IEEE Standard for Software Verification and Validation Plans

IEEE Standards Board
Approved November 14, 1986
Reaffirmed September 17, 1992

American National Standards Institute
Approved February 10, 1987

Sponsor
Software Engineering Technical Committee of the
of the
IEEE Computer Society

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 once 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
345 East 47th Street
New York, NY 10017 USA

Foreword

(This Foreword is not a part of IEEE Std 1012-1986, IEEE Standard for Software Verification and Validation Plans.)

This standard provides uniform and minimum requirements for the format and content of Software Verification and Validation Plans (SVVPs). Performing software verification and validation (V&V) as defined in this standard provides for a comprehensive evaluation throughout each phase of the software project to help ensure that:

1) Errors are detected and corrected as early as possible in the software life cycle
2) Project risk, cost, and schedule effects are lessened
3) Software quality and reliability are enhanced
4) Management visibility into the software process is improved
5) Proposed changes and their consequences can be quickly assessed

This standard applies to both critical and noncritical software.

1) For critical software, this standard:
   a) Requires that minimum V&V tasks, inputs, and outputs specified in this standard be included in SVVPs
   b) Permits the SVVP to be extended by selecting additional V&V tasks from the optional tasks described in this standard or new tasks identified by the V&V planner
2) For noncritical software, this standard:
   a) Recommends the use of minimum V&V tasks
   b) Permits the SVVP to be tailored to V&V efforts by selecting any of the V&V tasks (minimum, optional, new)

This standard applies to all phases of the software life cycle from the Concept Phase to the Operation and Maintenance Phase. Maximum benefits are derived when V&V is started early in the software life cycle, preferably at project initiation during the Concept Phase. Benefits can be derived for software already in development or in the Operation and Maintenance Phase if the V&V requirements from this standard are invoked consistent with cost and schedule constraints. When V&V is invoked for software in development or in operation and maintenance, required V&V inputs may not exist. Under these conditions, this standard permits the V&V tasks to be tailored to adjust for missing V&V inputs. In some instances, this may require the generation of appropriate software documentation.

V&V is performed in parallel with software development. Each V&V life-cycle phase ends when the V&V tasks of that phase are completed and the software development products are determined to be adequate. V&V life-cycle phases may overlap as activities of the new life-cycle phase are beginning and activities of the previous life-cycle phase are completing.

V&V tasks are iterative: as changes are made to the software product, selected V&V tasks from the previous life-cycle phases are reperformed, or additional V&V tasks are performed to address the changes. V&V tasks are reperformed if errors are discovered in the V&V inputs or outputs. The complexity and scope of changes determine the level of detail covered by the iteration. The SVVP identifies the criteria for performing the iterative V&V tasks.

This standard defines a V&V reporting structure by identifying format and content of the Software Verification and Validation Report (SVVR). The standard for Software Quality Assurance Plans (SQAP, ANSI/IEEE Std-730-1984) requires the SVVR to include both V&V and other quality assurance results. The SVVR defined here is flexible enough to include both types of results. The interim phase reports, final summary

report, and optional SQAP-related activity reports defined by the SVVP provide visibility into the development and V&V processes.

This standard considers both the software and its system or operating environment. It can be used where software is the system or where software is part of a larger system. V&V should have a total system scope (that is, including interfaces between software, hardware, and operators) during the product life cycle. Embedded software is strongly coupled to hardware and other subsystems, and requires a system-level SVVP.

This standard was written to provide direction to organizations responsible for preparing or assessing a Software Verification and Validation Plan. This standard may be used by project management software developers, quality assurance organizations, purchasers, end users, maintainers, and verification and validation organizations. If V&V is performed by an independent group, then the SVVP should specify the criteria for maintaining the independence of the V&V effort from the software development and maintenance efforts.

