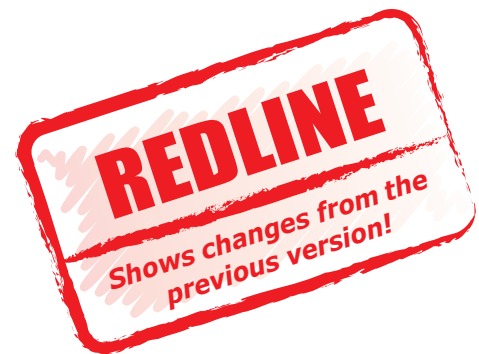


IEEE Standard for Software Quality Assurance Processes

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)

This is a Redline Document produced by Techstreet, a business of Thomson Reuters.

This document is intended to provide users with an indication of changes from one edition to the next. It includes a full-text version of the new document, plus an indication of changes from the previous version.

Redlines are designed to save time and improve efficiencies by using the latest software technology to find and highlight document changes. More professionals are using valuable new technologies like redlines, to help improve outcomes in a fastpaced global business world.

Because it may not be technically possible to capture all changes accurately, it is recommended that users consult previous editions as appropriate. In all cases, only the current base version of this publication is to be considered the official document.

Redline Processing Notes:

1. ~~Red Text~~ - Red strikethrough text denotes deletions.
2. Blue Text - Blue underlined text denotes modifications and additions.

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

Copyright © 2014 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 13 June 2014. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

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

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html>.

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

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

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate fee, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

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

1. Overview	1
1.1 Scope	1
1.2 Purpose	1
1.3 Field of application	2
1.4 Limitations	2
1.5 Conformance	2
1.6 Intended usage of this standard	3
1.7 Organization of this standard	3
2. Normative references	5
3. Definitions, acronyms, and abbreviations	5
3.1 Conventions	5
3.2 Definitions	5
3.3 Acronyms and abbreviations	9
4. Key concepts of Software Quality Assurance	9
4.1 Organizational management responsibility	9
4.2 Organization and project relationship	9
4.3 Software quality and relationship to requirements	12
4.4 Overview of SQA activities	14
4.5 Acquirer and supplier perspectives	15
4.6 Key concepts of this standard	15
4.7 Software process improvement	16
4.8 Maintaining quality management system consistency	16
5. Software Quality Assurance process	17
5.1 Purpose	17
5.2 Organization of SQA process outcomes	17
5.3 SQA process implementation activities	19
5.4 Product assurance activities	27
5.5 Process assurance activities	31
Annex A (informative) Mapping between 7.2.3 of ISO/IEC/IEEE 12207:2008 and IEEE Std 730-2014 ..	37
Annex B (informative) Mapping between SQA Plan outlines contained in IEEE Std 730-2002 and IEEE Std 730-2014	41
Annex C (informative) Guidance for creating Software Quality Assurance Plans	44
C.1 Introduction	44
C.2 Organization of this annex	47
C.3 Guidance	47
Annex D (informative) Mapping IEEE Std 730-2014 and ISO/IEC 15504 (SPICE)	81
D.1 Capability Level 1 for SQA	81
D.2 Capability Level 2 for SQA	81
D.3 Capability Level 3 to Capability Level 5 for SQA	83
D.4 Capability Level 3 to Capability Level 5 for other SLC processes	84
Annex E (informative) Applying IEEE Std 730-2014 — Industry-specific guidance	86
E.1 Medical device industry	86

E.2 Nuclear power generation industry	90
Annex F (informative) SQA activities and their relationship to the agile development process.....	95
F.1 Introduction	95
F.2 Modifying the 16 SQA activities to accommodate agile software development.....	96
Annex G (informative) Mapping between IEEE Std 730-2014 and ISO/IEC 29110 standard for Very Small Entities.....	99
Annex H (informative) Software tool validation.....	104
Annex I (informative) Assessing product risk: Software integrity levels and assurance cases	106
I.1 Software integrity levels.....	106
I.2 Overview of activities for integrity level determination.....	109
I.3 Assurance cases.....	111
Annex J (informative) Example corrective and preventive action process and root cause analysis process	116
Annex K (informative) Cross-reference	120
Annex L (informative) Bibliography.....	123

IEEE Standard for Software Quality

Assurance ~~Plans~~ Processes

IMPORTANT NOTICE: IEEE Standards documents are not intended to ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. Implementers of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

This IEEE document is made available for use subject to important notices and legal disclaimers. These notices and disclaimers appear in all publications containing this document and may be found under the heading "Important Notice" or "Important Notices and Disclaimers Concerning IEEE Documents." They can also be obtained on request from IEEE or viewed at <http://standards.ieee.org/IPR/disclaimers.html>.

