during the Project Executing Phase.

Table C.43—Questions and suggested inputs related to Risks to performing SQA to consider asking during Project Planning Phase

Questions	Suggested inputs
— Have all of the potential project risks that could prevent SQA from accomplishing their project responsibilities been identified and reviewed with project management? — Has a mitigation plan been identified for each risk?	— Contract — Project Plan

Table C.44—Questions and suggested inputs related to Risks to performing SQA to consider asking during Project Executing Phase

Questions	Suggested inputs
— Have additional project risks appeared that could prevent SQA for accomplishing their project responsibilities? — Has each of these new risks been identified and reviewed with project management? — Have a mitigation plan been identified for each risk?	— Contract — Project Plan

C.3.6.6 SQAP sect. 6.6 Communications strategy

This section of the SQAP defines the strategies for communicating SQA activities, tasks, and outcomes to the project team, management, and organizational quality management. Table C.45 shows questions and suggested inputs related to Communications Strategy to consider asking during the Project Planning Phase. Table C.46 shows questions and suggested inputs related to Communications Strategy to consider asking during the Project Executing Phase.

Table C.45—Questions and suggested inputs related to Communications Strategy to consider asking during Project Planning Phase

Questions	Suggested inputs
— What communication mechanisms would be most effective for this project? — What communications mechanisms are required for interfacing with multiple development sites, third parties, and subcontractors? — What reports and summaries will SQA prepare for presentation to appropriate stakeholders? — How often will these presentations be made?	— Contract — Project Plan

Table C.46—Questions and suggested inputs related to Communications Strategy to consider asking during Project Executing Phase

Questions	Suggested inputs
— What feedback has SQA received regarding its chosen communication strategies? — What changes are needed to accommodate project and stakeholder needs?	— Contract — Project Plan

C.3.6.7 SQAP sect. 6.7 Non-conformance process

This section of the SQAP defines activities and tasks related to the process for reporting non-conformances for the project. Non-conformances can be reported by any project member but can only be closed by SQA. Additional information on non-conformances, corrective and preventive actions, and root cause analysis can be found in Annex J of this standard. Table C.47 shows questions and suggested inputs related to Problem Reporting and Corrective Action to consider asking during the Project Planning Phase. Table C.48 shows questions and suggested inputs related to Problem Reporting and Corrective Action to consider asking during the Project Executing Phase.

Table C.47—Questions and suggested inputs related to Problem Reporting and Corrective Action to consider asking during Project Planning Phase

Questions	Suggested inputs
— What contractual requirements related to non-conformances need to be met? — What mechanisms need to be defined to support an effective system for reporting non-conformances? — What mechanisms need to be defined to support an effective corrective and preventive action process? — What mechanisms need to be defined to support and effective system for performing root cause analysis? — Are there existing processes and procedures related to reporting non-conformances that can be used for this project? — Are there existing processes and procedures related to corrective and preventive action that can be used for this project? — Are there existing processes and procedures related to root cause analysis that can be used for this project?	— Contract — Project Plan

Table C.48—Questions and suggested inputs related to Problem Reporting and Corrective Action to consider asking during Project Executing Phase

Questions	Suggested inputs
— Is the current non-conformance process working as intended? — What issues with the non-conformance process need to be addressed and improved? — Is the current corrective and preventive action process working as intended? — What issues with the corrective and preventive action process need to be addressed and improved?	— Contract — Project Plan

C.3.7 SQAP sect. 7 SQA records

This section identifies records and reports to be prepared by SQA as required by Project Management and Organizational Quality Management.

C.3.7.1 SQAP sect. 7.1 Analyze, identify, collect, file, maintain, and dispose

This section of the SQAP includes activities and tasks for analysis, identification, collection, filing, maintenance, and disposition of quality records. Required outcomes for this activity are defined in 5.3.5 of this standard.

Quality records document that activities were performed in accordance with project plans and the contract. These records enable information sharing, and support analysis to identify problems, causes, and eventually result in product and process improvements. Table C.49 shows questions and suggested inputs related to analyze, identify, collect, file, maintain, and dispose to consider asking during the Project Planning Phase. Table C.50 shows questions and suggested inputs related to analyze, identify, collect, file, maintain, and dispose to consider asking during the Project Executing Phase.

Table C.49—Questions and suggested inputs related to analyze, identify, collect, file, maintain, and dispose to consider asking during Project Planning Phase

Questions	Suggested inputs
— What set of records is the project required to produce? — What set of records is SQA required to produce? — Is the information required to be included on each record defined? — What mechanisms will be used to collect, file, maintain, and eventually dispose of quality records? — Who is responsible for collecting, filing, maintaining, and disposing of records? — Who is responsible for sharing records? — What records are to be shared with stakeholders? — What protections are required to be established in order to share these records, maintain the integrity of the records, and prevent their modification or inadvertent release? — What records are required from subcontractors?	— Contract — Project Plan — Documentation Plan

Table C.50—Questions and suggested inputs related to analyze, identify, collect, file, maintain, and dispose to consider asking during Project Executing Phase

Questions	Suggested inputs
— Are all required records being created? — Do all records contain required information content? — Are all required records properly filed and maintained? — Are all required records shared as intended with appropriate stakeholders?	— Contract — Project Plan — Documentation Plan

C.3.7.2 SQAP sect. 7.2 Availability of records

This section of the SQAP includes activities and tasks related to making records of project activities and tasks available as specified in the contract. Required outcomes for this activity are defined in 5.3.5 of this standard.

Quality records document that activities were performed in accordance with project plans and the contract. These records enable information sharing, and support analysis to identify problems, causes, and eventually result in product and process improvements. Table C.51 shows questions and suggested inputs related to Availability of Records to consider asking during the Project Planning Phase. Table C.52 shows questions and suggested inputs related to Availability of Records to consider asking during the Project Executing Phase.

Table C.51—Questions and suggested inputs related to Availability of Records to consider asking during Project Planning Phase

Questions	Suggested inputs
— Who is responsible for sharing records? — What records are to be shared with stakeholders? — What protections are required to be established in order to share these records? — What mechanisms are in place to enable records to be shared as required by the contract and project plans?	— Contract — Project Plan — Documentation Plan

Table C.52—Questions and suggested inputs related to Availability of Records to consider asking during Project Executing Phase

Questions	Suggested inputs
— Are all required records properly filed and maintained? — Are all required records shared with appropriate stakeholders according to the contract and project plans?	— Contract — Project Plan — Documentation Plan

Annex D

(informative)

Mapping IEEE Std 730-2014 and ISO/IEC 15504 (SPICE)

ISO/IEC 15504-2:2003 [B35] defines in Clause 5, Measurement framework for process capability, the Process Attributes (PA) to enable a rating of process capabilities from Capability Level 0*: Incomplete process* to Capability Level 5: *Optimizing process*. IEEE Std 730-2014 is composed of outcomes and tasks grouped into activities and is based on ISO/IEC/IEEE 12207:2008 which is also a commonly used model against which processes are assessed using ISO/IEC 15504-2:2003.

D.1 Capability Level 1 for SQA

Projects that comply with IEEE Std 730-2014 reach Capability Level 1: *Performed process* for the software quality assurance (SQA) process, which achieves its process purpose and the defined outcomes (PA 1.1).

D.2 Capability Level 2 for SQA

To also be rated as Capability Level 2: *Managed Process*, the SQA process fulfills PA 2.1 *Performance management* and PA 2.2 *Work product management*. The following tables illustrate their meanings by replacing “the process” of ISO/IEC 15504-2:2003 with “SQA process,” and replacing “work products” with “SQA work products,” and states where IEEE Std 730-2014 supports the fulfillment of the attribute.

Table D.1 shows mapping between ISO/IEC 15504-2:2003 PA2.1 and IEEE Std 730-2014.

Table D.1—Mapping between ISO/IEC 15504-2:2003 PA2.1 and IEEE Std 730-2014

ISO/IEC 15504-2:2003 PA 2.1 Performance management attribute	IEEE Std 730-2014 SQA
a) Objectives for the performance of SQA process are identified.	5.3.1 Establish the SQA processes 5.3.3 Document SQA planning Work product(s): SQAP Section 1 Purpose and scope
b) Performance of SQA process is planned and monitored.	5.3.3 Document SQA planning 5.3.4 Execute the SQA Plan Work product(s): SQAP
c) Performance of SQA process is adjusted to meet plans.	5.3.4 Execute the SQA Plan Work product(s): SQA Records
d) Responsibilities and authorities for performing SQA process are defined, assigned and communicated.	5.3.4 Execute the SQA Plan 5.3.6 Evaluate organizational independence and objectivity Work product(s): SQAP Section 4.5 Effort, resources, and schedule SQAP Section 6 Additional considerations
e) Resources and information necessary for performing SQA process are identified, made available, allocated and used.	5.3.4 Execute the SQA Plan Work product(s): SQAP Section 4.4 Standards, practices and conventions SQAP Section 4.5 Effort, resources, and schedule SQAP Section 6 Additional considerations
f) Interfaces between the involved parties are managed to ensure both effective communication and also clear assignment of responsibility.	5.3.2 Coordinate with related software processes 5.3.4 Execute the SQA Plan 5.3.5 Manage SQA records 5.3.6 Evaluate organizational independence and objectivity 5.4.6 Measure products 5.5.5 Measure processes Work product(s): SQAP Section 4.1 Organization and independence SQAP Section 7 SQA records SQAP Section 6 Additional considerations

Table D.2 shows mapping between ISO/IEC 15504-2:2003 PA2.2 and IEEE Std 730-2014.

Table D.2—Mapping between ISO/IEC 15504-2:2003 PA2.2 and IEEE Std 730-2014

ISO/IEC 15504-2 PA 2.2 Work product management attribute	IEEE Std 730-2014 SQA
a) Requirements for the SQA work products of the process are defined;	Work product(s): SQAP Section 7 SQA records
b) Requirements for documentation and control of the SQA work products are defined;	Work product(s): SQAP Section 7 SQA records
c) SQA work products are appropriately identified, documented, and controlled;	Work product(s): SQAP Section 7 SQA records
d) SQA work products are reviewed in accordance with planned arrangements and adjusted as necessary to meet requirements.	Work product(s): SQAP Section 7 SQA records

Requirements for the software quality assurance plan (SQAP) with regard to PA 2.2a) are typically described in the overall SQA process description, in checklists to be used in reviewing the SQAP, or in templates from which SQAP plans are derived. Declaring conformance of the SQAP to IEEE Std 730-2014 in the beginning of the SQAP also defines the requirements. Requirements for documentation and control of the SQAP itself may also be defined in an overall configuration management plan and/or a project documentation plan.

D.3 Capability Level 3 to Capability Level 5 for SQA

Capability Level 3: *Established Process* means that the SQA process fulfills PA 3.1 *Process definition* and PA 3.2 *Process deployment*. IEEE Std 730-2014 may be used as a basis for defining the standard process according to PA 3.1. Section 6 of the SQAP may be used to document how tailoring guidelines have been applied to the specific project. An SQAP template may offer appropriate help for the author by giving tailoring guidelines if they are not already documented elsewhere.

PA 3.2 *Process deployment* is well supported by 5.5, Process assurance activities, of IEEE Std 730-2014. All activities SQA undergoes to assure any Software Life Cycle (SLC) process are also valid for the SQA process itself. In order to ensure objective evidence, SQA processes for a specific project are to be assessed by the overall SQA of the organization or by independent consultants or other independent SQA personnel that are hired by upper management. Subclause 5.3.1, Establish the SQA processes, of IEEE Std 730-2014, gives some additional hints concerning adherence to ISO 9001:2008.

PA 3.2 in subclause f) requires collection and analysis of appropriate data for a better understanding of the process. Subclause 5.4.6, Measure products, of IEEE Std 730-2014, and 5.5.5, Measure processes, may also be interpreted to be used to measure SQA itself, but is aimed at measuring SLC processes.

Subclause 5.5.5, Measure processes, of IEEE Std 730-2014, supports both process attributes PA 4.1 *Process measurement* and PA 4.2 *Process control* for Capability Level 4: *Predictable process*. Industrial practice measures for SLC processes also indicate the effectiveness and efficiency of SQA itself. Care has to be taken to differentiate the process information needs and derived process measurement objectives for SQA and for other SLC processes.

Capability Level 5: *Optimizing the process* is, with its two process attributes PA 5.1 *Process innovation* and PA 5.2 *Process optimization*, closely related to Capability Level 4 and therefore also to 5.5.5, Measure processes, of IEEE Std 730-2014.