Suggestions for the improvement of this standard will be welcomed. They should be sent to

Secretary
IEEE Standards Board
Institute of Electrical and Electronics Engineers, Inc
345 East 47th Street
New York, New York 10017

Introduction

The working group that developed this standard consisted of the following members:

Roger U. Fujii, *Chair*
Doug McMann, *Vice Chair*
Dolores R. Wallace, *Secretary*

Julian O. Blosiu	Michael Edwards	David Schultz
Martha Branstad	John Horch	David M. Siefert
Fletcher J. Buckley	Ralph A. Kubek	Hugh Spillane
Francois Coallier	Joyce Lewis	David Turner
James A. Darling	Dennis E. Nickle	William S. Turner
Taz Daughtrey	Larry E. Nitszche	Adam Valentine
David C. Doty	A.E. Nountor	Jay W. Wiley
Sam Dugdale	Don J. Robbins	Andrea Williams
William Dupras	Hans Schaefer	Laurence Wilson

When the IEEE Standards Board approved this standard on September 18,1986, it had the following membership:

John E. May, *Chair*
Irving Kolodny, *Vice Chair*
Sava I. Sherr, *Secretary*

James H. Beall	Joseph L. Koepfinger*	Martha Sloan
Fletcher J. Buckley	Edward Lohse	Oley Wanaselja
Paul G. Cummings	Lawrence V. McCall	J. Richard Weger
Donald C. Fleckenstein	Donald T. Michael*	William B. Wilkens
Jay Forster	Marco W. Migliaro	Helen M. Wood
Daniel L. Goldberg	Stanley Owens	Charles J. Wylie
Kenneth D. Hendrix	John P. Riganati	Donald W. Zipse
Irvin N. Howell	Frank L. Rose	
Jack Kinn	Robert E. Rountree	

*Member emeritus

The standard was approved by the Software Engineering Standards Subcommittee of the IEEE Computer Society. At the time it approved this standard, the Ballot Group had the following membership:

John W. Horch, *Chair*

A. Frank Ackerman	Homer C. Carney	William P. Dupras
Jagdish Agrawal	C.L. Carpenter, Jr	Robert E. Dwyer
Tom. Armbruster	Ronald R. Carter	Mary Eads
Richard L. Aurbach	R.L. Chilavsky	John D. Earls
James Baldo, Jr	Tsun S. Chow	Michael Edwards
H. Jack Barnard	Jung K. Chung	L.G. Egan
Roy W.. Bass	Peter Coad, Jr	Wolfgang Ehrenberger
Leo Beltracchi	Francois Coallier	Steven R. Eisen
Yechiel Ben-Naftau	Sharon R. Cobb	Caroline L. Evans
H.R. Berlack	Christopher M. Cooke	David W. Favor
J. Emmett Black	Gail A. Cordes	John Fendrich
Michael A. Blackledge	A.J. Cote	Robert G. Ferreol
Ronald M. Blair	Patricia W. Daggett	Glenn S. Fields
Walter DuBlanica	James A. Darling	Gordon Force
Kevin W. Bowyer	George D. Darling	Julian Forster
Ingar Brauti	Taz Daughtrey	C.R. Frederick
Michael F. Brennter	P.A. Denny	Carl Friedlander
Kathleen L. Briggs	James H. Dobbins	Richard C. Fries
William L. Bryan	David C. Doty	Ismael Fuentes
Fletcher J.. Buckley	Einar Dragstedt	Roger U. Fujii
Douglas. Burt	Robert Dunn	Michel Galinier