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 12207:2008¹ 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

~~This standard applies to the development of a software quality assurance plan (SQAP). The existence of this standard should not be construed to prohibit additional content in a SQAP. An assessment should be made for the specific software item to assure adequacy of coverage. Where this standard is invoked for an organization or project engaged in producing several software items, the applicability of the standard should be specified for each of the software items.~~

~~Although this document does not require the use of IEEE/EIA Std 12207.0 1996 [B17] and IEEE/EIA Std 12207.1 1997 [B18], it is consistent with those two standards.¹ An SQAP meeting the requirements of this standard will be in document compliance with the SQAP information item of IEEE/EIA 12207.1 1997 [B18].~~

¹Information on normative references can be found in Clause 2.

²Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

¹~~The numbers in brackets correspond to those of the bibliography in Annex A.~~

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 ~~preparation and content of software quality assurance plans.~~ 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](#)) 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 ~~to this standard~~

1.5.1 Conformance language conventions

~~Content conformance to this standard can be claimed when all requirements (indicated by “shall”) are carried out as defined in this standard. Format conformance to this standard can be claimed if the resulting SQAP is in the format specified in Clause 4.~~ 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).

³IEEE publications are available from the Institute of Electrical and Electronics Engineers (<http://standards.ieee.org/>).

⁴The IEEE standards or products referred to in this clause are trademarks of the Institute of Electrical and Electronics Engineers, Inc.

- [Refer to Annex D for information about mapping between IEEE Std 730-2014 and ISO/IEC 15504-1:2004 \[B34\].](#)⁵
- [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.

⁵ [Information in brackets corresponds to those of the bibliography in Annex L.](#)

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 SOAP 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 Vocabulary⁷ should be consulted for terms not defined in this clause.

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>.

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**.

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 15939TM-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 1220TM-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.

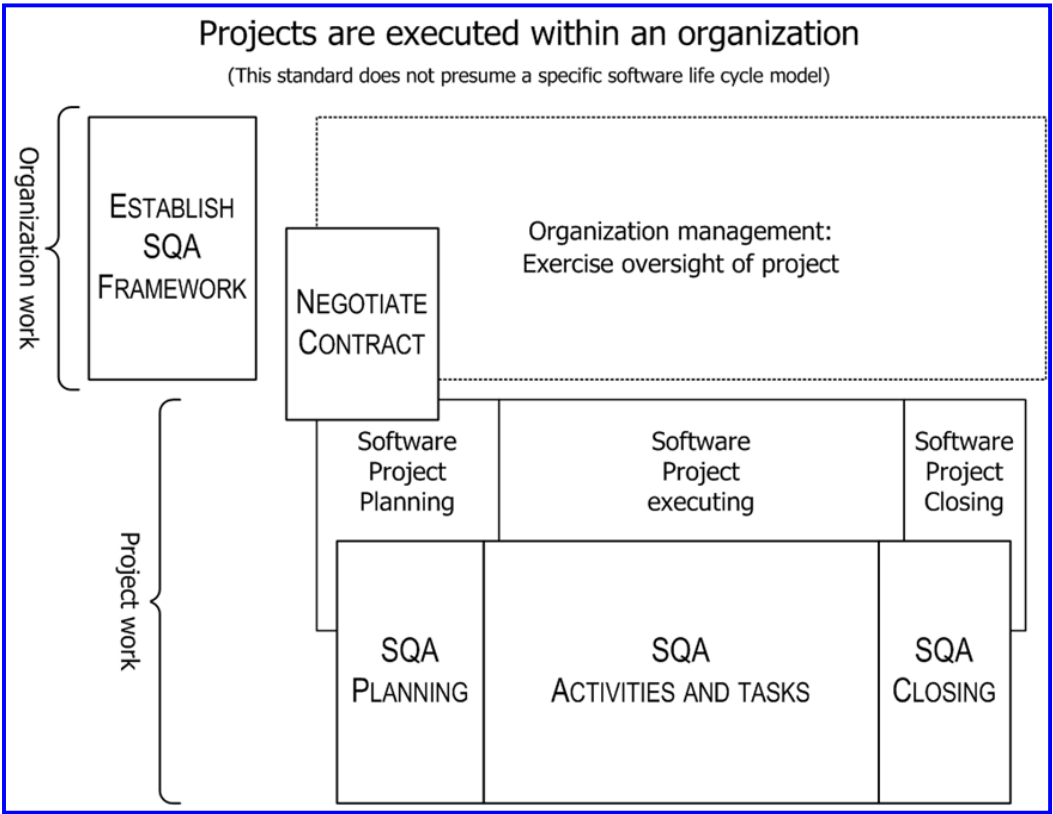
The organization's management supports the SQA function by providing the outcomes listed in 5.3.1.2.

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.



