730

IEEE Standard for
Software Quality Assurance Plans

Sponsor

**Technical Committee on Software Engineering
of the
IEEE Computer Society**

Approved August 17, 1989

IEEE Standards Board

Approved January 22, 1990

American National Standards Institute

IEEE Standards documents are developed within the Technical Committees of the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE which have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old, and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason IEEE and the members of its technical committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

> Secretary, IEEE Standards Board
> 445 Hoes Lane
> P.O. Box 1331
> Piscataway, NJ 08855-1331
> USA

IEEE Standards documents are adopted by the Institute of Electrical and Electronics Engineers without regard to whether their adoption may involve patents on articles, materials, or processes. Such adoption does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the standards documents.

Foreword

This standard assists in the preparation and content of Software Quality Assurance Plans and provides a standard against which such plans can be prepared and assessed. It is directed toward the development and maintenance of critical software—that is, where failure could impact safety or cause large financial or social losses.

The readers of this document are referred to ANSI/IEEE Std 983-1986, IEEE Guide for Software Quality Assurance Planning, for recommended approaches to good software quality assurance practices in support of this standard. While ANSI/IEEE Std 983-1986 specifically refers to ANSI/IEEE Std 730-1984, almost all of its content applies directly to this revision.

In this standard, firmware is considered to be software and is to be treated as such.

Footnotes are not part of the standard.

There are three groups to whom this standard applies: the user, the developer, and the public.

(1) The user, who may be another element of the same organization developing the software, has a need for the product. Further, the user needs the product to meet the requirements identified in the specification. The user thus cannot afford a "hands-off" attitude toward the developer and rely solely on a test to be executed at the end of the software development time period. If the product should fail, not only does the same need still exist, but also a portion of the development time has been lost. Therefore, the user needs to obtain a reasonable degree of confidence that the product is in the process of acquiring required attributes during software development.

(2) The developer needs an established standard against which to plan and to be measured. It is unreasonable to expect a complete reorientation from project to project. Not only is it not cost effective, but, unless there exists a stable framework on which to base changes, improvement cannot be made.

(3) The public may be affected by the users' use of the product. These users include, for example, depositors at a bank or passengers using a reservation system. Users have requirements, such as legal rights, which preclude haphazard development of software. At some later date, the user and the developer may be required to show that they acted in a reasonable and prudent professional manner to ensure that required software attributes were acquired.

This standard was prepared by the Software Engineering Standards Subcommittee of the Software Engineering Technical Committee of the IEEE Computer Society. It was initially approved by the IEEE Standards Board for "trial use" in December of 1979, with a subsequent "full use" approval in September of 1981, and a revision approved on June 14, 1984.

The Working Group consisted of the following members:

F. Buckley, *Chairperson*   **C. White-Partain,** *Vice-Chairperson*

Executive Committee:

S. Ali	R. Fordham	C. Modell
W. Barret	M. Hein	S. Norman
R. Braun	R. Kosinski	S. Trauth
W. Eventoff	E. Maginnius	A. Tegtmeier
	L. Marquis	

Members:

J. Agrawal	S. Jones	A. Serna, Jr.
M. Ben-Menachem	H. Kaplan	R. Shillato
J. Case	M. Karasik	T. Smith
J. Chihorek	R. Kenneavy	R. Stanton
M. Dewalt	T. Kurihara	R. Stenglein
D. Doty	B. Livson	A. Sullivan
R. Euler	A. Miller	O. Tanir
R. Evans	R. Martin	E. Testa
J. Ewin	P. Pfister	L. To
M. Fitch	R. Poston	G. Tucker
M. Ghiassi	F. Rienstra	H. van Doornum
J. Gilmore	J. Roberts	C. Von Schantz
B. Gundaker	F. Rose	D. Wallace
J. Horch	W. Schultzer	P. Willis
R. Ivy	W. Schunk	P. Wolfgang
	C. Seddin	

Special Representatives to the Software Engineering Standards Subcommittee for this action were as follows:

Nuclear Power Engineering Committee/Power Engineering Society: N. Farr

Quality Assurance Management Committee of the IEEE Communications Society: R. Mosher