Leonard B. Gardner
David Gelperin
J. Kaye Grau
Andres Grebene
Thomas Griest
James L. Gildersleeve
Shirley A. Gloss-Soler
Victor M. Guarnera
Lawrence M. Gunther
David A. Gustafson
Russell Gustin
Howard Hamer
Harry E. Hansen
Robert M. Haralick
Hans Ludwig Hausen
Clark M. Hay
Herb Hecht
Terry L. Hengl
Charles P. Hollocker
John W. Horch
Cheng Hu
Peter L. Hung
Shang-Sheng Jeng
Laurel Kaleda
Constantine Kaniklidis
Myron S. Karasik
Adi Kasad
Ron Kenett
R.A. Kessler
Shaye Koenig
Shaye Koenig
Joseph A. Krupinski
Joan Kundig
Tom Kurihara
Lak Ming Lam
John B. Lane
Robert A. Lane
William P. LaPlant
Greg Larsen
John A. Latimer
Paul Lebertz
J.A.N. Lee
Leon S. Levy
F.C. Lim
Bertil Lindberg
Gary Lindsay
David P. Linssen
Steven Litvintchouk
John M. Long
John K. Lowell
Bill Macre
Harold T. Maguire
Andy Mahindru
Kartik C. Majumdar
Henry A. Malec
Paulo C. Marcondes
Stuart Marcotte
Philip C. Marriott
Nicholas L. Marselos
Roger J. Martin
Paul Mauro
L.J. Mazlack
Ivano Mazza
J.A. McCall
Paul E. McKenney
Jack McKissick
Stanley E. McQueen
Glen A. Meldrum
Mordechai Ben Menachen
Belden Menkus
Charles S. Mooney
Gary Moorhead
Gene T. Morun
David G. Mullens
Myron L. Nack
Hironobu Nagano
Saied Najafi
G.R. Neidhart
Dennis E. Nickle
Perry R. Nuhn
J.H. Obbink
Wilma Osborne
D.J. Ostrom
Thomas D. Parrish
William E. Perry
Donald J. Pfeiffer
Harpal S. Phama
Robert M. Poston
Peter Prinzivalli
Thomas S. Radi
Jock Rader
Meir Razy
John Reddan
Larry K. Reed
Matthias F. Reese
T.D. Regulinski
Paul Renaud
Hom Sack
J. Gonzales Sanz
Lawrence R. Satz
Franz P. Schauer
Max J. Schindler
Norman Schneidewind
Wolf A. Schnoege
Robert Schueppert
David J. Schultz
Gregory D. Schumacher
Leonard W. Seagren
Craig L. Shermer
Robert W. Shillato
Victor Shtern
David M. Siefert
David J. Simkins
Jacob Slonim
Jean-Christopher Slucki
Marion P. Smith
Harry M. Sneed
Al Sorkowitz
Hugh B. Spillane
Lee Sprague
G. Wayne Staley
Vegard Stuan
Alan N. Sukert
William G Sutcliffe
Robert A. Symes
Richard H. Thayer
Paul U. Thompson
Michael H. Thursby
George Tice
R.L. Van Tilburg
Terrence L. Tillmanns
Lawrence F. Tracey
Henry J. Trochesset
Robert Troy
C.L. Troyanowski
Dana L. Ulery
David Usechak
P.M. Vater
Osmo Vikman
R. Wachter
Dolores R. Wallace
Thomas J. Walsh
William M. Walsh
Roger Warburton
Robert Werlwas
Charles J. Wertz
N.P. Wilburn
Patrick J. Wilson
Paul A. Willis
Walter L. Whipple
Theodore J. Wojcik
Paul Wolfgang
Tom Worthington
W. Martin Wong
Dennis L. Wood
Charles Wortz
A.W. Yonda
Natalie C. Yopconka
Michael E. York
Janusz Zalewski
Donald J. Zeleny
Marvin Zelkowitz
Hugh Zettel
Peter F. Zoll

The following organizations supported employee participation in the development of this standard:

ACEx Technology
Army Computer Systems Command
AT&T Technologies
Babcock & Wilcox
Bechtel Power Corporation

Bell Canada
The Boeing Company
Booz Allen Hamilton
Central Institute For Industrial Research
Computer Science Corporation
Data Logic
E-Systems
Gemini
Hewlett Packard
Jet Propulsion Laboratory
Johns Hopkins University Applied Physics Laboratory
Logicon, Inc
Lucas Micro, Ltd
National Bureau of Standards
NCR Corporation
RCA
STC - Standard Telecommunications
Teledyne Brown Engineering
Televideo Systems
TRW
U.S. Department of Agriculture
U.S. Department of Transportation
Veatch, Rich, & Nadler
Walt Disney World
Worldwide Service Technologies, Ltd

Contents

IEEE Standard for Software Verification and Validation Plans

1. Scope and References

1.1 Scope

This standard has a threefold purpose:

1) To provide, for both critical and noncritical software, uniform and minimum requirements for the format and content of Software Verification and Validation Plans (SVVPs)
2) To define, for critical software, specific minimum verification and validation (V&V) tasks and their required inputs and outputs that shall be included in SVVPsi
3) To suggest optional V&V tasks to be used to tailor SVVPs as appropriate for the particular V&V effort

This standard requires that an SVVP be written for both critical and noncritical software. Critical software is software in which a failure could have an impact on safety or could cause large financial or social losses.

This SVVP shall include V&V tasks to:

1) Verify that the products of each software life-cycle phase:
   a) Comply with previous life-cycle phase requirements and products (for example, for correctness, completeness, consistency, accuracy)
   b) Satisfy the standards, practices, and conventions of the phase
   c) Establish the proper basis for initiating the next life-cycle phase activities
2) Validate that the completed end product complies with established software and system requirements.

