

IEEE Standard for Software Maintenance

Sponsor

**Software Engineering Standards Subcommittee
of the
IEEE Computer Society**

Approved December 3, 1992

IEEE Standards Board

Abstract: The process for managing and executing software maintenance activities is described.

Keywords: maintenance, software, software maintenance

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

Copyright © 1993 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1993. Printed in the United States of America

ISBN 1-55937-279-6

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.

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

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

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

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

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

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

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.

Introduction

(This introduction is not a part of IEEE Std 1219-1992, IEEE Standard for Software Maintenance.)

This standard describes the process for managing and executing software maintenance activities. Clause 1 lists references to other standards useful in applying this standard. Clause 2 provides a set of definitions and acronyms that are either not found in other standards, or have been modified for use with this standard. Clause 3 contains required information pertaining to the software maintenance process. In order to be in compliance with this standard, clause 3 must be adhered to.

Annex A, Maintenance guidelines, contains additional information that is not required for compliance. Topics in this annex include: the source of maintenance forms discussed in the standard, validation and verification (V&V), software quality assurance, risk assessment, safety, security, software configuration management (SCM), metrics, software replacement policy, and the maintenance process. Annex B, Supporting maintenance technology, includes the topics re-engineering, reverse engineering, reuse, maintenance planning, impact analysis, and software tools.

The audience for which this standard is intended consists of software development managers, maintainers, software quality assurance personnel, SCM personnel, programmers, and researchers.

This standard was developed by a working group consisting of the following members who attended two or more meetings, provided text, or submitted comments on more than two drafts of the standards.

D. Vera Edelstein, *Chair*
Salvatore Mamone, *Secretary*

Shawn A. Bohner
Rita Costello
Robert Dufresne
Frank Douglas
Gregory Haines

Alan Huguley
Moisey Lerner
Julio Gonzalez Sanz
Ben Scheff
Carl Seddio

Ahlam Shalhout
Malcolm Slovin
Edwin Summers
Richard Weiss
Ralph Wootton

The following individuals also contributed to the development of the standard by attending one meeting or providing comments on one or two drafts.

Ray Abraham
Joseph Barrett
James E. Cardow
Neva Carlson
Sharon Cobb
Sandra Falcone
Mari Georges

Meinhard Hoffman
T. S. Katsoulakos
Thomas Kurihura
Robert Leif
Suzanne Leif
K. C. Leung

Yuan Liu
John Lord
Kenneth Mauer
Basil Sherlund
Harry Sneed
Harry Trivedi
Bob Walsh

The following persons were on the balloting group that approved this document for submission to the IEEE Standards Board:

Ron Berlack	Robert Kosinski	Hans Schaefer
William Boll	Thomas Kurihara	David Schultz
Fletcher Buckley	Robert Lane	Gregory Schumacher
Kay Bydalek	Boniface Lau	D.M. Siefert
Evelyn Dow	Moisey Lerner	Malcom Slovin
Einar Dragstedt	Ben Livson	Al Sorkowitz
Vera Edelstein	Joseph Maayan	Vijaya Srivastava
Caroline Evans	Kukka Marijarvi	Richard Thayer
John Fendrich	Roger Martin	George Tice
Kirby Fortenberry	Ivano Mazza	Leonard Tripp
Yar Gershkovich	James Perry	Dolores Wallace
David Gustafson	Andrew Sage	William Walsh
John Harauz	Julio Gonzalez Sanz	Robert Wells
John Horch	Stephen Schach	Natalie Yopconka
Roy Ko		Janusz Zalewski

When the IEEE Standards Board approved this standard on December 3, 1992, it had the following membership:

Marco W. Migliaro, *Chair* **Donald C. Loughry, *Vice Chair***
Andrew G. Salem, *Secretary*

Dennis Bodson	Donald N. Heirman	T. Don Michael*
Paul L. Borrill	Ben C. Johnson	John L. Rankine
Clyde Camp	Walter J. Karplus	Wallace S. Read
Donald C. Fleckenstein	Ivor N. Knight	Ronald H. Reimer
Jay Forster*	Joseph Koepfinger*	Gary S. Robinson
David F. Franklin	Irving Kolodny	Martin V. Schneider
Ramiro Garcia	D. N. "Jim" Logothetis	Terrance R. Whittemore
Thomas L. Hannan	Lawrence V. McCall	Donald W. Zipse