Canadian Standards Association: B. Dyczkowsky

Data Processing Manufacturers' Association: W. Perry

At the time that the IEEE Standards Board approved this revision, the Software Engineering Standards Subcommittee, which was the balloting committee that approved this document for submission to the IEEE Standards Board, had the following membership:

John Horch, *Chairman*

W. Arfvidson	J. Gilmore	P. Peterson
C. Arnold	S. Gloss-Soler	J. Phippen
R. Aurbach	J. Glynn	J. Pope
E. Baker	J. Gonzalez-Sanz	R. Poston
B. Banerjee	V. Guarnera	I. Pyle
B. Beizer	D. Gustafson	G. Ray
M. Ben-Menachem	J. Guttman	M. Razy
R. Berlack	B. Harbort	J. Reddan
B. Bina	C. Hay	S. Redwine
M. Blackledge	J. Hillawi	J. Roberts
R. Blasewitz	J. Horch	R. Roman
R. Both	P. Hung	R. Roth
K. Briggs	D. Johnson III	F. Ruhlman
A. Brown	J. Kalasky	S. Schach
W. Bryan	L. Kaleda	H. Schaefer
F. Buckley	M. Karasik	N. Schneidewind
C. Carpenter	R. Karcich	W. Schnoege
J. Case	O. Kato	H. Schock
T. Chow	P. Klopfenstein	D. Schultz
W. Chung	S. Koenig	G. Schumacher
F. Coallier	R. Kosinski	L. Seagren
G. Copeland	T. Kurihara	L. Shafer
P. Daggett	L. Lam	R. Shillato
M. Daniels	J. Lane	D. Siefert
T. Daughtrey	R. Lane	I. Sjors
K. DeJong	G. Larsen	A. Sorkowitz
D. Deluna	F. Lim	T. Stalhane
C. Dencker	K. Linberg	N. Stewart
P. Denny	B. Lindberg	W. Strigel
B. Derganc	B. Livson	R. Thayer
A. Dolinsky	H. Mains	P. Thompson
R. Dwyer	H. Malec	G. Tice
L. Egan	J. Malsbury	T. Tillmanns
M. Egeland	L. Marquis	E. Tomlin
S. Eisen	P. Marriott	S. Trauth
R. Erickson	R. Martin	L. Tripp
R. Euler	T. Matsubara	M. Uchida
C. Evans	I. Mazza	D. Ulery
J. Fendrich	J. Mersky	H. Verne
A. Foley	C. Modell	C. Von Schantz
R. Fordham	H. Nagano	D. Wallace
J. Forman	S. Najafi	J. Walter
J. Franklin	G. Neidhart	R. Weger
I. Fuentes	D. Nickle	P. Willis
A. Geraci	J. O'Day	P. Wolfgang
Y. Gershkovitch	M. Olson	T. Worthington
D. Geyer	W. Osbourne	N. Yopconka
E. Gibbs	T. Parrish	P. Zoll
	M. Perkins	

When the IEEE Standards Board approved this standard on August 17, 1989, it had the following membership:

Dennis Bodson, *Chairman* **Marco W. Migliaro,** *Vice Chairman*

Andrew G. Salem, *Secretary*

Arthur A. Blaisdell	Kenneth D. Hendrix	John E. May, Jr.
Fletcher J. Buckley	Theodore W. Hissey, Jr.	Lawrence V. McCall
Allen L. Clapp	John W. Horch	L. Bruce McClung
James M. Daly	David W. Hutchins	Donald T. Michael*
Stephen R. Dillon	Frank D. Kirschner	Richard E. Mosher
Donald C. Fleckenstein	Frank C. Kitzantides	Stig Nilsson
Eugene P. Fogarty	Joseph L. Koepfinger*	L. John Rankine
Jay Forster*	Edward Lohse	Gary S. Robinson
Thomas L. Hannan		Donald W. Zipse

*Member Emeritus

Contents

IEEE Standard for
Software Quality Assurance Plans

1. Scope and References

1.1 Scope. The purpose of this standard is to provide uniform, minimum acceptable requirements for preparation and content of Software Quality Assurance Plans (SQAPs).

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 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.

This standard applies to the development and maintenance of critical software. For non-critical software, or for software already developed, a subset of the requirements of this standard may be applied.