For critical software, this standard requires that minimum V&V tasks and their inputs and outputs be included in all SVVPs. For noncritical software, this standard does not specify minimum required V&V tasks; however, all other requirements of this standard shall be satisfied. This standard does recommend that the minimum V&V tasks for critical software also be employed for noncritical software.

This standard defines optional V&V tasks that permit V&V planners to tailor an SVVP for a V&V effort. For critical software, the minimum tasks may be supplemented with tasks selected from the optional tasks. For noncritical software, tasks may be selected from the minimum and optional tasks. Additional tasks identified by the user of this standard may be included in the SVVP for critical and noncritical software.

The life cycle used in this standard serves as a model and consists of the following life-cycle phases:

1) Concept
2) Requirements
3) Design
4) Implementation
5) Test

6) Installation and checkout
7) Operation and maintenance

Compliance with this standard does not require use of the life-cycle model presented here. If a different model is used, the SVVP shall include cross-references to this standard's life cycle and to the V&V tasks, inputs, and outputs specified here for each life-cycle phase.

This standard requires that the following be defined for each phase:

1) Verification and validation tasks
2) Methods and criteria
3) Inputs and outputs
4) Schedule
5) Resources
6) Risks and assumptions
7) Roles and responsibilities

This standard requires a management effort that encompasses all life-cycle phases. The management section of the SVVP defines information necessary to manage and perform the V&V effort, and to coordinate V&V with other aspects of the project. The standard requires the SVVP to specify how the V&V results shall be documented in the Software Verification and Validation Report (SVVR).

When this standard is invoked for existing software, the SVVP shall describe how V&V will be performed when required inputs do not exist. The standard does not prohibit the incorporation of additional content into an SVVP.

The SVVP standard derives its scope from ANSI/IEEE Std 730-1984 [2].[1] The SVVP standard may be applied in conjunction with, or independent of, other IEEE software engineering standards. This standard uses the definitions of ANSI/IEEE Std 729-1983 [1]. This SVVP standard contains V&V configuration analysis tasks that, in part or in whole, are reflected in ANSI/IEEE Std 828-1983 [3]. Test documentation is compatible with that in ANSI/IEEE Std 829-1983 [4].

1.2 References

This standard shall be used in conjunction with the following publications:

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

[2] ANSI/IEEE Std 730-1984, IEEE Standard for Software Quality Assurance Plans.

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

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

[1]Numbers in brackets correspond to those of the references in 1.2 of this standard.

[2]ANSI documents are available from the Sales Department, American National Standards Institute, 1430 Broadway, New York, NY 10018.

2. Conventions, Definitions, and Acronyms

2.1 Conventions

The use of the term *documentation* rather than *document* indicates that the information may exist in several documents or may be embedded within a document addressing more than one subject.

2.2 Definitions

The following terms, including those defined in other standards, are used as indicated in this standard.

acceptance testing: Formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system. (See ANSI/IEEE Std 729-1983 [1].)

anomaly: Anything observed in the documentation or operation of software that deviates from expectations based on previously verified software products or reference documents. A critical anomaly is one that must be resolved before the V&V effort proceeds to the next life-cycle phase.

component testing: Testing conducted to verify the implementation of the design for one software element (for example, unit, module) or a collection of software elements.

concept phase: The initial phase of a software development project, in which user needs are described and evaluated through documentation (for example, statement of needs, advance planning report, project initiation memo, feasibility studies, system definition documentation, regulations, procedures, or policies relevant to the project).

critical software: Software whose failure could have an impact on safety, or could cause large financial or social loss.

design phase: The period of time in the software life cycle during which the designs for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements. (See ANSI/IEEE Std 729-1983 [1].)

implementation phase: The period of time in the software life cycle during which a software product is created from design documentation and debugged. (See ANSI/IEEE Std 729-1983 [1].)

installation and checkout phase: The period of time in the software life cycle during which a software product is integrated into its operational environment and tested in this environment to ensure that it performs as required. (See ANSI/IEEE Std 729-1983 [1].)

integration testing: An orderly progression of testing in which software elements, hardware elements, or both are combined and tested until the entire system has been integrated. (See ANSI/IEEE Std 729-1983 [1].)

life-cycle phase: Any period of time during software development or operation that may be characterized by a primary type of activity (such as design or testing) that is being conducted. These phases may overlap one another; for V&V purposes, no phase is concluded until its development products are fully verified.