D.4 Capability Level 3 to Capability Level 5 for other SLC processes

IEEE Std 730-2014 supports reaching Capability Level 3 for any SLC process by assuring the adherence to the defined process as especially defined in subclause f) of PA 3.2 *Process deployment* and also gives help for fulfillment of PA 4.1, PA 4.2, PA 5.1, and PA 5.2. Nearly all activities defined under 5.5 of IEEE Std 730-2014 are directly supporting PA 3.2 as described in Table D.3 and therefore indirectly help to perform PA 3.1.

Table D.3 shows mapping between ISO/IEC 15504-2:2003 PA3.2 and IEEE Std 730-2014.

Table D.3— Mapping between ISO/IEC 15504-2:2003 PA3.2 and IEEE Std 730-2014

ISO/IEC 15504-2:2003 PA 3.2 Process deployment attribute	**IEEE Std 730-2014 SQA**
a) A defined process is deployed based upon an appropriately selected and/or tailored standard process;	5.5.2 Evaluate life cycle processes and plans for conformance
b) Required roles, responsibilities, and authorities for performing the defined process are assigned and communicated;	5.5.2 Evaluate life cycle processes and plans for conformance
c) Personnel performing the defined process are competent on the basis of appropriate education, training, and experience;	5.5.6 Assess staff skill and knowledge
d) Required resources and information necessary for performing the defined process are made available;	5.5.2 Evaluate life cycle processes and plans for conformance
e) Required infrastructure and work environment for performing the defined process are made available, managed, and maintained;	5.5.2 Evaluate life cycle processes and plans for conformance 5.5.3 Evaluate environments for conformance
f) Appropriate data are collected and analyzed as a basis for understanding the behavior of, and to demonstrate the suitability and effectiveness of the process, and to evaluate where continuous improvement of the process can be made.	5.4.6 Measure products 5.5.5 Measure processes

Capability Level 4: *Predictable process*, especially PA 4.1 *Process measurement*, is well supported by IEEE Std 730-2014, 5.4.6, Measure products, and 5.5.5, Measure processes.

Table D.4 shows mapping between ISO/IEC 15504-2:2003 PA4.1 and IEEE Std 730-2014.

Table D.4— Mapping between ISO/IEC 15504-2:2003 PA4.1 and IEEE Std 730-2014

ISO/IEC 15504-2:2003 PA 4.1 Process measurement attribute	IEEE Std 730-2014 SQA
a) Process information needs in support of relevant defined business goals are established;	5.5.5 Measure processes
b) Process measurement objectives are derived from process information needs;	5.5.5 Measure processes
c) Quantitative objectives for process performance in support of relevant business goals are established;	5.5.5 Measure processes
d) Measures and frequency of measurement are identified and defined in line with process measurement objectives and quantitative objectives for process performance;	5.5.5 Measure processes
e) Results of measurement are collected, analyzed and reported in order to monitor the extent to which the quantitative objectives for process performance are met;	5.5.5 Measure processes
f) Measurement results are used to characterize process performance.	5.4.6.3 5) Evaluate product measurement results to determine whether improvements implemented as a result of product quality measurements are effective.

Table D.5 shows mapping between ISO/IEC 15504-2:2003 PA4.2 and IEEE Std 730-2014.

Table D.5— Mapping between ISO/IEC 15504-2 PA4.2 and IEEE Std 730-2014

ISO/IEC 15504-2:2003 PA 4.2 Process control attribute	IEEE 730-2014 SQA
a) Analysis and control techniques are determined and applied were applicable;	5.5.5 Measure processes
b) Control limits of variation are established for normal process performance;	Not explicitly stated in IEEE Std 730-2014
c) Measurement data are analyzed for special causes of variation;	Not explicitly stated in IEEE Std 730-2014
d) Corrective actions are taken to address special causes of variation;	Not explicitly stated in IEEE Std 730-2014
e) Control limits are re-established (as necessary) following corrective action.	5.4.6.3 4) Analyze product measurement results to identify gaps and recommend improvements to close gaps between measurements and expectations.

Capability Level 5: *Optimizing Process* is not explicitly supported by IEEE Std 730-2014, as this involves tasks typically performed by software process improvement teams and not the SQA role.

Annex E

(informative)

Applying IEEE Std 730-2014 — Industry-specific guidance

The purpose of this annex is to provide guidance in the use and application of IEEE Std 730-2014 in different industries.

E.1 Medical device industry

The medical device industry is regulated by government agencies. In the US, the regulatory body is the Food and Drug Administration (US FDA). This discussion is focused on the regulatory requirements in effect in the US at the time this standard was published. While specific regulatory requirements in other parts of the world may differ from the US, the US requirements are generally considered to be the most stringent, especially with respect to medical device software.

Medical devices often contain software that is safety-critical. Many medical devices include embedded software and there are a few examples of software-only devices (software that is designed to run on general purpose computers). Medical device regulations apply to devices that contain embedded software as well as to software-only devices.

E.1.1 Relevant regulations, guidance documents, and international standards

E.1.1.1 Regulations

The primary US FDA medical device regulation falls under Title 21 of the US Code and is published in the Code of Federal Regulations (CFR) as 21 CFR Part 820—Quality System Regulation (QSR), 1996 [B8].

E.1.1.1.1 21 CFR Part 820 – Quality System Regulation (QSR), 1996

The QSR includes requirements related to the methods used in, and the facilities and controls used for, designing, manufacturing, packaging, labeling, storing, installing, and servicing of medical devices intended for human use.

This regulation applies to all medical devices and the software embedded within them. It also applies to software-only devices. The regulation is supplemented by many device-specific guidance documents that help explain how the FDA interprets the regulation in specific situations.

The core of the regulation is Section 820.30, Design Controls. This section identifies specific requirements in the following areas:

- General Requirements
- Design and Development Planning
- Design Inputs
- Design Outputs

— Design Reviews
— Design Verification
— Design Validation
— Design Changes
— Design Transfer
— Design History File

The following is the FDA definition of Quality:

> "The totality of features and characteristics that bear on the ability of a device to satisfy fitness-for-use, including safety and performance." [B8]

Note that the word "quality" appears 87 times in this regulation.

E.1.1.2 FDA guidance documents

E.1.1.2.1 General Principles of Software Validation (GPSV) Final Guidance for Industry and Staff, 2002 [B16]

The purpose of this guidance document is to provide a common understanding of software verification and software validation for both medical device manufacturers and FDA inspectors. This guidance document also identifies requirements for validating software tools (purchased, off-the-shelf, open source, or internally developed) that may affect the safety and efficacy of medical devices developed with those tools.

Key definitions specific to the medical device industry include:

Software verification: provides objective evidence that the design outputs of a particular phase of the software development life cycle meet all of the specified requirements for that phase. Software verification looks for consistency, completeness, and correctness of the software and its supporting documentation, as it is being developed, and provides support for a subsequent conclusion that software is validated. Software testing is one of many verification activities intended to confirm that software development output meets its input requirements. Other verification activities include various static and dynamic analyses, code and document inspections, walkthroughs, and other techniques.

FDA considers "software validation" to be "confirmation by examination and provision of objective evidence that software specifications conform to user needs and intended uses, and that the particular requirements implemented through software can be consistently fulfilled." [B16]

Key quotes from GPSV:

> "Software quality assurance needs to focus on preventing the introduction of defects into the software development process and not on trying to 'test quality into' the software code after it is written. Software testing is very limited in its ability to surface all latent defects in software code." [B16]

> "Software testing is a necessary activity. However, in most cases software testing by itself is not sufficient to establish confidence that the software is fit for its intended use." [B16]

E.1.1.2.2 Guidance for Content of Premarket Submissions for Software Contained in Medical Devices, 2005 [B20]

This guidance document provides information regarding the documentation that FDA requires in premarket submissions for software devices, including standalone software applications and hardware-based devices that incorporate software.

E.1.1.2.3 Guidance for Industry, Cybersecurity for Networked Medical Devices Containing Off-the-Shelf (OTS) Software, 2005 [B18]

A growing number of medical devices are designed to be connected to computer networks. Many of these networked medical devices incorporate off-the-shelf software that is vulnerable to cyber security threats such as viruses and worms. These vulnerabilities may represent a risk to the safe and effective operation of networked medical devices and typically require an ongoing maintenance effort throughout the product life cycle to assure an adequate degree of protection. FDA issued this guidance to clarify how existing regulations, including the Quality System (QS) Regulation, apply to such cyber security maintenance activities.

E.1.1.2.4 Guidance for Industry, FDA Reviewers and Compliance on Off-The-Shelf Software Use in Medical Devices, 1999 [B19]

Off-the-shelf (OTS) software is commonly being considered for incorporation into medical devices as the use of general purpose computer hardware becomes more prevalent. The use of OTS software in a medical device allows the manufacturer to concentrate on the application software needed to run device-specific functions. However, OTS software intended for general purpose computing may not be appropriate for a given specific use in a medical device. The medical device manufacturer using OTS software generally gives up software life cycle control, but still bears the responsibility for the continued safe and effective performance of the medical device.

This guidance document was developed to address the many questions asked by medical device manufacturers regarding what they need to provide in a pre-market submission to the FDA when they use OTS software. The specific response to these questions depends on the medical device in question and the impact on patient, operator, or bystander safety if the OTS software fails. Thus, the answer to the question, “What do I need to document?” may differ and is based on the risk analysis that is an integral part of designing a medical device. The detail of documentation to be provided to the FDA and the level of life cycle control necessary for the medical device manufacturer increase as severity of the hazards to patients, operators, or bystanders from OTS software failure increases.

This document lays out in broad terms how the medical device manufacturer can consider what is necessary to document for submission to the agency. A BASIC set of need-to-document items is recommended for all OTS software, and a detailed discussion is provided on additional (SPECIAL) needs and responsibilities of the manufacturer when the severity of the hazards from OTS software failure become more significant.

E.1.1.2.5 Guidance for Industry, Computerized Systems Used in Clinical Trials, 2007 [B17]

This document provides to sponsors, contract research organizations (CROs), data management centers, clinical investigators, and institutional review boards (IRBs), recommendations regarding the use of computerized systems in clinical investigations. The computerized system applies to records in electronic form that are used to create, modify, maintain, archive, retrieve, or transmit clinical data required to be maintained, or submitted to the FDA. Because the source data are necessary for the reconstruction and evaluation of the study to determine the safety of food and color additives and safety and effectiveness of

new human and animal drugs, and medical devices, this guidance is intended to assist in ensuring confidence in the reliability, quality, and integrity of electronic source data and source documentation (i.e., electronic records).

E.1.1.3 Relevant international standards

E.1.1.3.1 ISO/IEC 13485:2003 Medical devices — Quality management systems — Requirements for regulatory purposes [B32]

This international standard specifies requirements for a quality management system when an organization needs to demonstrate its ability to provide medical devices and related services that consistently meet customer requirements and regulatory requirements applicable to medical devices and related services.

Compliance with ISO/IEC 13485 requires establishment of a Quality System which is often embodied in a Quality Manual with accompanying standard operating procedures (SOPs). From the perspective of IEEE Std 730-2014, this Quality Manual and the accompanying SOPs are an example of an organizational Quality Plan.

E.1.1.3.2 ISO/IEC 14971:2007 Medical devices — Application of risk management to medical devices [B33]

This international standard specifies a process for a manufacturer to identify the hazards associated with medical devices, including *in vitro* diagnostic (IVD) medical devices, to estimate and evaluate the associated risks, to control these risks, and to monitor the effectiveness of the controls. The requirements of this international standard are applicable to all stages of the life-cycle of a medical device.

Note that Risk Management for medical devices is not the same as risk management as defined in ISO/IEC/IEEE 12207:2008.

E.1.1.3.3 ANSI/AAMI/IEC 62304:2006 Medical device software — Software life cycle processes [B3]

The purpose of this standard is to define the life cycle requirements for medical device software. The set of activities and tasks defined in this standard establishes a common framework for medical device life cycle development processes.

This standard is based on ISO/IEC/IEEE 12207:2008 and is specifically aimed at development of medical device software.

E.1.1.3.4 IEC 60601-1-4, Medical electrical equipment – Part 1–4: General requirements for safety – Collateral Standard: programmable electrical medical systems, 2000 [B21]

This standard applies to the safety of medical electrical systems, as defined as follows: combination of items of equipment, at least one of which is medical electrical equipment and inter-connected by functional connection or use of a multiple portable socket-outlet. It describes the safety requirements necessary to provide protection for the patient, the operator, and surroundings.

E.2 Nuclear power generation industry

The nuclear power generation industry is regulated by government agencies. Links to several regulatory bodies are listed below in Table E.1.