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.

1.2 References. The standards listed below should be used for further information. In using these references, the latest revisions should be obtained. Compliance with this standard does not require nor imply compliance with any of those listed.

[1] ANSI/ASME NQA-1-1983, Quality Assurance Program Requirements for Nuclear Facilities.[1]

[2] ANSI/IEEE Std 603-1980, IEEE Standard Criteria for Safety Systems for Nuclear Power Generating Stations.

[3] ANSI/IEEE Std 729-1983, IEEE Standard Glossary of Software Engineering Terminology.[2]

[4] ANSI/IEEE Std 828-1983, IEEE Standard for Software Configuration Management Plans.

[5] ANSI/IEEE Std 829-1983, IEEE Standard for Software Test Documentation.

[6] ANSI/IEEE Std 830-1984, IEEE Guide for Software Requirements Specifications.

[7] IEEE Std 982.1-1988, IEEE Standard Dictionary of Measures to Produce Reliable Software.

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

[9] ANSI/IEEE Std 983-1986, IEEE Guide for Software Quality Assurance Planning. (This is being redesignated to 730.2.)

[10] ANSI/IEEE Std 990-1987, IEEE Recommended Practice for Ada as a Program Design Language.

[11] ANSI/IEEE Std 1002-1987, IEEE Standard Taxonomy of Software Engineering Standards.

[12] ANSI/IEEE Std 1008-1987, IEEE Standard for Software Unit Testing.

---

[1]ANSI/ASME publications are available from the Sales Department, American National Standards Institute, 1430 Broadway, New York, NY 10018 or from the ASME Order Department, 22 Law Drive, P.O. Box 2300, Fairfield, NJ 07007-2300.

[2]ANSI/IEEE publications may be obtained from the IEEE Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331 or from the Sales Department, American National Standards Institute, 1430 Broadway, New York, NY 10018.

[13] ANSI/IEEE Std 1012-1986, IEEE Standard for Software Verification and Validation Plans.

[14] ANSI/IEEE Std 1016-1987, IEEE Recommended Practice for Software Design Descriptions.

[15] IEEE Std 1028-1988, IEEE Standard for Software Reviews and Audits.

[16] ANSI/IEEE Std 1033-1985, IEEE Recommended Practice for Application of IEEE Std 828 to Nuclear Power Generating Stations.

[17] ANSI/IEEE Std 1042-1987, IEEE Guide to Software Configuration Management.

[18] ANSI/IEEE Std 1058.1-1988, IEEE Standard for Software Project Management Plans.

[19] ANSI/IEEE Std 1063-1987, IEEE Standard for Software User Documentation.

[20] ANSI/IEEE/ANS 7-4.3.2-1982, Application Criteria for Programmable Digital Computer Systems in Safety Systems of Nuclear Power Generating Stations.

2. Definitions and Acronyms

2.1 Definitions. The definitions listed below establish meaning in the context of this standard. Other definitions can be found in ANSI/IEEE Std 729-1983, IEEE Standard Glossary of Software Engineering Terminology, or latest revision thereof [3].[3] For the purpose of this standard, the term "software" includes firmware, documentation, data, and execution control statements (eg, command files, job control language, etc).

branch metric. The result of dividing the total number of modules in which every branch has been executed at least once by the total number of modules.[4]

critical software. Software whose failure would impact safety or cause large financial or social losses.

decision point metric. The result of dividing the total number of modules in which every decision point has had (1) all valid conditions, and (2) at least one invalid condition, correctly processed, divided by the total number of modules.[5]

domain metric. The result of dividing the total number of modules in which one valid sample and one invalid sample of every class of input data items (external messages, operator inputs, and local data) have been correctly processed, by the total number of modules.[6]

error message metric. The result of dividing the total number of error messages that have been formally demonstrated, by the total number of error messages.

quality assurance. A planned and systematic pattern of all actions necessary to provide adequate confidence that the item or product conforms to established technical requirements.

requirements demonstration metric. The result of dividing the total number of separately-identified requirements in the software requirements specification (SRS) that have been successfully demonstrated by the total number of separately-identified requirements in the SRS.