minimum tasks: Those V&V tasks applicable to all projects. V&V planning for critical software shall include all such tasks; these tasks are recommended for the V&V of noncritical software.

operation and maintenance phase: The period of time in the software life cycle during which a software

product is employed in its operational environment, monitored for satisfactory performance, and modified as necessary to correct problems or to respond to changing requirements. (See ANSI/IEEE Std 729-1983 [1].)

optional tasks: Those V&V tasks that are applicable to some, but not all, software, or that may require the use of specific tools or techniques. These tasks should be performed when appropriate. The list of tasks provided in Table 2 is not exhaustive.

required inputs: The set of items necessary to perform the minimum V&V tasks mandated within any life-cycle phase.

required outputs: The set of items produced as a result of performing the minimum V&V tasks mandated within any life-cycle phase.

requirements phase: The period of time in the software life cycle during which the requirements, such as functional and performance capabilities for a software product, are defined and documented. (See ANSI/IEEE Std 729-1983 [1].)

software design description: A representation of software created to facilitate analysis, planning, implementation, and decision making. The software design description is used as a medium for communicating software design information, and may be thought of as a blueprint or model of the system.

software requirements specification: Documentation of the essential requirements (functions, performance, design constraints, and attributes) of the software and its external interfaces. (See ANSI/IEEE Std 730-1984 [2].)

software verification and validation plan: A plan for the conduct of software verification and validation.

software verification and validation report: Documentation of V&V results and appropriate software quality assurance results.

system testing: The process of testing an integrated hardware and software system to verify that the system meets its specified requirements. (See ANSI/IEEE Std 729-1983 [1].)

test case: Documentation specifying inputs, predicted results, and a set of execution conditions for a test item. (See ANSI/IEEE Std 829-1983 [4].)

test design: Documentation specifying the details of the test approach for a software feature or combination of software features and identifying the associated tests. (See ANSI/IEEE Std 829-1983 [4].)

test phase: The period of time in the software life cycle in which the components of a software product are evaluated and integrated, and the software product is evaluated to determine whether or not requirements have been satisfied. (See ANSI/IEEE Std 729-1983 [1].)

test plan: Documentation specifying the scope, approach, resources, and schedule of intended testing activities. (See ANSI/IEEE Std 829-1983 [4].)

test procedure: Documentation specifying a sequence of actions for the execution of a test. (See ANSI/IEEE Std 829-1983 [4] .)

validation: The process of evaluating software at the end of the software development process to ensure compliance with software requirements. (See ANSI/IEEE Std 729-1983 [1].)

verification: The process of determining whether or not the products of a given phase of the software development cycle fulfill the requirements established during the previous phase. (See ANSI/IEEE Std 729-1983 [1].)

2.3 Acronyms

The following acronyms appear in this standard:

SDD	Software Design Description
SRS	Software Requirements Specification
SVVP	Software Verification and Validation Plan
SVVR	Software Verification and Validation Report
V&V	Verification and Validation

3. Software Verification and Validation Plan

The Software Verification and Validation Plan (also referred to as the Plan) shall include the sections shown below to be in compliance with this standard. If there is no information pertinent to a section or a required paragraph within a section, the following shall appear below the section or paragraph heading together with the appropriate reason for the exclusion: *This section/paragraph is not applicable to this plan*. Additional sections may be added at the end of the plan as required. Some of the material may appear in other documents. If so, reference to those documents shall be made in the body of the Plan.

Software Verification and Validation Plan Outline

1. Purpose
2. Referenced Documents
3. Definitions
4. Verification and Validation Overview
   - 4.1 Organization
   - 4.2 Master Schedule
   - 4.3 Resources Summary
   - 4.4 Responsibilities
   - 4.5 Tools, Techniques, and Methodologies
5. Life-Cycle Verification and Validation
   - 5.1 Management of V&V
   - 5.2 Concept Phase V&V
   - 5.3 Requirements Phase V&V
   - 5.4 Design Phase V&V
   - 5.5 Implementation Phase V&V
   - 5.6 Test Phase V&V
   - 5.7 Installation and Checkout Phase V&V
   - 5.8 Operation and Maintenance Phase V&V
6. Software Verification and Validation Reporting
7. Verification and Validation Administrative Procedures
   - 7.1 Anomaly Reporting and Resolution
   - 7.2 Task Iteration Policy
   - 7.3 Deviation Policy
   - 7.4 Control Procedures
   - 7.5 Standards, Practices, and Conventions

3.1 Purpose

(Section 1 of the Plan.) This section shall delineate the specific purpose and scope of the Software Verification and Validation Plan, including waivers from this standard. The software project for which the Plan is

being written and the specific software product items covered by the Plan shall be identified. The goals of the verification and validation efforts shall be specified.