Table E.1—Nuclear regulatory agencies by country

Country	Nuclear Regulatory Agency	Link
Canada	Canadian Nuclear Safety Commission (CNSC)	www.nuclearsafety.gc.ca/eng
China	China Atomic Energy Authority (CAEA)	www.caea.gov.cn
France	French Safety Authority (ASN)	www.french-nuclear-safety.fr
Germany	Federal Ministry for the Environment, Nature Conservation, and Nuclear Safety	www.bmu.de
India	Atomic Energy Regulatory Board	www.aerb.gov.in
Japan	Nuclear and Industrial Safety Agency (NISA)	www.nisa.meti.go.jp
Russia	Russian Nuclear Safety Administration	None found
United Kingdom	Office for Nuclear Regulation	www.hse.gov.uk/nuclear
United States	Nuclear Regulatory Commission (NRC)	www.nrc.gov

This discussion is focused on the regulatory requirements in effect in the US at the time this standard was published. Specific regulatory requirements in the US ensure that safety-related structures, systems and components, as defined in Title 10 of the US Code of Federal Regulations, Part 50 [B7], perform satisfactorily in service to assure safe plant operation.

Nuclear power plants have predominantly used analog controls and highly reliable components. Recent developments in the nuclear power industry have led to the increased use of digital computers and software for some safety-related (e.g., Reactor Protection Systems) and many non-safety related (e.g., chiller control) functions. Organizations such as the US NRC and the Electric Power Research Institute (EPRI) have developed regulatory guides and technical reports that provide guidance and recommendations in developing software specifically for use in nuclear power plants. In addition, proposed designs for future nuclear plants include increased used of digital control systems that require highly reliable software.

Regulatory requirements are also applied to software used in the design and analysis of safety-related structures, systems, and components. The regulatory guides and technical reports from the US NRC and EPRI also apply to design and analysis software.

The regulations, guidance, and standards discussed here typically require more stringent application of software quality assurance (SQA) practices throughout the software development life cycle for software used in the operation of both safety-related and non-safety related systems. SQA practices throughout the software development life cycle also apply to software used for the design and analysis of nuclear power plants.

Commercial systems (i.e., software-based systems not developed specifically for nuclear power plant applications) may be used as part of a nuclear power plant's safety-related systems and components. Commercially-available software may also be used in the design and analysis of safety-related systems. IEEE Std 730-2014 can be used as a guide for acquisition and dedication of commercial software for use in nuclear power plant applications.

In this annex, examples of relevant regulations, guidelines, technical reports, and standards are listed and discussed. Each section contains a description of the relationship between this standard (IEEE Std 730-2014) and the listed documents.

E.2.1 Examples of relevant regulations, guidance documents, and international standards

E.2.1.1 US NRC regulations

The primary US NRC regulation falls under Title 10 of the US Code of Federal Regulations (CFR). 10 CFR Part 50 [B7] Appendix B addresses quality assurance, and 10 CFR Part 21 [B6] addresses reporting of defects and noncompliance, and touches on the subject of dedication of commercial grade items. Neither section mentions software specifically, but the regulations are applied to software.

a) 10 CFR Part 50 — Domestic Licensing of Production and Utilization Facilities. [B7]

   This regulation is used in the US as the basis for licensing nuclear power generating stations. 10 CFR Part 50 Appendix B to this regulation is Quality Assurance Criteria for Nuclear Power Plants and Fuel Reprocessing Plants.

   Appendix B requirements are directly applicable to US licensees (including applicants), and contractually imposed on suppliers.

   Appendix B includes the following provisions:

   US licensees (including applicants) are required to supply a description of the *quality assurance program* in the design of the structures, systems, and components of the facility.

   "Quality assurance" comprises all those planned and systematic actions necessary to provide adequate confidence that a structure, system, or component will perform satisfactorily in service. Quality assurance includes quality control, which comprises those quality assurance actions related to the physical characteristics of a material, structure, component, or system which provide a means to control the quality of the material, structure, component, or system to predetermined requirements.

   As defined by this standard, "Software Quality Assurance" is a set of independent activities that define and assess the adequacy of software processes to provide evidence that establishes confidence that the processes are appropriate for and produce software products of suitable quality for their intended purposes. This is consistent with the regulatory requirement that a quality assurance program provide adequate confidence that structures, systems, or components, including software, will perform satisfactorily in service.

   While not specifically endorsed by the US Nuclear Regulatory Commission as a means of compliance with 10 CFR Part 50 Appendix B, IEEE Std 730-2014 is aligned with ASME NQA-1-2008 [B1] and ASME NQA-1a-2009 [B2], and this standard may be used to aid in the development of a software quality assurance program that meets the requirements of 10 CFR Part 50 Appendix B. Conformance with IEEE Std 730-2014 does not guarantee compliance with applicable NRC Regulations.

E.2.1.2 Canadian NSC regulations

Two Canadian regulations address or have an impact on SQA. CAN/CSA N286.7-99 [B4] addresses design computer programs, and CAN/CSA N290.14-07 [B5] addresses pre-developed software for use in instrumentation and control (I&C) applications.

E.2.1.2.1 CAN/CSA N286.7-99, Quality Assurance of Analytical, Scientific and Design Computer Programs for Nuclear Power Plants

CAN/CSA N286.7-99 specifies the Quality Program requirements applicable to the design, development, maintenance, modification, and use of computer programs used in nuclear power plant applications to perform or support:

a) Design and analysis of safety related equipment, systems, structures, and components

b) Deterministic and probabilistic safety analyses and reliability studies

c) Reactor physics and fuel management calculations

d) Transfer of data between computer programs associated with a), b), or c)

e) Pre or post-processing calculations associated with a), b), or c)

The standard does not apply to computer programs used to control plant safety systems or operational control systems.

Compliance with IEEE Std 730-2014, so long as independence is satisfied, is expected to satisfy CAN/CSA N286.7-99 requirements.

E.2.1.2.2 CAN/CSA N290.14-07, Qualification of Pre-Developed Software for Use in Safety-Related Instrumentation and Control Applications in Nuclear Power Plants

CAN/CSA N290.14-07 establishes a qualification process for pre-developed software used in nuclear safety-related I&C applications. It categorizes software as falling within one of four safety categories and then assigns levels of rigor in Verification and Validation (V&V) based upon this along with software complexity and product maturity.

Compliance with IEEE Std 730-2014, so long as independence is satisfied, is expected to satisfy CAN/CSA N290.14-07 requirements.

E.2.1.3 Examples of guidance documents

Guidance documents include regulatory guides and technical reports from the US NRC, and Technical Reports produced by EPRI, and the Nuclear Energy Institute (NEI). Many of these documents impose specific requirements for SQA; while others deal with other parts of the software development life cycle. This section lists a sampling of the applicable guidance documents.

The activities in IEEE Std 730-2014 may be used to meet all or part of the requirements in the guidance documents below.

E.2.1.3.1 US NRC Regulatory Guides

The US NRC publishes many Regulatory Guides, which provide guidance in interpreting and applying the provisions of the 10 CFR Part 50 Appendix B regulations and endorse consensus standards as meeting the requirements of 10 CFR Part 50 Appendix B. These Regulatory Guides are available from the US NRC website (www.nrc.gov). Some examples include:

— Regulatory Guide 1.28, Quality Assurance Program Criteria (Design and Construction) [B52]

— Regulatory Guide 1.33, Quality Assurance Program Requirements (Operation) [B53]

— Regulatory Guide 1.152, Criteria for use of Computers in Safety Systems of Nuclear Power Plants [B54]

— Regulatory Guide 1.168, Verification, Validation, Reviews, and Audits for Digital Computer Software used in Safety Systems of Nuclear Power Plants [B55]

— Regulatory Guide 1.169, Configuration Management Plans for Digital Computer Software Used in Safety Systems of Nuclear Power Plants [B56]

— Regulatory Guide 1.170, Test Documentation for Digital Computer Software Used in Safety Systems of Nuclear Power Plants [B57]

— Regulatory Guide 1.171, Software Unit Testing for Digital Computer Software Used in Safety Systems of Nuclear Power Plants [B58]

— Regulatory Guide 1.172, Software Requirement Specifications for Digital Computer Software and Complex Electronics Used in Safety Systems of Nuclear Power Plants [B59]

— Regulatory Guide 1.173, Developing Software Life-Cycle Processes for Digital Computer Software Used in Safety Systems of Nuclear Power Plants [B60]

E.2.1.3.2 Technical reports

The US NRC has published technical reports and other documents that provide additional guidance on established industry practices. Some examples include:

— U.S. Nuclear Regulatory Commission Standard Review Plan, Branch Technical Position 7-14 — Guidance On Software Reviews For Digital Computer-Based Instrumentation And Control Systems (Section B.3.1.3 Software Quality Assurance Plan (SQAP) is applicable) [B61]

— NUREG/CR-6101, Software Reliability and Safety in Nuclear Reactor Protection Systems (Section 3.1.2 and Section 4.1.2 are related to SQA plans) [B62]

— NUREG/CR-6421, A Proposed Acceptance Process for Commercial Off-the-Shelf (COTS) Software in Reactor Applications [B63]

— NUREG/CR-6430, Software Safety Hazard Analysis [B64]

— NUREG/CR-6463, Review Guidelines on Software Languages for use in Nuclear Power Plant Safety Systems – Final Report [B65]

E.2.1.3.3 Electric Power Research Institute technical reports

The EPRI has published technical reports that provide additional guidance on established industry practices. Some examples include:

— EPRI NP-5652, Guideline for the Utilization of Commercial Grade Items in Nuclear Safety Related Applications [B10]

— EPRI NP-6406, Guidelines for the Technical Evaluation of Replacement Items in Nuclear Power Plants [B11]

— EPRI TR-1025243 Plant Engineering: Guideline for the Acceptance of Commercial-Grade Design and Analysis Computer Programs Used in Nuclear Safety-Related Applications [B12]. This guideline explicitly refers to IEEE Std 730 as an example of a standard for a documented and effective quality assurance program.

— EPRI TR-103291-CD, Handbook for Verification and Validation of Digital Systems [B13]. This handbook explicitly refers to industry standards such as IEEE Std 730, IEEE Std 1012, IEEE Std 1028™, and IEEE Std 1298™ for guidance with respect to good SQA practices.

— EPRI TR-106439 1996, Guideline on Evaluation and Acceptance of Commercial Grade Digital Equipment for Nuclear Safety Applications [B14]

E.2.1.4 Examples of standards

Several organizations have developed standards for the nuclear power industry. These organizations include the IEEE Nuclear Power Engineering Committee, the IEEE Computer Society, the American Society of Mechanical Engineers (ASME) Board of Nuclear Codes and Standards, and the American Nuclear Society (ANS).

This section discusses two of the most widely applied standards in the nuclear power generation industry. ASME NQA-1-2008 [B1], with ASME NQA-1a-2009 [B2] (together referred to as "NQA-1"), describes the requirements for a nuclear quality assurance program, including several sections that address software specifically. IEEE Std 7-4.3.2™-2010 [B24] discusses criteria for computers in nuclear power generating stations.

The activities in IEEE Std 730-2014 may be used to meet all or part of the requirements in the standards below.

— ASME NQA-1 [B1] [B2]. The following parts of the standard apply to the development and use of software in nuclear facilities:
  — Requirement 3, Sections 100, 400, 800, and 900
  — Requirement 11, Sections 100, 200, 400, 500, and 600
  — Subpart 2.7, Quality Assurance Requirements for Computer Software for Nuclear Facility Applications
  — Subpart 2.14, Quality Assurance Requirements for Commercial Grade Items and Services
  — Subpart 4.1, Application Appendix: Guide on Quality Assurance Requirements for Computer Software
— IEEE Std 7-4.3.2-2010, IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations [B24]. This standard explicitly calls out IEEE Std 730.

Annex F

(informative)

SQA activities and their relationship to the agile development process

F.1 Introduction

Agile methods, such as Scrum, Extreme Programming, Dynamic Systems Development Methodology (DSDM), Adaptive Software Development, Lean Methodology, and Feature Driven Development (FDD) are approaches to building software to adapt to rapidly changing customer requirements. These methods provide software suppliers the ability to respond in an agile manner.

These approaches typically follow the 12 principles below (See www.agilemanifesto.org/principles.html for more information):

1) Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2) Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3) Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4) Business people and developers work together daily throughout the project.
5) Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6) The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7) Working software is the primary measure of progress.
8) Agile processes promote sustainable development. The sponsors, developers, and users are expected to be able to maintain a constant pace indefinitely.
9) Continuous attention to technical excellence and good design enhances agility.
10) Simplicity—the art of maximizing the amount of work not done—is essential.
11) The best architectures, requirements, and designs emerge from self-organizing teams.
12) At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile approaches include but are not limited to the following elements:

— Burndown charts
— Collaborative development
— Collective code ownership
— Continuous feedback
— Continuous integration

— Customer involvement
— Pair programming
— Refactoring
— Small development teams
— Small releases
— Sprint/timeboxes
— Test-driven development

F.2 Modifying the 16 SQA activities to accommodate agile software development

When implementing an agile software development process, the 16 software quality assurance (SQA) activities in IEEE Std 730-2014 could be modified according to Table F.1 below. Note that standard conformance, regulatory concerns, or customer requirements are to be included in any tailoring for agile.