*Member Emeritus

Also included are the following nonvoting IEEE Standards Board liaisons:

Satish K. Aggarwal
James Beall
Richard B. Engelman
David E. Soffrin
Stanley Warshaw

Christopher J. Booth
IEEE Standards Project Editor

Contents

CLAUSE	PAGE
1. Overview.....	1
1.1 Scope.....	1
1.2 References.....	1
1.3 Terminology.....	2
1.4 Conventions	3
2. Definitions and acronyms	3
2.1 Definitions.....	3
2.2 Acronyms	4
3. Software maintenance.....	4
3.1 Problem/modification identification, classification, and prioritization	4
3.2 Analysis.....	8
3.3 Design	10
3.4 Implementation	11
3.5 System test	13
3.6 Acceptance test	14
3.7 Delivery.....	16
Annexes	
Annex A	
A.1 Definitions.....	19
A.2 References.....	20
A.3 Maintenance planning.....	20
A.3.1 Determine maintenance effort.....	20
A.3.2 Determine current maintenance process	21
A.3.3 Quantify maintenance effort	21
A.3.4 Project maintenance requirements	21
A.3.5 Develop maintenance plan	22
A.4 Maintenance process.....	23
A.4.1 Problem/modification identification and classification	23
A.4.2 Analysis.....	24
A.4.3 Design	24
A.4.4 Implementation	25
A.4.5 System test	26
A.4.6 Acceptance test	26
A.4.7 Delivery.....	27

CLAUSE	PAGE
A.5	Maintenance forms..... 27
A.6	Verification and validation (V&V)..... 28
A.7	Software quality assurance 28
A.8	Risk assessment 28
A.8.1	External exposure identification 29
A.8.2	Structural exposure analysis 29
A.8.3	Software failure likelihood 29
A.9	Safety 30
A.10	Security 30
A.11	Software configuration management (SCM) 30
A.11.1	Problem/modification identification and classification 31
A.11.2	Analysis..... 31
A.11.3	Design 31
A.11.4	Implementation 31
A.11.5	System testing 32
A.11.6	Acceptance testing 32
A.11.7	Delivery..... 32
A.12	Metrics/measures 33
A.13	Software replacement policy..... 34
Annex B	
B.1	Definitions..... 35
B.2	Re-engineering 35
B.3	Reverse engineering..... 36
B.4	Holistic reusing 36
B.5	Software tools 37

IEEE Standard for Software Maintenance

1. Overview

1.1 Scope

This standard describes an iterative process for managing and executing software maintenance activities. Use of this standard is not restricted by size, complexity, criticality, or application of the software product. This standard uses the process model, depicted in table 2, to discuss and depict each phase of software maintenance. The criteria established apply to both the planning of maintenance for software while under development, as well as the planning and execution of software maintenance activities for existing software products. Ideally, maintenance planning should begin during the stage of planning for software development (see annex clause A.3 for guidance).

This standard prescribes requirements for process, control, and management of the planning, execution, and documentation of software maintenance activities. In totality, the requirements constitute a minimal set of criteria that are necessary and sufficient conditions for conformance to this standard. Users of this standard may incorporate other items by reference or as text to meet their specific needs.

The basic process model includes input, process, output, and control for software maintenance. Metrics/measures captured for maintenance should enable the manager to manage the process and the implementor to implement the process (see table 3). This standard does not presuppose the use of any particular development model (e.g., waterfall, spiral, etc.).

This standard provides additional software maintenance guidance on associated topics in annex A and tools/technology assistance in annex B.

1.2 References

The following standards are directly referenced in this standard. Table 1 provides a cross-reference of IEEE standards that address various topics related to software maintenance. These standards are binding to the extent specified within this standard and are referenced to avoid duplication of requirements.

IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology (ANSI).¹

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

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

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

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

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

IEEE Std 983-1986 (w1992), IEEE Guide for Software Quality Assurance Planning (ANSI).²

¹IEEE publications are available from the Institute of Electrical and Electronics Engineers, Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

²IEEE Std 983-1986 has been withdrawn; however, copies can be obtained from the IEEE Standards Department, IEEE Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

IEEE Std 1012-1986, IEEE Standard for Software Verification and Validation Plans (ANSI).

IEEE Std 1028-1988, IEEE Standard for Software Reviews and Audits (ANSI).

IEEE Std 1042-1987, IEEE Guide to Software Configuration Management (ANSI).