(New)

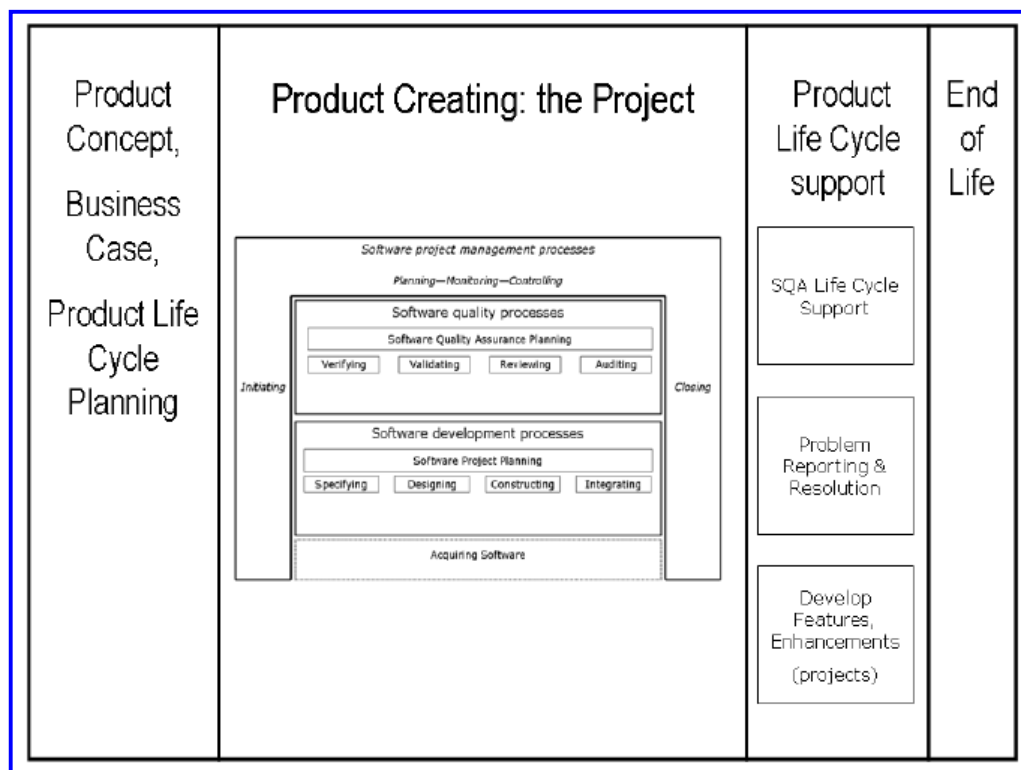
Figure 1 – Relationship between the organization and the project

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.



(New)

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.

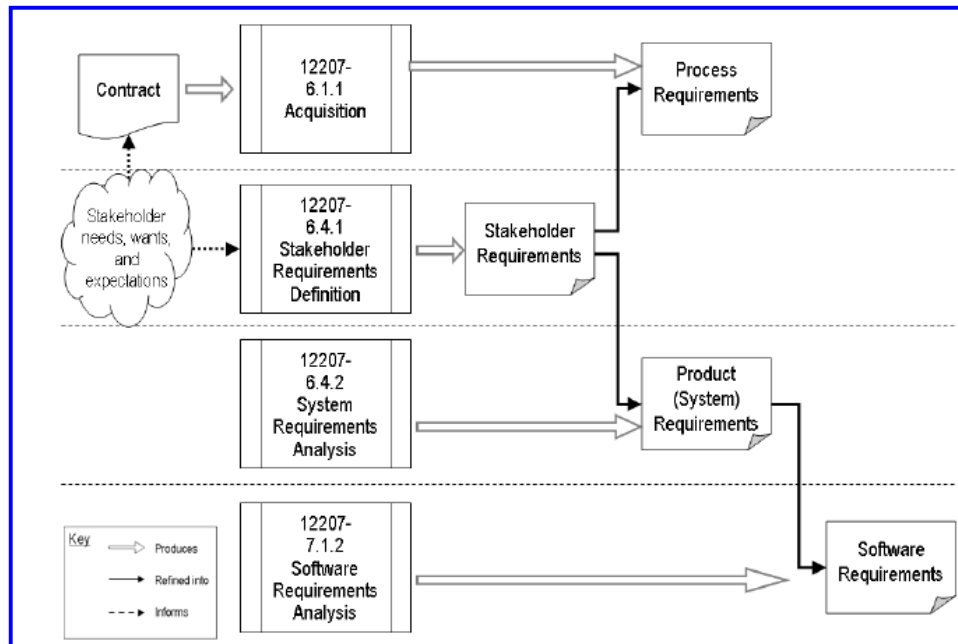
Software quality is achieved by conforming to established requirements as described in 5.4.3. In some industries, conformance to requirements also includes the constraint that the product meets its intended use and is evaluated in the actual use setting. Because requirements are the basis for quality, 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.



(New)

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.

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

(New)

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

(New)

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 be 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.

(New)

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