Table F.1—SQA activities and their adaptation to the agile development process

Subclause	Task	Adaptation to Agile
5.3	**SQA process implementation activities**	
5.3.1	Establish the SQA processes	In an agile environment, the "contract" is represented by the backlog, which is continually updated by mutual agreement between the software producer and customer according to a defined incremental process. The SQA process itself needs to be flexible, with as-needed updates to the SQA Plan to reflect current events. The "elements required by the contract" are placed in the backlog. In addition to software products, the "elements required by the contract" include artifacts such as plans, documentation, test artifacts, and others as specified in the SQAP and project management plans, which agile teams typically address as items in the backlog to be independently estimated and prioritized by the customer. Agile teams tailor the process, mostly through retrospectives at iteration completions, to fit the project's plan. Successful agile projects incorporate SQA and testers as collaborators within the iteration process itself, rather than having them as a separate and subsequent activity to development. SQA elements include test-driven development, tester, and customer inclusion in development teams, continual integration, automated builds, regression testing. Requirements are embodied and managed within the backlog. They are generally expressed as defined statements (usually outlined as brief user stories), which include acceptance criteria to demonstrate that the user story has been satisfied or confirmed. The user stories are carried out in time-boxed iterations and releases, can be reasonably accurately estimated relative to similar efforts, and can be clearly tested with a result of "done" completely or "not done." Note that those involved in an agile project may resist SQA, believing it to represent gatekeepers and bottlenecks. While that is not the case—see especially the 9th and 12th principles above—this resistance needs to be proactively addressed in many agile environments.

Table F.1—SQA activities and their adaptation to the agile development process (continued)

Subclause	Task	Adaptation to Agile
5.3.2	Coordinate with related software processes	The main process with which to coordinate is the agile software development process itself. The primary related software processes that SQA coordinates with are Verification, Validation, Review, and Audit. In an agile environment, all of these processes are integrated within each iteration in a collaborative process. For example, 1) Verification and validation can generate the tests used for Test-Driven Development; 2) Unit and integration testing can be provided by the observing member of a programming pair; 3) Reviews can be done on a daily basis; and 4) Audits can be part of the end-of-iteration retrospective. A narrow focus of agile projects can miss or disregard significant software processes, such as business analysis, project management, and SQA, which typically are more evident and explicitly related to development in other methodologies. These processes need to be involved in either the iteration team, the product-owner team in charge of the backlog, or both.
5.3.3	Document SQA planning	The product assurance portion of the SQAP assures that there is a testable definition of "done" for each of the requirement elements (i.e., items in the backlog). The process assurance portion of the SQAP aims to assure that agile values and principles (see the 12 principles above) are applied throughout the established process to ensure process effectiveness and efficiency. In some agile methods, this is the responsibility of a defined role. For example, in Scrum, the Scrum Master is responsible for ensuring that the Scrum process is followed. The SQA Plan for an agile project is tailored for products and processes that may be more incremental and changeable. Therefore, the SQAP itself becomes an evolving document.
5.3.4	Execute the SQA Plan	Because the SQAP is addressing an iterative process, the SQAP may be tailored at the beginning of each iteration to reflect the project's unique circumstances.
5.3.5	Manage SQA records	Any non-conformances raised may be inserted as an item in the product backlog and prioritized by the product owner. Some suggestions are that the records created by agile iteration teams be easy to generate (e.g., taking a digital picture of the whiteboard on which progress is tracked), with more formal records being the responsibility of the product owner/project manager team.
5.3.6	Evaluate organizational independence and objectivity	Local team-specific evaluations of objectivity and independence may be carried out by the team through retrospectives. Project and enterprise-wide evaluations would be carried out through the usual SQA audit process. Preserving the technical, managerial, and financial independence of SQA can be a challenge in an agile environment due to its team-centric and informal nature. Dotted-line relationships to the organization's quality or senior management may address this. As a minimum, the project needs to carry out the objectivity and independence standards required by the industry in which it is engaged.
5.4	**Product assurance activities**	
5.4.2	Evaluate plans for conformance to contracts, standards, and regulations	In an agile environment, the plans tend to be more informal, with emphasis on frequent delivery of working software, strong customer involvement, and robust reaction to change. The strong customer involvement will be leveraged to assess conformance of the plans.

Table F.1—SQA activities and their adaptation to the agile development process (continued)

Subclause	Task	Adaptation to Agile
5.4.3	Evaluate product for conformance to established requirements	Individual backlog items are determined as "done" (conforms to requirements) or "not done" (does not conform to requirements). SQA evaluates whether the processes that determine whether items are "done" or not are indeed effective. The strong customer involvement will be leveraged to assess conformance of the product. The testing function plays an important role in determining software product conformity and in the development of the software products themselves.
5.4.4	Evaluate product for acceptability	This is a continual process in agile, not just at final delivery. Agile methods tend to identify necessary subsequent product changes as "refactoring," which demonstrates agile development's flexibility and reaction to change. As part of determining that user stories are "done," the team completes the tasks listed as SQA activities and SQA is to be part of the team. At the end of the project, the customer indicates that all of the accepted deliveries constitute acceptable completion of the project.
5.4.5	Evaluate product life cycle support for conformance	Agile methodology provides an in-team SQA coach that produces value with the team. The customer is expected to commit their own resources to provide support and feedback.
5.4.6	Measure products	Measures are to focus on the development (iteration) teams and overall project management. Measures focused on the development teams (e.g., burn-down charts, test cases and results, burn-up charts) track progress within the iteration itself, while measures focused on project management (e.g., changes in project backlog by iteration) track progress on the overall product backlog, interactions among product components, overall quality, and customer satisfaction. In an agile project, all team members provide continual feedback. Product and process measures are updated and available on a continual basis. To the extent that the product and process measures are collected by the development teams, it is necessary to have tools and processes that allow the development teams to capture the data easily and quickly.
5.5	**Process assurance activities**	
5.5.2	Evaluate life cycle processes and plans for conformance	Additional processes include the backlog administration process, continuous integration, test inspection, and automation of integration and regression tests. In agile, these SQA activities are processes.
5.5.3	Evaluate environments for conformance	In the agile software development life cycle process, the evaluation of the software development and test environments against the project plans are to be done on a continual basis, since both environments may need to change.
5.5.4	Evaluate subcontractor processes for conformance	In agile methods, the project management and staff including SQA are customers of the subcontractor and therefore participate in the project more closely. SQA plays a similar role that the acquirer's SQA plays in the main project itself.
5.5.5	Measure processes	Same adaptation steps listed for 5.4.6, Measure products
5.5.6	Assess staff skill and knowledge	No change, except that "staff" includes the customer or customer representative.

Annex G

(informative)

Mapping between IEEE Std 730-2014 and ISO/IEC 29110 standard for Very Small Entities

This annex presents a description of the coverage of the tasks of IEEE Std 730-2014 by the Basic profile of ISO/IEC 29110:2011 [B40] standard.

ISO/IEC 29110:2011 standards and technical reports are targeted at Very Small Entities. A Very Small Entity (VSE) is defined as an enterprise, organization, department or project having up to 25 people [B40]. The ISO/IEC 29110:2011 is a series of international standards and technical reports entitled "Software Engineering — Lifecycle profiles for Very Small Entities." A collection of four profiles provides a progressive approach to satisfying a vast majority of the needs of VSEs involved in the development of non-critical software.

From studies and surveys conducted [B45] [B46], it is clear that the majority of international standards do not address the needs of VSEs. Conformance with these standards is difficult, if not impossible, giving VSEs no way, or very limited ways, to be recognized as entities that produce quality software in their domain. Therefore, VSEs are often cut off from some economic activities.

In addition, it has been found that VSEs find it difficult to relate international standards to their business needs and to justify their application to their business practices. Most VSEs can neither afford the resources, in terms of number of employees, budget, and time, nor do they see a net benefit in establishing software life cycle processes. To rectify some of these difficulties, a set of standards and technical reports (e.g., guides) have been developed according to a set of VSE characteristics.

The ISO/IEC 29110:2011 standards are based on subsets of appropriate standards elements, referred to as profiles. The purpose of a profile is to define a subset of international standards relevant to the VSE context (e.g., processes and outcomes of ISO/IEC/IEEE 12207:2008 and information items of ISO/IEC/IEEE 15289:2011). For VSEs developing non-critical software, a set of four profiles have been developed: Entry, Basic, Intermediate, and Advanced. The Entry profile is suitable for a start-up VSE that started the business less than three years ago, or a project with a size of less than six person-months. The Basic profile describes software development practices of a single application by a single project team with no special risk or situational factors. The Intermediate profile is targeted at VSEs developing multiple projects within the organizational context taking advantage of it. The Advanced profile is targeted to VSEs that want to sustain and grow as an independent competitive software development business.

All profiles provide a Software Implementation process and a Project Management process. The purpose of the Project Management process is to establish and carry out in a systematic way the tasks of the software implementation process, which allows complying with the project's objectives in the expected quality, time, and cost. Table G.1 lists the objectives of the Project Management process. Most software quality assurance (SQA) activities are covered by "PM.O7: Software Quality Assurance is performed to provide assurance that work products and processes comply with the Project Plan and Requirements Specification." The implementation of SQA is through the performance of verifications, validations, and review tasks described in the project management process and the software implementation process.

Table G.1—Objectives for the Project Management process of the Basic profile of ISO/IEC 29110:2011

Identification of the objective	Description of the objective
PM.O1	The Project Plan for the execution of the project is developed according to the Statement of Work and validated with the Customer. The tasks and resources necessary to complete the work are sized and estimated.
PM.O2	Progress of the project is monitored against the Project Plan and recorded in the Progress Status Record. Corrections to remediate problems and deviations from the plan are taken when project targets are not achieved. Closure of the project is performed to get the Customer acceptance documented in the Acceptance Record.
PM.O3	The Change Requests are addressed through their reception and analysis. Changes to the software requirements are evaluated for cost, schedule and technical impact.
PM.O4	Review meetings with the Work Team and the Customer are held. Agreements are registered and tracked.
PM.O5	Risks are identified as they develop and during the conduct of the project.
PM.O6	A software Version Control Strategy is developed. Items of Software Configuration are identified, defined and baselined. Modifications and releases of the items are controlled and made available to the Customer and Work Team including the storage, handling and delivery of the items.
PM.O7	Software Quality Assurance is performed to provide assurance that work products and processes comply with the Project Plan and Requirements Specification.

The purpose of the Software Implementation process is the systematic performance of the analysis, design, construction, integration, and tests activities for new or modified software products according to the specified requirements. Table G.2 lists the objectives of the Software Implementation Process of the Basic profile of ISO/IEC 29110:2011.

Table G.2—Objectives for the Software Implementation process of the Basic profile of ISO/IEC 29110:2011

Identification of the objective	Description of the objective
SI.O1	Tasks of the activities are performed through the accomplishment of the current Project Plan.
SI.O2	Software requirements are defined, analyzed for correctness and testability, approved by the Customer, baselined, and communicated.
SI.O3	Software architectural and detailed design is developed and baselined. It describes the software items and internal and external interfaces of them. Consistency and traceability to software requirements are established.
SI.O4	Software components defined by the design are produced. Unit test are defined and performed to verify the consistency with requirements and the design. Traceability to the requirements and design are established.
SI.O5	Software is produced performing integration of software components and verified using Test Cases and Test Procedures. Results are recorded at the Test Report. Defects are corrected and consistency and traceability to Software Design are established.

Table G.2—Objectives for the Software Implementation process of the Basic profile of ISO/IEC 29110:2011 (continued)

Identification of the objective	Description of the objective
SI.O6	A Software Configuration, that meets the Requirements Specification as agreed to with the Customer, which includes user, operation, and maintenance documentations is integrated, baselined, and stored at the Project Repository. Needs for changes to the Software Configuration are detected and related Change Requests are initiated.
SI.O7	Verification and Validation tasks of all required work products are performed using the defined criteria to achieve consistency among output and input products in each activity. Defects are identified and corrected; records are stored in the Verification/Validation Results.