IEEE Std 1058.1-1987, IEEE Standard for Software Project Management Plans (ANSI).

IEEE Std 1074-1991, IEEE Standard for Developing Software Life Cycle Processes (ANSI).

1.3 Terminology

The words *shall* and *must* identify the mandatory (essential) material within this standard. The words *should* and *may* identify optional (conditional) material. The terminology in this standard is based on IEEE Std 610.12-1990.³ New terms and modified definitions as applied in this standard are included in clause 2.

Table 1—The relationship of IEEE software engineering standards to IEEE Std 1219-1992

Relationship		IEEE standard
Process	Problem ID/classification	
	Analysis	830-1984, 1074-1991
	Design	830-1984, 1016-1987, 1074-1991
	Implementation	1008-1987, 1074-1991
	System testing	829-1983, 1074-1991, 1028-1988, 1012-1986
	Acceptance testing	1074-1991, 1012-1986
	Delivery	
Control	Problem ID/classification	
	Analysis	830-1984
	Design	830-1984, 1016-1987
	Implementation	829-1983, 1008-1987
	System testing	829-1983, 1028-1988, 1012-1986
	Acceptance testing	829-1983, 1028-1988, 1012-1986
	Delivery	1063-1987
Management	Configuration management	828-1990, 1042-1987
	Measurement/metrics	982.1-1988, 982.2-1988
	Planning	829-1983, 1012-1986, 1058-1987
	Tools/techniques	
	QA	730-1989, 983-1986 (w)1992)
	Risk assessment	730-1989, 982.2-1988
	Safety	
	Security	

³Information on references can be found in 1.2.

1.4 Conventions

The conventions used in each figure depicting a maintenance phase are shown in figure 1.

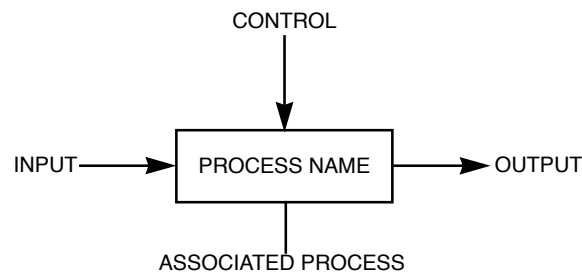


Figure 1—Conventions

The term *associated processes* refers to external processes that are defined in other standards; i.e., software quality assurance, software configuration management (SCM), and verification and validation (V&V). The term *associated processes* also refers to the metrics process illustrated within this document.

2. Definitions and acronyms

2.1 Definitions

The definitions listed below establish meaning in the context of this standard. These are contextual definitions serving to augment the understanding of software maintenance activities as described within this standard. Other definitions can be found in IEEE Std 610.12-1990.

2.1.1 adaptive maintenance: Modification of a software product performed after delivery to keep a computer program usable in a changed or changing environment.

2.1.2 corrective maintenance: Reactive modification of a software product performed after delivery to correct discovered faults.

2.1.3 customer: The person, or persons, for whom the product is intended, and usually (but not necessarily) who decides the requirements.

2.1.4 emergency maintenance: Unscheduled corrective maintenance performed to keep a system operational.

2.1.5 interoperability testing: Testing conducted to ensure that a modified system retains the capability of exchanging information with systems of different types, and of using that information.

2.1.6 modification request (MR): A generic term that includes the forms associated with the various trouble/problem-reporting documents (e.g., incident report, trouble report) and the configuration change control documents [e.g., software change request (SCR), IEEE Std 1042-1987].

2.1.7 perfective maintenance: Modification of a software product after delivery to improve performance or maintainability.

2.1.8 project: A subsystem that is subject to maintenance activity.

2.1.9 regression test: Retesting to detect faults introduced by modification.

2.1.10 repository: (A) A collection of all software-related artifacts (e.g., the software engineering environment) belonging to a system. (B) The location/format in which such a collection is stored.

2.1.11 reverse engineering: The process of extracting software system information (including documentation) from source code.

2.1.12 software maintenance: Modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment.

2.1.13 system: A set of interlinked units organized to accomplish one or several specific functions.

2.1.14 user: The person or persons operating or interacting directly with the system.