2.2 Acronyms. The following alphabetical contractions appear within the text of this standard:
CDR—critical design review
PDR—preliminary design review
SCMP—software configuration management plan
SCMPR—software configuration management plan review
SDD—software design description
SQA—software quality assurance
SQAP—software quality assurance plan
SRR—software requirements review
SRS—software requirements specification
SVVP—software verification and validation plan
SVVPR—software verification and validation plan review
SVVR—software verification and validation report
UDR—user documentation review

---

[3]The numbers in brackets correspond to those of the references listed in 1.2.
[4]This definition assumes that the modules are essentially the same size.

---

[5] See footnote 4.
[6] See footnote 4.

8

3. Software Quality Assurance Plan

The Software Quality Assurance Plan shall include the sections listed below to be in compliance with this standard. The sections should be ordered in the described sequence. If the sections are not ordered in the described sequence, then a table shall be provided at the end of the SQAP that provides a cross-reference from the lowest numbered subsection of this standard to that portion of the SQAP where that material is provided. If there is no information pertinent to a section, the following shall appear below the section heading, "This section is not applicable to this plan," together with the appropriate reasons for the exclusion.

(1) Purpose
(2) Reference documents
(3) Management
(4) Documentation
(5) Standards, practices, conventions, and metrics
(6) Reviews and audits
(7) Test
(8) Problem reporting and corrective action
(9) Tools, techniques, and methodologies
(10) Code control
(11) Media control
(12) Supplier control
(13) Records collection, maintenance, and retention
(14) Training
(15) Risk management

Additional sections may be added as required.

Some of the material may appear in other documents. If so, then reference to these documents should be made in the body of the SQAP. In any case, the contents of each section of the plan shall be specified either directly or by reference to another document.

The SQAP shall be approved by the chief operating officer of each unit of the organization having responsibilities defined within this SQAP or their designated representatives.

Details for each section of the SQAP are described in 3.1 through 3.15 of this standard.[7]

---

[7]Guidance in the use of this standard can be found in ANSI/IEEE Std 983-1986 [9]. For an expansion of the quality and equipment qualification requirements of IEEE Std 603-1980, IEEE Standard Criteria for Safety Systems for Nuclear Power Generating Stations [2]; to encompass software design, software implementation, and computer systems validation, see ANSI/IEEE/ANS 7-4.3.2-1982 [20].

3.1 Purpose (Section 1 of the SQAP). This section shall delineate the specific purpose and scope of the particular SQAP. It shall list the name(s) of the software items covered by the SQAP and the intended use of the software. It shall state the portion of the software life cycle covered by the SQAP for each software item specified.

3.2 Reference Documents (Section 2 of the SQAP). This section shall provide a complete list of documents referenced elsewhere in the text of the SQAP.

3.3 Management (Section 3 of the SQAP). This section shall describe organization, tasks, and responsibilities.[8]

3.3.1 Organization. This paragraph shall depict the organizational structure that influences and controls the quality of the software. This shall include a description of each major element of the organization together with the delegated responsibilities. Organizational dependence or independence of the elements responsible for SQA from those responsible for software development and use shall be clearly described or depicted.

3.3.2 Tasks. This paragraph shall describe (a) that portion of the software life cycle covered by the SQAP, (b) the tasks to be performed with special emphasis on software quality assurance activities, and (c) the relationships between these tasks and the planned major check-points. The sequence of the tasks shall be indicated.

3.3.3 Responsibilities. This paragraph shall identify the specific organizational elements responsible for each task.

3.4 Documentation (Section 4 of the SQAP)

3.4.1 Purpose. This section shall perform the following functions:

(1) Identify the documentation governing the development, verification and validation, use, and maintenance of the software.
(2) State how the documents are to be checked for adequacy. This shall include the criteria and the identification of the review or audit by which the adequacy of each document shall be confirmed, with reference to Section 6 of the SQAP.

---

[8]See ANSI/IEEE Std 1002-1987 [11] and ANSI/IEEE Std 1058.1-1988 [18].