Table G.3 shows coverage of the Project Management and Software Implementation activities and tasks of the Basic profile of ISO/IEC 29110:2011 against IEEE Std 730-2014. An assumption is that a VSE has fully implemented the Project Management and Software Implementation activities and tasks of the Basic profile. The coverage is indicated in Table G.3 using the following convention:

— Full Coverage = F

— Partial Coverage = P

— No Coverage = N

Table G.3—Coverage of the Project Management and Software Implementation activities and tasks of the Basic profile of ISO/IEC 29110:2011 against IEEE Std 730-2014

IEEE Std 730-2014 subclause	Activities and Tasks	Coverage (F/P/N)	ISO/IEC 29110:2011 Basic Profile	Comments
5.3	**SQA process implementation activities**			
5.3.1	Establish the SQA processes	P	There is no specific SQA process in the Basic profile. Objective PM.O7.	The SQA tasks are embedded in the project management and software implementation processes.
5.3.2	Coordinate with related software processes	F	There are only 2 processes: project management and software implementation Objectives PM.O1 and SI.O1.	There are interfaces between the two processes. Verification and validation tasks are embedded in the two processes. Roles are defined and members of team are assigned role(s).
5.3.3	Document SQA planning	P	There is no SQA plan. Objective PM.O1.	The SQA tasks and embedded in the project management and the software implementation processes. SQA tasks are planned and described in the project plan.

Table G.3—Coverage of the Project Management and Software Implementation activities and tasks of the Basic profile of ISO/IEC 29110:2011 against IEEE Std 730-2014 (continued)

IEEE Std 730-2014 subclause	Activities and Tasks	Coverage (F/P/N)	ISO/IEC 29110:2011 Basic Profile	Comments
5.3.4	Execute the SQA Plan	P	There is no SQA plan. Objectives PM.O2, PM.O3, PM.O4, and SI.O7.	SQA tasks are executed in the project management and the software implementation processes.
5.3.5	Manage SQA records	F	Objectives PM.O2, PM.O3, PM.O4, PM.O7, and SI.O7.	
5.3.6	Evaluate organizational independence and objectivity	N		Technical, financial, managerial independence are not met.
5.4	**Product assurance activities**			
5.4.1	Defining product assurance	P	Objectives PM.O1, PM.O2, PM.O3, PM.O4, PM.O6, PM.O7, SI.O6, and SI.O7.	Tasks to evaluate the project plan and products are defined in the processes. There is no SQA (department, group, person) specified in the processes.
5.4.2	Evaluate plans for conformance to contracts, standards, and regulations	P	Objectives PM.O1, PM.O2, PM.O3, PM.O4, PM.O6, PM.O7, SI.O6, and SI.O7.	Evaluation of the project plan and products are defined in the processes. There is no SQA (department, group, person) specified in the processes.
5.4.3	Evaluate product for conformance to established requirements	P	Objectives PM.O1, PM.O2, PM.O7, SI.O6, and SI.O7.	There is no SQA (department, group, person) specified in the processes.
5.4.4	Evaluate product for acceptability	P	Objectives PM.O1, PM.O2, PM.O7, SI.O6, and SI.O7.	There is no SQA (department, group, person) specified in the processes.
5.4.5	Evaluate product life cycle support for conformance	P	Objective PM.O2.	There is no SQA (department, group, person) specified in the processes.
5.4.6	Measure products	P	Objective PM.O1, PM.O2, and SI.O1.	There is no SQA (department, group, person) specified in the processes.
5.5	**Process assurance activities**			
5.5.1	Defining process assurance	N		There is no SQA (department, group, person) specified in the processes.
5.5.2	Evaluate life cycle processes and plans for conformance	P	Objective PM.O2.	There is no SQA (department, group, person) specified in the processes. The evaluation tasks are embedded in the project management and software implementation processes. The contract is not used to verify compliance. The statement of work is used instead.

Table G.3—Coverage of the Project Management and Software Implementation activities and tasks of the Basic profile of ISO/IEC 29110:2011 against IEEE Std 730-2014 (continued)

IEEE Std 730-2014 subclause	Activities and Tasks	Coverage (F/P/N)	ISO/IEC 29110:2011 Basic Profile	Comments
5.5.3	Evaluate environments for conformance	P	Objective PM.O2.	There is no SQA (department, group, person) specified in the processes.
5.5.4	Evaluate subcontractor processes for conformance	N		For VSEs, it is assumed, for the Basic profile, that no work would be subcontracted.
5.5.5	Measure processes	P	Objective PM.O1, PM.O2, and SI.O1.	There is no SQA (department, group, person) specified in the processes.
5.5.6	Assess staff skill and knowledge	N		One assumption for this profile is: the project working team, including project manager, is trained.

Annex H

(informative)

Software tool validation

In some regulated industries such as nuclear power and medical devices, software tools used to develop and test software may need to be validated based on their intended use. This requirement exists because software tools are just as likely to have defects as any other software. This requirement applies to off-the-shelf software tools (both purchased and open source) as well as software tools developed by the software supplier for their own internal use.

Such development tools are to be validated in a manner consistent with the overall software integrity level. Only those features of the tool that are used on the project need to be validated. The validation is to be based on published documentation provided by the tool supplier (for purchased and open source tools) or tool requirements specifications (for tools developed in-house). The degree of validation (breadth and depth) is to be commensurate with software integrity level.

In some cases, explicit tool validation may not be needed for a specific tool if it can be shown that the tool function is not directly related to safety-critical aspects of the project. In addition, tool validation for a specific tool may not be needed if there is a downstream process that would catch a defect injected by the tool into the software product. For example, a Computer Aided Design (CAD) drawing tool is useful in preparing flowcharts. The flowchart itself is something that can be reviewed and determined to be correct. Therefore, the CAD tool that creates the flowchart does not need to be validated.

The rationale for not validating a tool needs to be documented.

The project is to identify and maintain a list of all software tools used for both development and testing. The list includes the following minimum information:

- Tool name and version number
- Tool supplier
- Tool users [e.g., development, software quality assurance (SQA), test]
- Tool purpose
- Tool features used
- Tool validation status (i.e., validated, to be validated, not validated—justification)
- Tool validation records (pointer to where validation records are stored)
- Release notes (location of release notes for tool, if applicable)

Exception: Implicit validation of compilers is performed using means that may include testing and several other activities described below. Testing of the application implicitly validates compilers for their intended use. Explicit testing of compilers is not required. Specific validation activities for compilers include the following tasks:

- Compiler supplier qualification.

  Compiler supplier qualification provides confidence that the compiler supplier has a robust development process and provides information regarding known compiler problems to their customers in a timely manner. The Development Team will identify all compilers used on projects.

This identification will include supplier name, location, contact information, and the specific version/release number of the compiler(s) that have been and are being used.

— Review Compiler Bug List.

The Development Team is responsible for obtaining and reviewing, on a regular basis, current bug lists that are specific to the compiler(s) being used to develop software. The Development Team may also use other means, such as Compiler User Groups, etc. to obtain factual information regarding potential compiler bugs and related information.

— Identify Compiler Features to Avoid.

Based on reviewing compiler bug list, collective development experience using the compiler(s), and any other sources of factual information, the Software Development Team will identify and document those compiler features that the Development staff is to avoid using in order to ensure that those defects do not adversely affect the software under development.

— Compiler Change Control.

In general, the version of a compiler is not to be changed once development begins. This reduces the likelihood of introducing defects that result from bugs in the compiler.

— Maintenance of Compiler Versions.

Once a release is deployed, the specific version of the compiler and related development environment tools used to develop that release are archived and preserved such that Development could re-create, if necessary, the environment used to develop that release. Specifically, if the need arose to provide a field update for a released version, the exact set of development tools that were used to create that release are available to Development staff. This requirement pertains to all releases of software that are currently in active use in the field. Development and V&V together are jointly responsible for ensuring that this requirement is met.

The requirement for tool validation in the medical device industry can be found in:

— General Principles of Software Validation, Final Guidance for Industry and FDA Staff , January 2002 [B16]

The requirements for tool validation in the nuclear power industry can be found in:

— IEEE Std 7-4.3.2™-2010, Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations [B24]

— ASME NQA-1-2008, Quality Assurance Requirements for Nuclear Facility Applications [B1]

— ASME NQA-1a-2009, Addenda to ASME NQA-1-2008, Quality Assurance Requirements for Nuclear Facility Applications [B2]

Annex I

(informative)

Assessing product risk: Software integrity levels and assurance cases

I.1 Software integrity levels

Software integrity levels are a range of values that represent software complexity, criticality, risk, safety level, security level, desired performance, reliability, or other project-unique characteristics that define the importance of the software to the user and acquirer. The characteristics used to determine software integrity level vary depending on the intended application and use of the system. The software is a part of the system, and its integrity level is to be determined as a part of that system.

The assigned software integrity levels may change as the software evolves. Design, coding, procedural, and technology features implemented in the system or software can raise or lower the assigned software integrity levels. The software integrity levels established for a project will result from agreements among the acquirer, supplier, developer, and independent assurance authorities (e.g., a regulatory body or responsible agency).

A software integrity level scheme is a tool used in determining software integrity levels. IEEE Std 1012-2012 [B26] provides an example of a four-level scheme, shown in Table I.1 below.

Table I.1—Example of four-level software integrity level scheme

Level	Description
4	Software element executes correctly or grave consequences (loss of life, loss of system, economic, or social loss) will occur. No mitigation is possible.
3	Software element executes correctly or the intended use (mission) of the system/software will not be realized, causing serious consequences (permanent injury, major system degradation, economic, or social impact). Partial to complete mitigation is possible.
2	Software element executes correctly or an intended function will not be realized, causing minor consequences. Complete mitigation possible.
1	Software element executes correctly or intended function will not be realized, causing negligible consequences. Mitigation not required.

A system integrity level can be assigned to an entire system, inclusive of software. In addition, software integrity levels can be applied to individual elements or components of the system. A risk-based approach is used to define an appropriate set of integrity levels for a given system and elements as illustrated in Figure I.1 below.

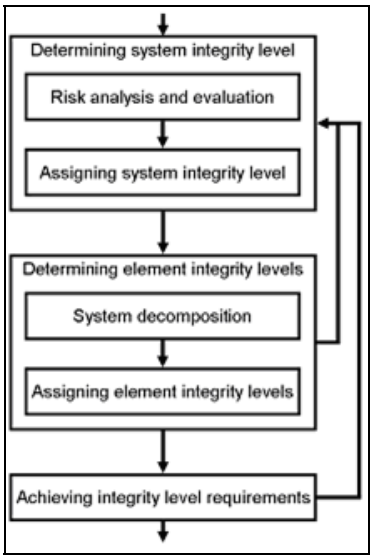

Figure I.1—SQA activities assigned based on integrity level

The four integrity levels in Table I.2, below, are examples, not normative. An organization can choose the number of integrity levels to suit their own purposes. Organizations can use this chart to cross-reference IEEE Std 730-2014 tasks by modifying the number of columns for integrity levels.

Table I.2—Example of integrity level mapping to IEEE Std 730-2014 tasks

	1 (Lowest)	2	3	4 (Highest)
SQA process implementation activities				
Establish the SQA processes			x	x
Coordinate with related software processes			x	x
Document SQA planning	x	x	x	x
Execute the SQA Plan	x	x	x	x
Manage SQA records	x	x	x	x
Evaluate organizational independence and objectivity			x	x
Product assurance activities				
Evaluate plans for conformance to contracts, standards, and regulations			x	x
Evaluate product for conformance to established requirements			x	x
Evaluate product for acceptability		x	x	x
Evaluate product life cycle support for conformance			x	x
Measure products			x	x

Table I.2—Example of integrity level mapping to IEEE Std 730-2014 tasks (continued)

	1 (Lowest)	2	3	4 (Highest)
Process assurance activities				
Evaluate life cycle processes and plans for conformance			x	x
Evaluate environments for conformance			x	x
Evaluate subcontractor processes for conformance			x	x
Measure processes			x	x
Assess staff skills and knowledge			x	x

SQA tasks based on integrity levels are shown in Table I.3, below.

Table I.3—SQA Tasks Based on integrity levels

	1 (Lowest)	2	3	4 (Highest)
PROCESS IMPLEMENTATION TASKS				
Establish SQA role			x	x
Establish SQA policy/processes			x	x
Identify SQA organizational interfaces			x	x
Prepare SQA plan		x	x	x
Manage SQA effort		x	x	x
PROCESS TASKS				
Create SQA records/forms			x	x
PRODUCT TASKS				
Evaluate project plans			x	x
Evaluate supplier plans/documents			x	x
Review acceptance criteria			x	x
Review product measurements and measurements procedures			x	x
PROCESS TASKS				
Perform activity audits			x	x
Perform project reviews			x	x
Perform supplier process reviews/audits			x	x

I.2 Overview of activities for integrity level determination

The degree of rigor and intensity in performing software quality assurance (SQA) activities and tasks is commensurate with the software integrity level. As the software integrity level changes, so does the required scope, intensity, and degree of rigor associated with SQA tasks.

SQA proposes an integrity level scheme if one is not already defined for the project. The integrity levels established for a project will result from agreements among the acquirer, supplier, developer, and independent assurance authorities (e.g., a regulatory body or responsible agency).