3.2 Referenced Documents

(Section 2 of the Plan.) This section shall identify the binding compliance documents, documents referenced by this Plan, and any supporting documents required to supplement or implement this Plan.

3.3 Definitions

(Section 3 of the Plan.) This section shall define or provide a reference to the definitions of all terms required to properly interpret the Plan. This section shall describe the acronyms and notations used in the Plan.

3.4 Verification and Validation Overview

(Section 4 of the Plan.) This section shall describe the organization, schedule, resources, responsibilities, and tools, techniques, and methodologies necessary to perform the software verification and validation.

3.4.1 Organization

(Section 4.1 of the Plan.) This section shall describe the organization of the V&V effort. It shall define the relationship of V&V to other efforts such as development, project management, quality assurance, configuration or data management, or end user. It shall define the lines of communication within the V&V effort, the authority for resolving issues raised by V&V tasks, and the authority for approving V&V products.

3.4.2 Master Schedule

(Section 4.2 of the Plan.) This section shall describe the project life cycle and milestones, including completion dates. It shall summarize the scheduling of V&V tasks and shall describe how V&V results provide feedback to the development process to support project management functions (for example, comments on design review material).

If the life cycle used in the Plan differs from the life-cycle model in the standard, this section shall show how all requirements of the standard are satisfied (for example, cross-reference for life-cycle phases, tasks, inputs, and outputs). When planning V&V tasks, it should be recognized that the V&V process is iterative. The summary of tasks may be in narrative, tabular, or graphic form (for example, Figure 1). The life-cycle model in Figure 1 is a sample model used for this standard.