9

3.4.2 Minimum Documentation Requirements.
To ensure that the implementation of the software satisfies requirements, the following documentation is required as a minimum:

3.4.2.1 Software Requirements Specification (SRS). The SRS shall clearly and precisely describe each of the essential requirements (functions, performances, design constraints, and attributes) of the software and the external interfaces. Each requirement shall be defined such that its achievement is capable of being objectively verified and validated by a prescribed method; for example, inspection, analysis, demonstration, or test.[9]

3.4.2.2 Software Design Description (SDD). The SDD shall depict how the software will be structured to satisfy the requirements in the SRS. The SDD shall describe the components and subcomponents of the software design, including data bases and internal interfaces. The SDD shall be prepared first as the Preliminary SDD (also referred to as the Top-Level SDD) and shall be subsequently expanded to produce the Detailed SDD.[10]

3.4.2.3 Software Verification and Validation Plan (SVVP). The SVVP shall identify and describe the methods (for example, inspection, analysis, demonstration, or test) to be used:[11]

(1) To verify that (a) the requirements in the SRS have been approved by an appropriate authority, (b) the requirements in the SRS are implemented in the design expressed in the SDD; and (c) the design expressed in the SDD is implemented in the code.

(2) To validate that the code, when executed, complies with the requirements expressed in the SRS.

3.4.2.4 Software Verification and Validation Report (SVVR). The SVVR shall describe the results of the execution of the SVVP.

3.4.2.5 User Documentation. User documentation (eg, manual, guide, etc) shall specify and describe the required data and control inputs, input sequences, options, program limitations, and other activities or items necessary for successful execution of the software. All error messages shall be identified and corrective actions described. A method of describing user-identified errors or problems to the developer or the owner of the software shall be described. (Embedded software that has no direct user interaction has no need for user documentation and is therefore exempted from this requirement.)[12]

3.4.2.6 Software Configuration Management Plan (SCMP). The SCMP shall document methods to be used for identifying software items, controlling and implementing changes, and recording and reporting change implementation status.[13]

3.4.3 Other. Other documentation may include the following:

(1) Software Development Plan
(2) Standards and Procedures Manual
(3) Software Project Management Plan
(4) Software Maintenance Manual.

3.5 Standards, Practices, Conventions and Metrics (Section 5 of the SQAP)

3.5.1 Purpose. This section shall:

(1) Identify the standards, practices, conventions and metrics to be applied.
(2) State how compliance with these items is to be monitored and assured.

3.5.2 Content. The subjects covered shall include the basic technical, design, and programming activities involved, such as documentation, variable and module naming, programming, inspection, and testing. As a minimum, the following information shall be provided:[14]

(1) Documentation standards
(2) Logic structure standards
(3) Coding standards
(4) Commentary standards
(5) Testing standards and practices
(6) Selected software quality assurance product and process metrics such as:
   (a) Branch metric
   (b) Decision point metric
   (c) Domain metric
   (d) Error message metric
   (e) Requirements demonstration metric

---

[9]See ANSI/IEEE Std 830-1984 [6].
[10]See ANSI/IEEE Std 1016-1987 [14].
[11]See ANSI/IEEE Std 829-1983 [5], ANSI/IEEE Std 1008-1987 [12], and ANSI/IEEE Std 1012-1986 [13].

[12]See ANSI/IEEE Std 1063-1988 [19].
[13]See ANSI/IEEE Std 828-1983 [4] and ANSI/IEEE Std 1042-1987 [17]. See also ANSI/IEEE Std 1033-1985 [16].
[14]See ANSI/IEEE Std 990-1987 [10], ANSI/IEEE Std 982.1-1988 [7] and ANSI/IEEE Std 982.2-1988 [8].

3.6 Reviews and Audits (Section 6 of the SQAP)

3.6.1 Purpose. This section shall:[15]

(1) Define the technical and managerial reviews and audits to be conducted.
(2) State how the reviews and audits are to be accomplished.
(3) State what further actions are required and how they are to be implemented and verified.

3.6.2 Minimum Requirements. As a minimum, the following reviews and audits shall be conducted:

3.6.2.1 Software Requirements Review (SRR). The SRR is held to ensure the adequacy of the requirements stated in the SRS.

3.6.2.2 Preliminary Design Review (PDR). The PDR (also known as top-level design review) is held to evaluate the technical adequacy of the preliminary design (also known as top-level design) of the software as depicted in the preliminary software design description.