The integrity level assigned to reused, commercial off the shelf (COTS), and government off the shelf (GOTS) components will be in accordance with the integrity level scheme adopted for the project, and the reused COTS or GOTS component will be evaluated for use in the context of its application. Design, development, procedural, and technology features implemented in the system can raise or lower the assigned integrity levels.

Tools that insert or translate code (e.g., optimizing compilers, auto-code generators) will be assigned the same integrity level as the integrity level assigned to the element that the tool affects.

The mapping of the integrity level scheme and the associated minimum SQA tasks will be documented in the SQA Plan. The basis for assigning integrity levels to components will be documented in the appropriate report in accordance with agreements and legal, regulatory, or product sector requirements.

The integrity level assignment will be continually reviewed and updated throughout the life cycle. If the integrity level is revised, the effect of the revision on existing requirements is evaluated to identify additional activities and tasks to be performed regressively on the system, software, and hardware. Corresponding SQA activities and tasks are identified to assure that the proper SQA effort is applied based on its revised integrity level.

This standard recommends the use of software integrity levels for projects where the risk of adverse effects is of concern to project stakeholders. While not explicitly required by this standard, use of integrity levels may be required by regulatory bodies in safety-critical industries.

Software integrity level schemes are often required in safety-critical industries such as nuclear power, medical devices, etc. Safety-critical industries require compliance with industry regulations and use of industry-specific standards. Safety-critical industries have their own unique terms for defining risk and integrity levels. Examples of software integrity levels in several safety-critical industries are included in Table I.4 below. For some industries, safety classified systems are systems defined by the regulator as having safety significance.

Table I.4—Summary of terms used for safety-critical software

Industry	Standard or reference	Terms used for safety critical software
Avionics	RTCA DO-178B, Software Considerations in Airborne Systems and Equipment Certification [B51]	DO-178B defines five software levels. Each level is defined by the failure condition that can result from anomalous software behavior. Failure Condition — Software Level Catastrophic — Level A Hazardous/Severe — Level B Major — Level C Minor — Level D No Effect — Level E
Nuclear Power	IEEE Std 7-4.3.2-2010, Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations [B24]	Software integrity levels. "The software V&V effort shall be performed in accordance with IEEE Std 1012-2012. The IEEE Std 1012-2012 V&V requirements for the highest integrity level (software integrity level 4) apply to systems developed using this standard (i.e., IEEE Std 7-4.3.2). See IEEE Std 1012-2012 Annex B for a definition of integrity level 4."
Nuclear Power	IEC 61513, Nuclear power plants — Instrumentation and control important to safety — General requirements for systems [B23]	Important to Safety The standard provides requirements and recommendations for the instrumentation and control for systems important to safety of nuclear power plants.
Nuclear Power	EPRI 1025243, Plant Engineering: Guideline for the Acceptance of Commercial Grade Design and Analysis Computer Programs Used in Nuclear Safety-Related Applications [B12]	Software Safety Classification EPRI 1025243 provides for safety classification by Failure Modes and Effects or by Impact Categorization.
Medical Devices	ANSI/AAMI/IEC Standard 62304:2006, Medical device Software — software life cycle processes [B3]	Software Safety Classification: IEC Standard 62304:2006 software safety classification are based on severity as follows: Class A — No injury or damage to health is possible. Class B — Non-serious injury is possible. Class C — Death or serious injury is possible.

Table I.4—Summary of terms used for safety-critical software (continued)

<table>
<tr><th>Industry</th><th>Standard or reference</th><th>Terms used for safety critical software</th></tr>
<tr><td>Space Exploration</td><td>NASA Technical Standard 8739.8 2004, Software Assurance Standard [B47]</td><td>Software Class:
Class A Human Rated Software Systems
Class B Non-Human Space Rated
Class C Mission Support Software
Class D Analysis and Distribution Software
Class E Development Support Software</td></tr>
<tr><td>Transportation</td><td>IEC 61508, Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems [B22]

Rail

EN 50128, Railway applications – Software for railway control and protection systems [B6], provides a specific interpretation of IEC 61508 for railway applications.

Automotive Software

The development of software for safety related automotive systems is predominantly covered by the Motor Industry Software Reliability Association guidelines (MISRA). The MISRA project was conceived to develop guidelines for the creation of embedded software in road vehicle electronic systems. In November 1994, Development Guidelines for Vehicle Based Software was published. This document provides the first automotive industry interpretation of the principles of the emerging standard IEC 61508.</td><td>Safety Integrity level

The Safety Integrity level (SIL) is determined from the probability of failure. For systems that operate continuously the allowable frequency of failure is determined. For systems that operate intermittently the probability of failure is specified as the probability that the system will fail to respond on demand.

<table><tr><th>SIL</th><th>Low demand mode: average probability of failure on demand</th><th>High demand or continuous mode: probability of dangerous failure per hour</th></tr><tr><td>1</td><td>$\geq 10^{-2}$ to $< 10^{-1}$</td><td>$\geq 10^{-6}$ to $< 10^{-5}$</td></tr><tr><td>2</td><td>$\geq 10^{-3}$ to $< 10^{-2}$</td><td>$\geq 10^{-7}$ to $< 10^{-6}$</td></tr><tr><td>3</td><td>$\geq 10^{-4}$ to $< 10^{-3}$</td><td>$\geq 10^{-8}$ to $< 10^{-7}$</td></tr><tr><td>4</td><td>$\geq 10^{-5}$ to $< 10^{-4}$</td><td>$\geq 10^{-9}$ to $< 10^{-8}$</td></tr></table></td></tr>
</table>

Additional information on the definition and use of integrity levels can be found in ISO/IEC 15026-3:2013 [B38].

I.3 Assurance cases

In the report, Software for Dependable Systems—Sufficient Evidence?, by the Committee on Certifiably Dependable Software Systems of the National Research Council [B44], the Committee recommends an *evidence-based approach* for assessing and assuring dependability in software systems that argues for and justifies dependability claims based on explicit evidence supporting such arguments and claims.

"Software assurance is an important part of the software development process to reduce risks and ensure that the software is trustworthy. The critical importance of establishing and assuring dependability and

trustworthiness (e.g., safety, security, reliability, etc.) of systems and/or software in avionics, industrial control systems and other safety and mission-critical systems has long been recognized" [B49].

An assurance case can be defined as a documented body of evidence that provides a convincing and valid argument that a specified set of critical claims about a software system's properties are adequately justified for a given application in a given context.

An assurance case presents an argument that a software system (e.g., a combination of hardware and software) is acceptably safe, secure, reliable, in a given context. Experience with assurance cases has mainly been in the area of safety-critical systems. Such assurances cases have also been called safety cases.

An assurance case requires claims, evidence, and arguments linking evidence to claims:

— Claim: A statement regarding a critical characteristic (i.e., safety, security, reliability) of the system that is being asserted.

— Arguments: Explanations that can be reasonably interpreted as indicating that the critical characteristics are met, usually by demonstrating compliance with requirements, sufficient mitigation of hazards, or avoidance of hazards.

— Evidence: Results of observing, analyzing, testing, simulating, and estimating the properties of a system that provides fundamental information from which the presence of some system characteristic can be inferred.

An assurance case is a representation of a claim or claims, and the support for these claims. These claims can be the claims in which confidence is needed. The structure of an assurance case is shown in Table I.5.

Table I.5—Structure of an assurance case

Assurance Case Part	Explanation
Claim:	A claim being made about some aspect of a software system
Arguments:	Specific arguments supporting the claim: • Argument #1 • Argument #2 • … • Argument #n
Evidence:	Factual evidence (including reviews, records, and test results) that support each of the arguments. • Evidence #1 • Evidence #2 • … • Evidence #n

A software assurance case is *constructed* during the course of a software engineering project, in support of that project. Once constructed, an assurance case provides a degree of confidence that the execution of the software will not cause adverse results. The degree of confidence depends upon the breadth and depth of the assurance case. The higher the necessary degree of confidence, the broader and deeper the assurance case or the set of assurance cases will be.

I.3.1 Claims

A claim is a statement regarding some safety-critical aspect of a system that may be of concern to the project stakeholders. For example:

Example claim:	The software is safe for use in a *specified environment*, for a *specified intended use*.

Claims can be made at the system level or at the software level and are intended to provide a level of confidence in the developed system.

I.3.2 Arguments

An argument is a statement intended to demonstrate why the claim is true. For example:

Example Claim:	The software is safe for use in a *specified environment*, for a *specified intended use*.
Example Arguments:	Argument #1: • The System Requirements Specification was reviewed by qualified engineers and was determined to be correct. Argument #2: • The Software Requirements Specification was reviewed by qualified engineers and was determined to be correct. Argument #3: • Safety-critical source code was reviewed by qualified engineers against the Software Requirements Specification and coding guidelines and was determined to be correct. Argument #4: • All software requirements were tested by an independent test team.

I.3.3 Evidence

Evidence provides factual information that directly supports the arguments. For example,

Example Claim:	The software is safe for use in a *specified environment*, for a *specified intended use*.
Example Arguments:	Argument 1: • The System Requirements Specification was reviewed by qualified engineers and was determined to be correct. Argument 2: • The Software Requirements Specification was reviewed by qualified engineers and was determined to be correct. Argument #3: • Safety-critical source code was reviewed by qualified engineers against the Software Requirements Specification and coding guidelines and was determined to be correct. Argument #4: • All software requirements were tested by an independent test team.
Example Evidence:	The following evidence is available to support each of the arguments listed above: Evidence for Argument #1: • Meeting minutes for the System Requirements Review • Resolution of Corrective Actions taken for issues raised against System Requirements Specification Evidence for Argument #2: • Meeting minutes for the Software Requirements Review • Resolution of Corrective Actions taken for issues raised against Software Requirements Specification Evidence for Argument #3: • Meeting minutes for source code review • Resolution of Corrective Actions taken for issues raised during source code review Evidence for Argument #4: • Meeting minutes for all Technical Reviews • Test Execution Records and Requirements Trace Matrix showing all software requirements and tests performed against those requirements

This standard recommends the use of assurance cases for projects where risk of adverse effects is of concern to project stakeholders. While not explicitly required by this standard, use of assurance cases may be required by regulatory bodies in safety-critical industries.

Further details on assurance cases can be found in the following:

— ISO/IEC 15026-1 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary [B36]

— ISO/IEC 15026-2 Systems and software engineering — Systems and software assurance — Part 2: Assurance case[B37]

— ISO/IEC 15026-3 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels [B38]

— ISO/IEC 15026-4 Systems and software engineering — Systems and software assurance — Part 4: Assurance in the life cycle [B39]

Annex J

(informative)

Example corrective and preventive action process and root cause analysis process

Software quality assurance (SQA) documents non-conformances using a defined process. This process may be a process for resolving software problems defined in the project plan or a separate process documented in either the SQA Plan or the organizational Quality Management Plan.

Non-conformances are addressed by the project team using a defined Corrective Action (CA) process which may be documented in the project plan, the SQA Plan, or the organizational Quality Management Plan. In response to a non-conformance, the project team proposes a corrective action. SQA reviews each proposed corrective action to determine whether it addresses the associated non-conformance. If the proposed corrective action does address the non-conformance, SQA identifies appropriate effectiveness measures that determine whether a proposed corrective action is effective in resolving the non-conformance. Once a corrective action is implemented, SQA evaluates the related activity and determine whether the implemented corrective action is effective. An example of a CA Process is shown in Figure J.1.

A Root Cause Analysis (RCA) process is used to identify effective corrective actions. An example of an RCA process is shown in Figure J.2. The RCA Process can be defined either in the SQA Plan or in the organizational Quality Management Plan.

Preventive actions are taken to prevent occurrence of problems that may occur in the future. Non-conformances and other project information may be used to identify Preventive Actions as shown in Figure J.3. SQA reviews proposed preventive actions and identifies effectiveness measures. Once the preventive action is implemented, SQA evaluates the activity and determine whether the preventive action is effective. The Preventive Action (PA) Process can be defined either in the SQA Plan or in the organizational Quality Management Plan.