2.2 Acronyms

The following acronyms are referred to in this standard:

SCM	Software configuration management
CSA	Configuration status accounting
FCA	Functional configuration audit
PCA	Physical configuration audit
PDL	Program design language
SQA	Software quality assurance
SCR	System/software change request
V&V	Verification and validation
VDD	Version description document

3. Software maintenance

This standard defines changes to software process through a defined maintenance process that includes the following phases:

- a) Problem/modification identification and classification
- b) Analysis
- c) Design
- d) Implementation
- e) Regression/system testing
- f) Acceptance testing
- g) Delivery

These phases are graphically depicted in table 2. Software maintenance factors in table 3 are the entities qualified by the associated metrics/measures identified for each phase.

3.1 Problem/modification identification, classification, and prioritization

In this phase, software modifications are identified, classified, and assigned an initial priority ranking. Each modification request (MR) shall be evaluated to determine its classification and handling priority. Classification shall be identified from the following maintenance types: corrective, adaptive, perfective, and emergency. Metrics/measures and associated factors identified for this phase should be collected and reviewed at appropriate intervals (see table 3 and IEEE Std 982.1-1988 and 982.2-1988).

Table 2—Process model for software maintenance

	Problem identification	Analysis	Design	Implementation	System test	Acceptance test	Delivery
Input	MR	Project/system document Repository information Validated MR	Project/system document Source code Databases Analysis phase output	Source code Product/system document Results of design phase	Updated software documentation Test readiness review report Updated system	Test readiness review report Fully integrated system Acceptance test •Plans •Cases •Procedures	Tested/accepted system
Process	Assign change number Classify Accept or reject change Preliminary magnitude estimate Prioritize	Feasibility analysis Detailed analysis Redocument, if needed	Create test cases Revise •Requirements •Implementation plan	Code Unit test Test readiness review	Functional test Interface testing Regression testing Test readiness review	Acceptance test Interoperability test	PCA Install Training
Control	Uniquely identify MR Enter MR into repository	Conduct technical review Verify •Test strategy •Documentation is updated Identify safety & security issues	Software inspection/review Verify design	Software inspection/review Verify •CM control of software •Traceability of design	CM control of •Code •Listings •MR •Test documentation	Acceptance test Functional audit Establish baseline	PCA VDD
Output	Validated MR Process determinations	Feasibility report Detailed analysis report Updated requirements Preliminary modification list Implementation plan Test strategy	Revised •Modification list •Detail analysis •Implementation plan Updated •Design baseline •Test plans	Updated •Software •Design documents •Test documents •User documents •Training material Test readiness review report	Tested system Test reports	New system baseline Acceptance test report FCA report	PCA report VDD
Metrics	See table 3						

Table 3—Process model metrics for software maintenance

	Problem identification	Analysis	Design	Implementation	System test	Acceptance test	Delivery
Factors	Correctness Maintainability	Flexibility Traceability Reusability Usability Maintainability Comprehensibility	Flexibility Traceability Reusability Testability Maintainability Comprehensibility Reliability	Flexibility Traceability Maintainability Comprehensibility Reliability	Flexibility Traceability Verifiability Testability Interoperability Comprehensibility Reliability	Flexibility Traceability Interoperability Testability Comprehensibility Reliability	Completeness Reliability
Metrics	# of omissions on MR # of MR submittals # of duplicate MRs Time expended for problem validation	Requirement changes Documentation error rates Effort per function area Elapsed time Test plans and procedure changes Error rates generated by priority and type Number of lines of code added, deleted, modified, tested Number of applications	S/W complexity Design changes Effort per function area Elapsed time Test plans and procedure changes Error rates generated by priority and type Number of lines of code added, deleted, modified, tested Number of applications	Volume/functionality (function points or SLOC) Error rates generated by priority & type	Error rates by priority & type • Generated • Corrected	Error rates by priority & type • Generated • Corrected	Documentation changes (i.e., version description documents, training manuals, operation guidelines)

3.1.1 Input

Input for the problem/modification identification and classification phase shall be an MR.

3.1.2 Process

If a modification to the software is required, the following determinative activities must occur within the maintenance process:

- a) Assign an identification number
- b) Classify the type of maintenance
- c) Analyze the modification to determine whether to accept, reject, or further evaluate
- d) Make a preliminary estimate of the modification size/magnitude
- e) Prioritize the modification
- f) Assign MR to a block of modifications scheduled for implementation

Figure 2 summarizes the input, process, control, and output for the problem/modification identification and classification phase of maintenance. For additional information, see also A.4.1.