3.6.2.3 Critical Design Review (CDR). The CDR (also known as detailed design review) is held to determine the acceptability of the detailed software designs as depicted in the detailed software design description in satisfying the requirements of the SRS.

3.6.2.4 Software Verification and Validation Plan Review (SVVPR). The SVVPR is held to evaluate the adequacy and completeness of the verification and validation methods defined in the SVVP.

3.6.2.5 Functional Audit. This audit is held prior to the software delivery to verify that all requirements specified in the SRS have been met.

3.6.2.6 Physical Audit. This audit is held to verify that the software and its documentation are internally consistent and are ready for delivery.

3.6.2.7 In-Process Audits. In-process audits of a sample of the design are held to verify consistency of the design, including:

(1) Code versus design documentation
(2) Interface specifications (hardware and software)
(3) Design implementations versus functional requirements
(4) Functional requirements versus test descriptions

3.6.2.8 Managerial Reviews. Managerial reviews are held periodically to assess the execution of all of the actions and the items identified in the SQAP. These reviews shall be held by an organizational element independent of the unit being reviewed, or by a qualified third party. This review may require additional changes in the SQAP itself.

3.6.2.9 Software Configuration Management Plan Review (SCMPR). The SCMPR is held to evaluate the adequacy and completeness of the configuration management methods defined in the SCMP.

3.6.2.10 Post Mortem Review. This review is held at the conclusion of the project to assess the development activities implemented on that project and to provide recommendations for appropriate actions.

3.6.3 Other. Other reviews and audits may include the user documentation review (UDR). This review is held to evaluate the adequacy (eg, completeness, clarity, correctness, and usability) of user documentation.

3.7 Test (Section 7 of the SQAP). This section shall identify all the tests not included in the SVVP for the software covered by the SQAP and shall state the methods to be used.[16]

3.8 Problem Reporting and Corrective Action (Section 8 of the SQAP). This section shall:

(1) Describe the practices and procedures to be followed for reporting, tracking, and resolving problems identified in both software items and the software development and maintenance process.
(2) State the specific organizational responsibilities concerned with their implementation.

3.9 Tools, Techniques, and Methodologies (Section 9 of the SQAP). This section shall identify the special software tools, techniques, and methodologies that support SQA, state their purposes, and describe their use.

3.10 Code Control (Section 10 of the SQAP). This section shall define the methods and facilities used to maintain, store, secure and document controlled versions of the identified software during all phases of the software life cycle. This may be implemented in conjunction with a computer program library. This may be provided as a part of the SCMP.

---

[15]See IEEE Std 1028-1988 [15].

[16]See ANSI/IEEE Std 829-1983 [5] and ANSI/IEEE Std 1008-1987 [12].

If so, an appropriate reference shall be made thereto.

3.11 Media Control (Section 11 of the SQAP).
This section shall state the methods and facilities to be used to (a) identify the media for each computer product and the documentation required to store the media, including the copy and restore process, and (b) protect computer program physical media from unauthorized access or inadvertent damage or degradation during all phases of the software life cycle. This may be provided as a part of the SCMP. If so, an appropriate reference shall be made thereto.

3.12 Supplier Control (Section 12 of the SQAP).
This section shall state the provisions for assuring that software provided by suppliers meets established requirements. In addition, this section shall state the methods that will be used to assure that the software supplier receives adequate and complete requirements. For previously-developed software, this section shall state the methods to be used to assure the suitability of the product for use with the software items covered by the SQAP. For software that is to be developed, the supplier shall be required to prepare and implement a SQAP in accordance with this standard. This section shall also state the methods to be employed to assure that the developers comply with the requirements of this standard.

3.13 Records Collection, Maintenance, and Retention (Section 13 of the SQAP).
This section shall identify the SQA documentation to be retained, shall state the methods and facilities to be used to assemble, safeguard, and maintain this documentation, and shall designate the retention period.

3.14 Training (Section 14 of the SQAP).
This section shall identify the training activities necessary to meet the needs of the SQAP.

3.15 Risk Management (Section 15 of the SQAP).
This section shall specify the methods and procedures employed to identify, assess, monitor, and control areas of risk arising during the portion of the software life cycle covered by the SQAP.