Further details on Root Cause Analysis can be found in the following:

— *Apollo Root Cause Analysis – A* New *Way of Thinking,* by Dean Gano [B15]

— *Root Cause Analysis – Basic Tools and Techniques*, by Denise Robitille [B50]

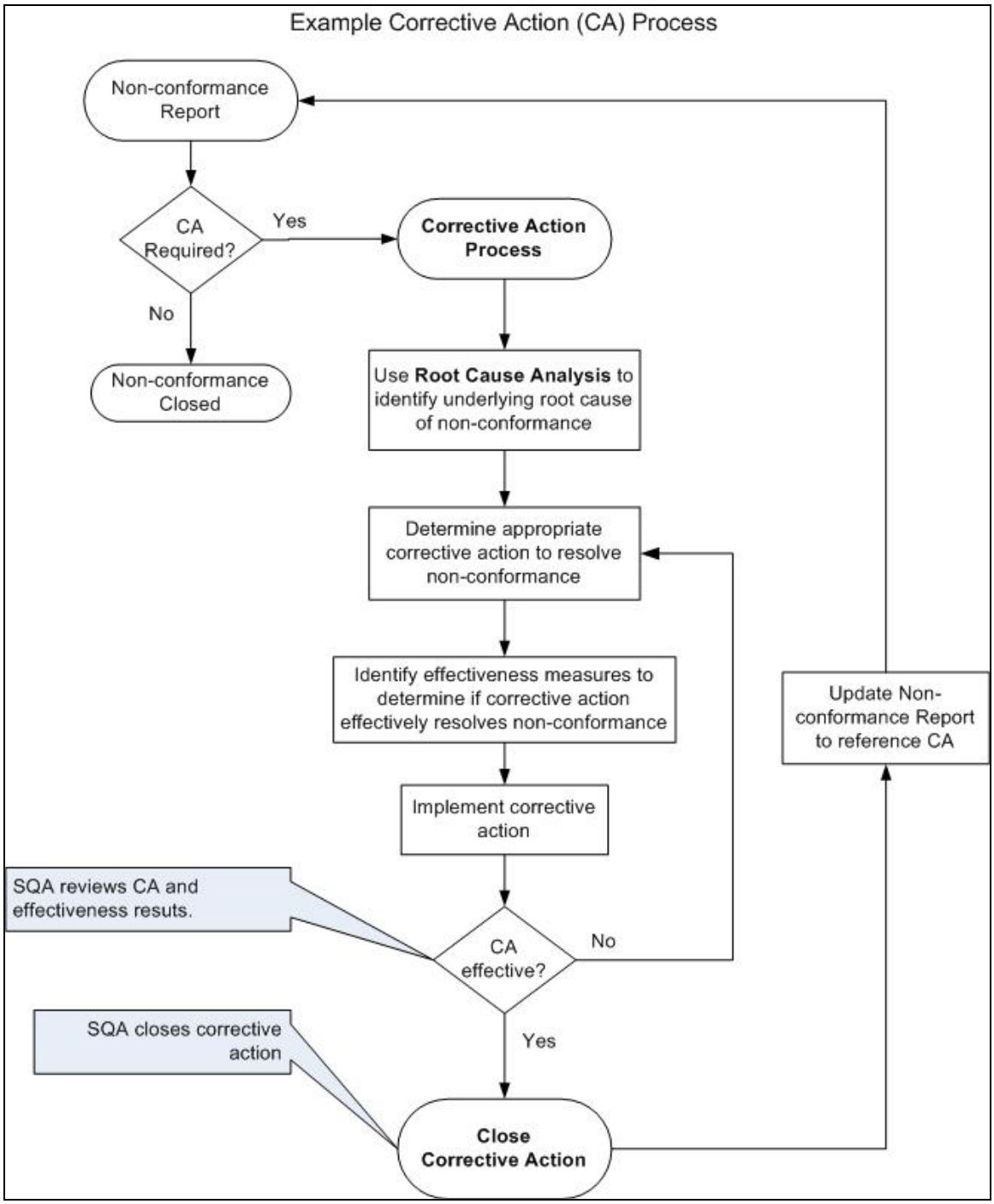

Figure J.1—Example corrective action process

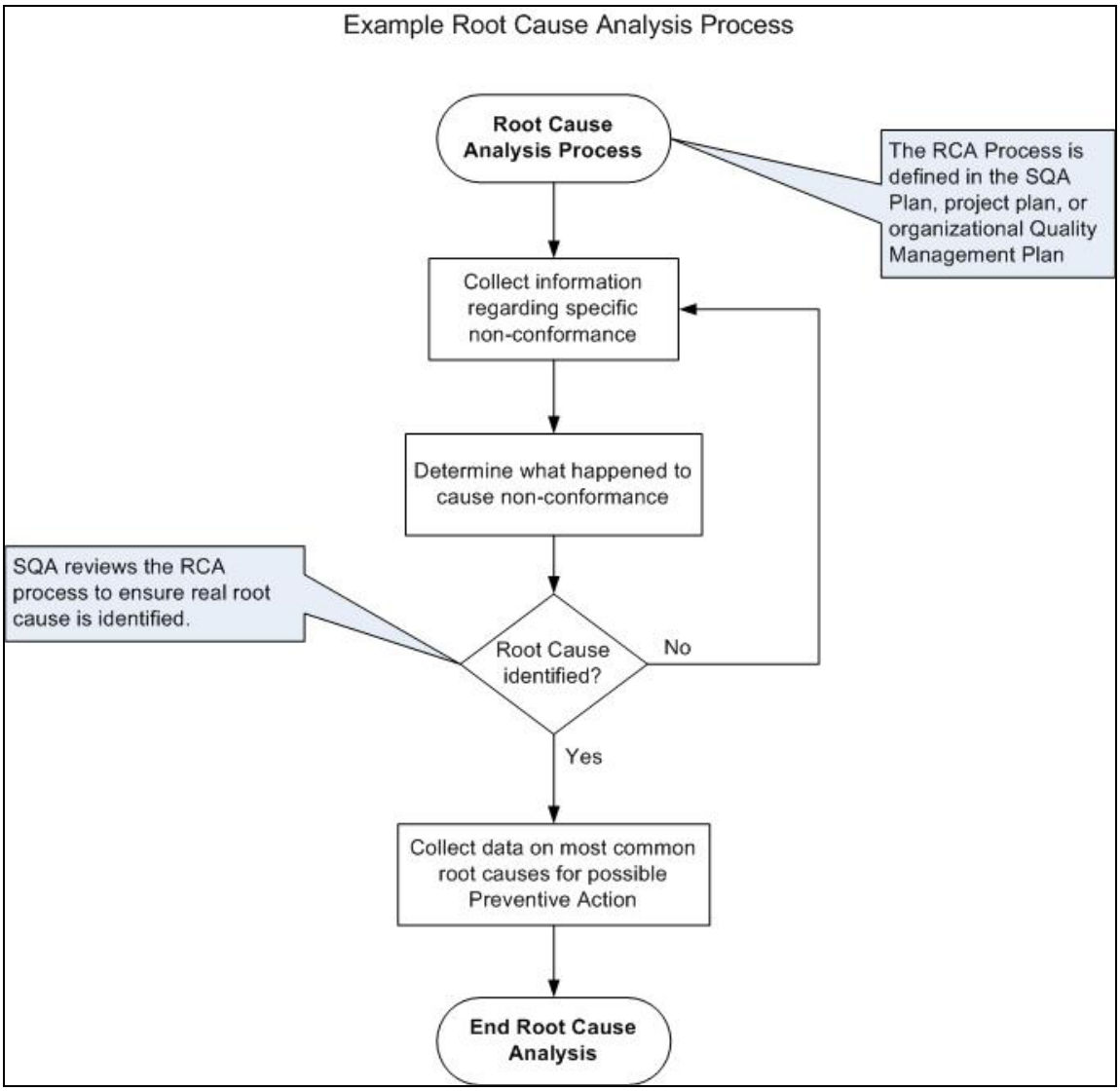

Figure J.2—Example root cause analysis process

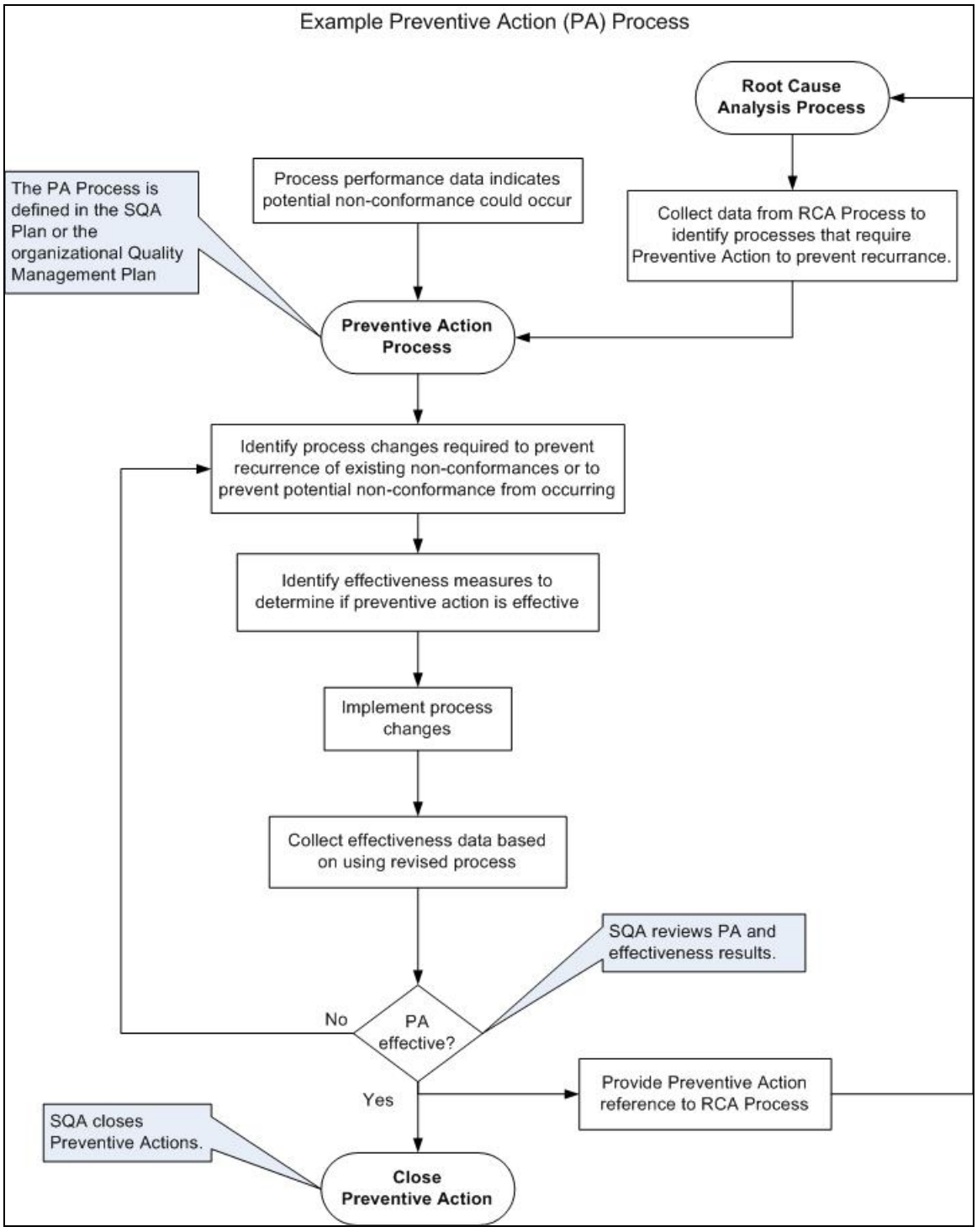

Figure J.3—Example preventive action process

Annex K

(informative)

Cross-reference

The intent of this annex is to illustrate the relationship between the 16 SQA activities defined in IEEE Std 730-2014 and the Life Cycle Processes as defined in ISO/IEC/IEEE 12207:2008. The life cycle processes defined in ISO/IEC/IEEE 12207:2008 include those activities in which software quality assurance (SQA) can have an impact that are part of Project Processes (6.3), software-specific activities included as part of Technical Processes (6.4), and activities that are part of the Software Implementation Process (7.1).

6.3 Project Processes

- 6.3.1 Project Planning
- 6.3.2 Project Assessment and Control

6.4 Technical Processes

- 6.4.1 Stakeholder Requirements Definition
- 6.4.7 Software Installation
- 6.4.8 Software Acceptance Support
- 6.4.9 Software Operation
- 6.4.10 Software Maintenance
- 6.4.11 Software Disposal

7.1 Software Implementation Process

- 7.1.2 Software Requirements Analysis
- 7.1.3 Software Architectural Design
- 7.1.4 Software Detailed Design
- 7.1.5 Software Construction
- 7.1.6 Software Integration
- 7.1.7 Software Qualification Testing

Table K.1, below, indicates where each of the SQA Activities could be applied throughout the Software Development Life Cycle based on the nature of the contract, the supplier-acquirer relationship, integrity levels, and other factors that are project-specific.