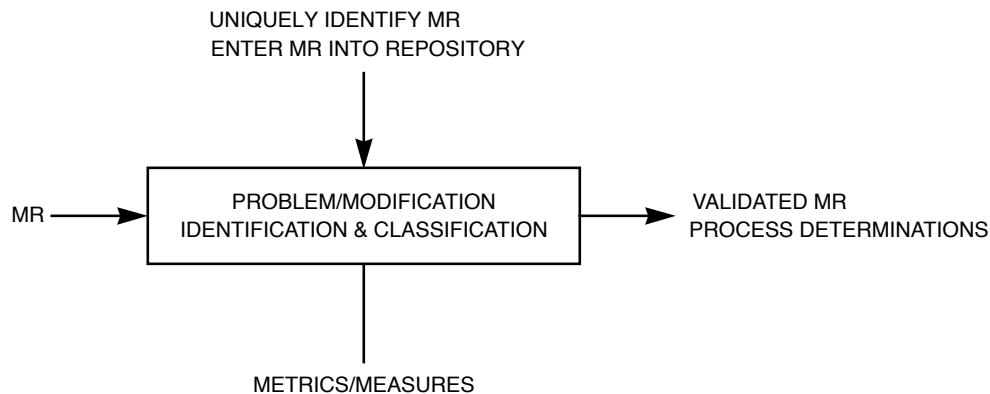


Figure 2—Problem/modification identification and classification phase

3.1.3 Control

MR and process determinations shall be uniquely identified and entered into a repository. See also A.11.1 for guidance.

3.1.4 Output

The output of this process shall be the validated MR and the process determinations that were stored in a repository. The repository shall contain the following items:

- a) Statement of the problem or new requirement
- b) Problem or requirement evaluation
- c) Classification of the type of maintenance required
- d) Initial priority
- e) Verification data (for corrective modifications)
- f) Initial estimate of resources required to modify the existing system

3.2 Analysis

The analysis phase shall use the repository information and the MR validated in the modification identification and classification phase, along with system and project documentation, to study the feasibility and scope of the modification and to devise a preliminary plan for design, implementation, test, and delivery. Metrics/measures and associated factors identified for this phase should be collected and reviewed at appropriate intervals (see table 3 and IEEE Stds 982.1-1988 and 982.2-1988).

Figure 3 summarizes the input, process, control, and output for the analysis phase of maintenance. For additional guidance, see A.4.2.

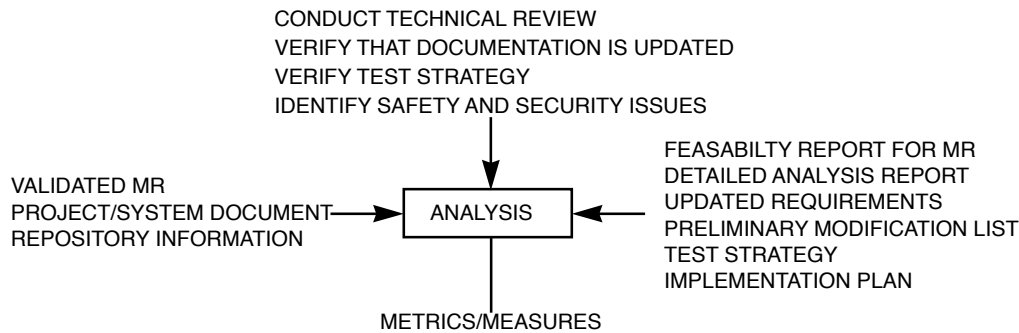


Figure 3—Analysis phase

3.2.1 Input

The input to the analysis phase of the maintenance process shall include the following:

- a) Validated MR
- b) Initial resource estimate and other repository information
- c) Project and system documentation, if available

3.2.2 Process

Analysis is an iterative process having at least two components: (a) a feasibility analysis, and (b) a detailed analysis. If the documentation is not available or is insufficient and the source code is the only reliable representation of the software system, reverse engineering is recommended (see annex B for guidance).

3.2.2.1 Feasibility analysis

A feasibility analysis shall be performed for MR and a feasibility report (FR) shall be prepared. This FR should contain the following:

- a) Impact of the modification
- b) Alternate solutions, including prototyping
- c) Analysis of conversion requirements
- d) Safety and security implications
- e) Human factors
- f) Short-term and long-term costs
- g) Value of the benefit of making the modification