Table K.1—Cross-reference Process Assurance and Product Assurance SQA Activities to SLC Activities

SQA Activities	ISO/IEC/IEEE 12207:2008 Project Processes (6.3)		ISO/IEC/IEEE 12207:2008 Software Implementation Process (7.1)						ISO/IEC/IEEE 12207:2008 Technical Processes (6.4)					
	Project Planning	Project Assessment and Control	Software Requirements Analysis	Software Architectural Design	Software Detailed Design	Software Construction	Software Integration	Software Qualification Testing	Stakeholder Requirements Definition	Software Installation	Software Acceptance Support	Software Operation	Software Maintenance	Software Disposal
SQA process implementation activities														
Establish the SQA processes (5.3.1)	X													
Coordinate with related software processes (5.3.2)	X			Update as needed	Update as needed	Update as needed	Update as needed	Update as needed			Update as needed	Update as needed	Update as needed	
Document SQA planning (5.3.3)	X			Revise as needed	Revise as needed	Revise as needed	Revise as needed	Revise as needed			Revise as needed	Revise as needed	Revise as needed	
Execute the SQA Plan (5.3.4)		X	X	X	X	X	X	X		X	X	X	X	X
Manage SQA records (5.3.5)	X	X	X	X	X	X	X	X		X	X	X	X	X
Evaluate organizational independence and objectivity (5.3.6)	X			Monitor	Monitor	Monitor	Monitor	Monitor			Monitor	Monitor	Monitor	
Product assurance activities														
Evaluate plans for conformance to contracts, standards, and regulations (5.4.2)	X		X	X	X	X	X	X	X		X	X	X	X

Table K.1—Cross-reference Process Assurance and Product Assurance SQA Activities to SLC Activities (continued)

SQA Activities	ISO/IEC/IEEE 12207:2008 Project Processes (6.3)		ISO/IEC/IEEE 12207:2008 Software Implementation Process (7.1)						ISO/IEC/IEEE 12207:2008 Technical Processes (6.4)					
	Project Planning	Project Assessment and Control	Software Requirements Analysis	Software Architectural Design	Software Detailed Design	Software Construction	Software Integration	Software Qualification Testing	Stakeholder Requirements Definition	Software Installation	Software Acceptance Support	Software Operation	Software Maintenance	Software Disposal
Evaluate product for conformance to established requirements (5.4.3)				X	X	X	X	X			X	X	X	
Evaluate product for acceptability (5.4.4)							X	X		X	X	X	X	
Evaluate product life cycle support for conformance (5.4.5)	X	X	X	X		X	X		X	X	X	X	X	
Measure products (5.4.6)			X	X	X	X	X	X		X	X	X	X	X
Process assurance activities														
Evaluate life cycle processes and plans for conformance (5.5.2)	X	X	X	X	X	X	X	X	X	X	X	X	X	
Evaluate environments for conformance (5.5.3)			X	X	X	X	X	X		X	X	X	X	
Evaluate subcontractor processes for conformance (5.5.4)	X	X	X	X	X	X	X	X			X	X	X	
Measure processes (5.5.5)	X	X	X	X	X	X	X	X	X	X	X	X	X	
Assess staff skill and knowledge (5.5.6)	X	X	X	X	X	X	X	X			X	X	X	

Annex L

(informative)

Bibliography

Bibliographical references are resources that provide additional or helpful material but do not need to be understood or used to implement this standard. Reference to these resources is made for informational use only.

[B1] ASME NQA-1-2008, Quality Assurance Requirements for Nuclear Facility Applications.[8]

[B2] ASME NQA-1a-2009, Addenda to ASME NQA-1-2008, Quality Assurance Requirements for Nuclear Facility Applications.

[B3] ANSI/AAMI/IEC 62304:2006 Medical device software — Software life cycle processes.[9]

[B4] Canadian Standards Association. N286.7-99 — Quality Assurance of Analytical, Scientific and Design Computer Programs for Nuclear Power Plants.

[B5] Canadian Standards Association. N290.14-07 (R2012) — Qualification of Pre-Developed Software for Use in Safety-Related Instrumentation and Control Applications in Nuclear Power Plants.

[B6] Code of Federal Regulations Title 10 Part 21 (10 CFR 21) — Reporting of Defects and Noncompliance.[10]

[B7] Code of Federal Regulations Title 10 Part 50 (10 CFR 50) — Domestic Licensing of Production and Utilization Facilities.

[B8] Code of Federal Regulations Title 21 Part 820 (21 CFR 820) — Quality System Regulation (QSR), 1996.

[B9] EN 50128, Railway applications — Software for railway control and protection systems, 2011.[11]

[B10] EPRI NP-5652, Guideline for the Utilization of Commercial Grade Items in Nuclear Safety Related Applications, 1988.

[B11] EPRI NP-6406, Guidelines for the Technical Evaluation of Replacement Items in Nuclear Power Plants, 1990.

[B12] EPRI TR-1025243 Plant Engineering: Guideline for the Acceptance of Commercial-Grade Design and Analysis Computer Programs Used in Nuclear Safety-Related Applications, Revision 1, 2013.

[B13] EPRI TR-103291-CD, Handbook for Verification and Validation of Digital Systems, 1998.

[B14] EPRI TR-106439, Guideline on Evaluation and Acceptance of Commercial Grade Digital Equipment for Nuclear Safety Applications, 1996.

[B15] Gano, D. L., *Apollo Root Cause Analysis — A New Way of Thinking*, *2nd Ed.*, Apollonian Publications, 1999.

[B16] General Principles of Software Validation; Final Guidance for Industry and FDA Staff, 2002.

[B17] Guidance for Industry, Computerized Systems Used in Clinical Trials, 2007

[B18] Guidance for Industry, Cybersecurity for Networked Medical Devices Containing Off-the-Shelf (OTS) Software, 2005.

[8] ASME publications are available from the American Society of Mechanical Engineers (http://www.asme.org/).
[9] ANSI publications are available from the American National Standards Institute (http://www.ansi.org/).
[10] CFR publications are available from the U.S. Government Printing Office (http://www.gpo.gov).
[11] EN publications are available from the European Committee for Standardization (CEN) (http://www.cen.eu/).

[B19] Guidance for Industry, FDA Reviewers and Compliance on Off-The-Shelf Software Use in Medical Devices, 1999.

[B20] Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices, 2005.

[B21] IEC 60601-1-4, Medical electrical equipment — Part 1–4: General requirements for safety — Collateral Standard: programmable electrical medical systems, 2000.[12]

[B22] IEC 61508, Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems, 2010.

[B23] IEC 61513, Nuclear power plants — Instrumentation and control important to safety — General requirements for systems, 2011.

[B24] IEEE Std 7-4.3.2-2010 IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations.

[B25] IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.

[B26] IEEE Std 1012-2012 IEEE Standard for System and Software Verification and Validation.

[B27] IEEE Std 1220-2005 IEEE Standard for Application and Management of the Systems Engineering Process.

[B28] IEEE Std 15939-2008 IEEE Standard Adoption of ISO/IEC 15939:2007 — Systems and Software Engineering — Measurement Process.

[B29] IEEE Std 90003-2008 IEEE Guide — Adoption of ISO/IEC 90003:2004 Software Engineering — Guidelines for the Application of ISO 9001:2000 to Computer Software.

[B30] ISO 9000:2005 Quality management systems — Fundamentals and vocabulary.[13]

[B31] ISO 9001:2008 Quality management systems – Requirements.

[B32] ISO/IEC 13485:2003 Medical devices — Quality management systems — Requirements for regulatory purposes.

[B33] ISO/IEC 14971:2007 Medical devices — Application of risk management to medical devices.

[B34] ISO/IEC 15504-1:2004 Information Technology – Process Assessment — Part 1: Concepts and Vocabulary.

[B35] ISO/IEC 15504-2:2003 Information technology — Process assessment — Part 2: Performing an assessment.

[B36] ISO/IEC 15026-1:2013 Systems and Software Engineering — Systems and software Assurance — Part 1: Concepts and Vocabulary.

[B37] ISO/IEC 15026-2:2011 Systems and software engineering — Systems and software assurance — Part 2: Assurance case.

[B38] ISO/IEC 15026-3:2011, Systems and software engineering — Systems and software assurance — Part 3: System integrity levels.

[B39] ISO/IEC 15026-4:2012 Systems and software engineering — Systems and software assurance — Part 4: Assurance in the life cycle.

[B40] ISO/IEC 29110:2011, Software engineering — Lifecycle profiles for Very Small Entities (VSEs).

[B41] ISO/IEC TR 24774:2010 Systems and Software Engineering — Life Cycle Management — Guidelines for Process Description.

[12] IEC publications are available from the International Electrotechnical Commission (http://www.iec.ch/). IEC publications are also available in the United States from the American National Standards Institute (http://www.ansi.org/).
[13] ISO and ISO/IEC publications are available from the ISO Central Secretariat (http://www.iso.org/). ISO publications are also available in the United States from the American National Standards Institute (http://www.ansi.org/).

[B42] ISO/IEC/IEEE 24765:2010 Systems and software engineering – Vocabulary.

[B43] ISO/IEC/IEEE 29148:2011 Systems and Software Engineering — Life Cycle Processes — Requirements Engineering.

[B44] Jackson, D., Martyn T., Millett, L. (Editors), *Software for Dependable Systems: Sufficient Evidence?* Committee on Certifiably Dependable Software Systems, Computer Science and Telecommunications Board, National Research Council, National Academies Press, ISBN: 0-309-66738-0.

[B45] Land, S. K., *Results of the IEEE Survey of Software Engineering Standards Users*. Software Engineering Standards Symposium and Forum, 1997. Emerging International Standards. ISESS 97, 1–6 June 1997, pp. 242–270.

[B46] Laporte, C. Y., Alexandre, S., O'Connor, R. V., *A Software Engineering Lifecycle Standard for Very Small Enterprises*, EuroSPI 2008, CCIS 16, 3–5 September, pp. 129–141.

[B47] NASA Technical Standard 8739.8 2004, Software Assurance Standard, 2011.[14]

[B48] Project Management Institute, *Guide to the Project Management Body of Knowledge (PMBOK® Guide) 4th Ed*. Pittsburgh, PA: Project Management Institute, 2008.

[B49] Rhodes, T., et. al., "Software Assurance Using Structured Assurance Case Models," J. Res. Natl. Inst. Stand. Technol. 115, pp. 209–216, 2010.

[B50] Robitille, D. *Root Cause Analysis — Basic Tools and Techniques*, Paton Press, 2004.

[B51] RTCA DO-178B, Software Considerations in Airborne Systems and Equipment Certification, 1992.

[B52] U.S. Nuclear Regulatory Commission Regulatory Guide 1.28, Quality Assurance Program Criteria (Design and Construction), 2010.

[B53] U.S. Nuclear Regulatory Commission Regulatory Guide 1.33, Quality Assurance Program Requirements (Operation), 2013.

[B54] U.S. Nuclear Regulatory Commission Regulatory Guide 1.152, Criteria for use of Computers in Safety Systems of Nuclear Power Plants, 2011.

[B55] U.S. Nuclear Regulatory Commission Regulatory Guide 1.168, Verification, Validation, Reviews, and Audits for Digital Computer Software used in Safety Systems of Nuclear Power Plants, 2004.

[B56] U.S. Nuclear Regulatory Commission Regulatory Guide 1.169, Configuration Management Plans for Digital Computer Software Used in Safety Systems of Nuclear Power Plants, 2013.

[B57] U.S. Nuclear Regulatory Commission Regulatory Guide 1.170, Test Documentation for Digital Computer Software Used in Safety Systems of Nuclear Power Plants, 2013.

[B58] U.S. Nuclear Regulatory Commission Regulatory Guide 1.171, Software Unit Testing for Digital Computer Software Used in Safety Systems of Nuclear Power Plants, 2013.

[B59] U.S. Nuclear Regulatory Commission Regulatory Guide 1.172, Software Requirement Specifications for Digital Computer Software and Complex Electronics Used in Safety Systems of Nuclear Power Plants, 2013.

[B60] U.S. Nuclear Regulatory Commission Regulatory Guide 1.173, Developing Software Life-Cycle Processes for Digital Computer Software Used in Safety Systems of Nuclear Power Plants, 2013.

[B61] U.S. Nuclear Regulatory Commission. Branch Technical Position 7-14 — Guidance On Software Reviews For Digital Computer-Based Instrumentation And Control Systems, Rev 5, March 2007.

[B62] U.S. Nuclear Regulatory Commission. NUREG/CR-6101, Software Reliability and Safety in Nuclear Reactor Protection Systems. 1993.

[B63] U.S. Nuclear Regulatory Commission. NUREG/CR-6421, A Proposed Acceptance Process for Commercial Off-the-Shelf (COTS) Software in Reactor Applications, 1996.

[14] NASA publications are available from NASA Center for Aerospace Information (CASI) (http://www.nasa.gov/).

[B64] U.S. Nuclear Regulatory Commission. NUREG/CR-6430, Software Safety Hazard Analysis, 1996.

[B65] U.S. Nuclear Regulatory Commission. NUREG/CR-6463, Review Guidelines on Software Languages for use in Nuclear Power Plant Safety Systems, 1996.