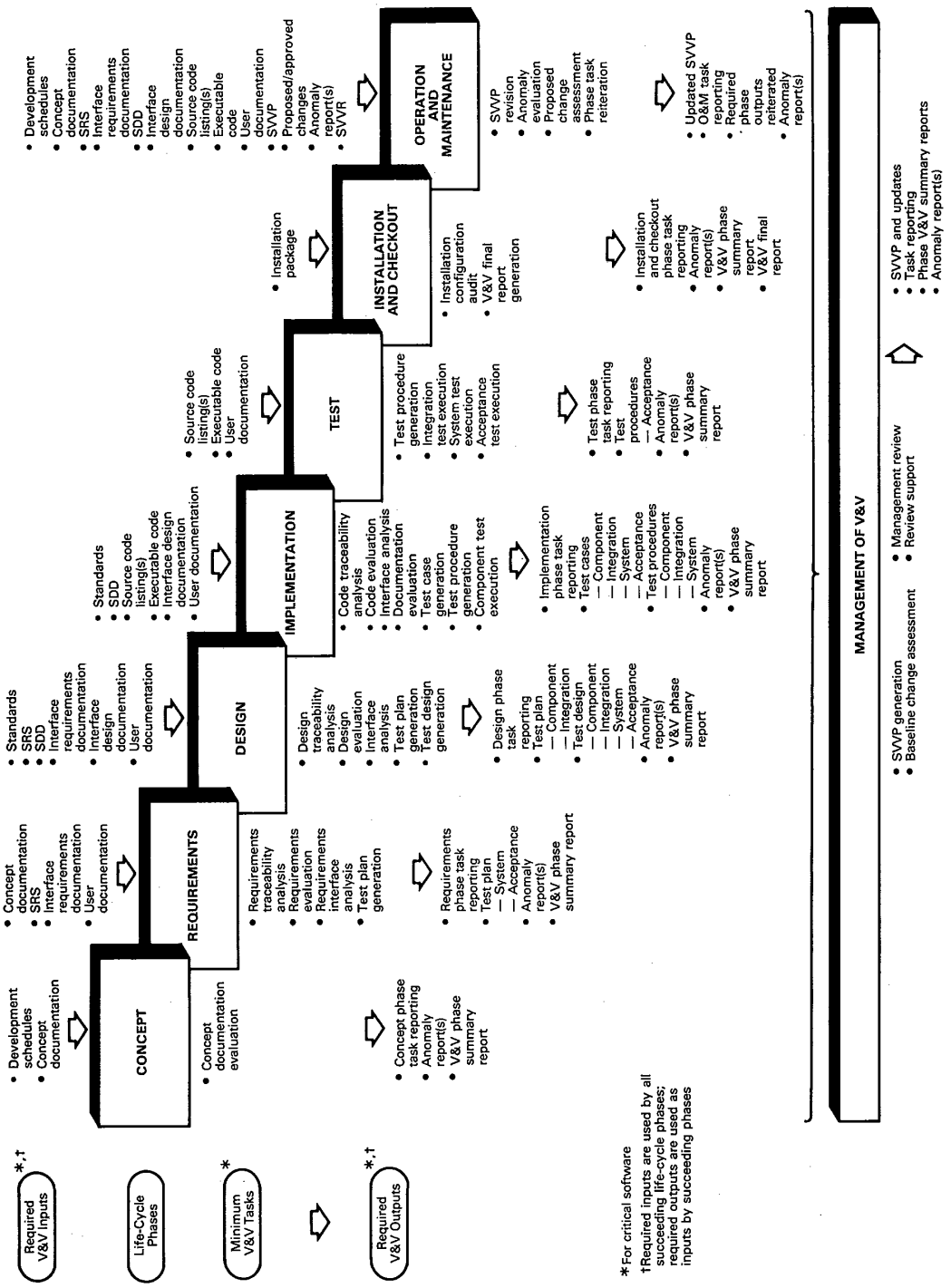

Figure 1 — Software Verification and Validation Plan Overview

3.4.3 Resources Summary

(Section 4.3 of the Plan.) This section shall summarize the resources needed to perform the V&V tasks, including staffing, facilities, tools, finances, and special procedural requirements such as security, access rights, or documentation control.

3.4.4 Responsibilities

(Section 4.4 of the Plan.) This section shall identify the organizational element(s) responsible for performing each V&V tasks. It shall identify the specific responsibility of each element for tasks assigned to more than one element. This section may be a summary of the roles and responsibilities defined in each of the life-cycle phases (see 3.5 of this standard).

3.4.5 Tools, Techniques, and Methodologies

(Section 4.5 of the Plan.) This section shall identify the special software tools, techniques, and methodologies employed by the V&V effort. The purpose and use of each shall be described. Plans for the acquisition, training, support, and qualification for each shall be included. This section may reference a V&V Tool Plan.

3.5 Life-Cycle Verification and Validation

(Section 5 of the Plan.) This section of the Plan shall provide the detailed plan for the V&V tasks throughout the life cycle. The detailed plan (Section 5.1—Management, and Sections 5.2 through 5.8—Life-Cycle Phases) shall address the following topics:

1) *Verification and Validation Tasks*. Identify the V&V tasks for the phase. Describe how each task contributes to the accomplishment of the project V&V goals. For all critical software, the SVVP shall include all minimum V&V tasks for the management of V&V and for each life-cycle phase. Any or all of these minimum tasks may be used for noncritical software. These minimum V&V tasks are referenced in the management and life-cycle phases sections of the standard (3.5.1 through 3.5.8), and are described in Table 1 . The minimum tasks are also consolidated in graphic form in Figure 1 .

   Optional V&V tasks may also be selected to tailor the V&V effort to project needs for critical or noncritical software. Optional V&V tasks are defined in the Appendix and a suggested application for the management of V&V and for each life-cycle phase is presented in Table 2 . The optional V&V tasks identified in this standard may be applicable to some, but not all, critical software. These tasks may require the use of specific tools or techniques. The list in Table 2 is illustrative and not exhaustive. The standard allows for the optional V&V tasks and any others identified by the planner to be used as appropriate.

   Testing requires advance planning that spans several life-cycle phases. Test documentation and its occurrence in specific life-cycle phases are shown in Figure 2 as a recommended approach. To be in compliance with this standard, the test documentation and test execution specified in Figure 2 shall be required. If the V&V planner uses different test documentation or test types (for example, component, integration, system, acceptance) from those in this standard, the SVVP shall contain a mapping of the proposed test documentation and execution to the items shown in Figure 2 . Test planning criteria defined in Table 1 (Tasks 5.3 (4a), 5.3 (4b), 5.4 (4a), 5.4 (4b)) shall be implemented in the test plan, test design(s), test case(s), and test procedure(s) documentation, and shall be validated by test execution.

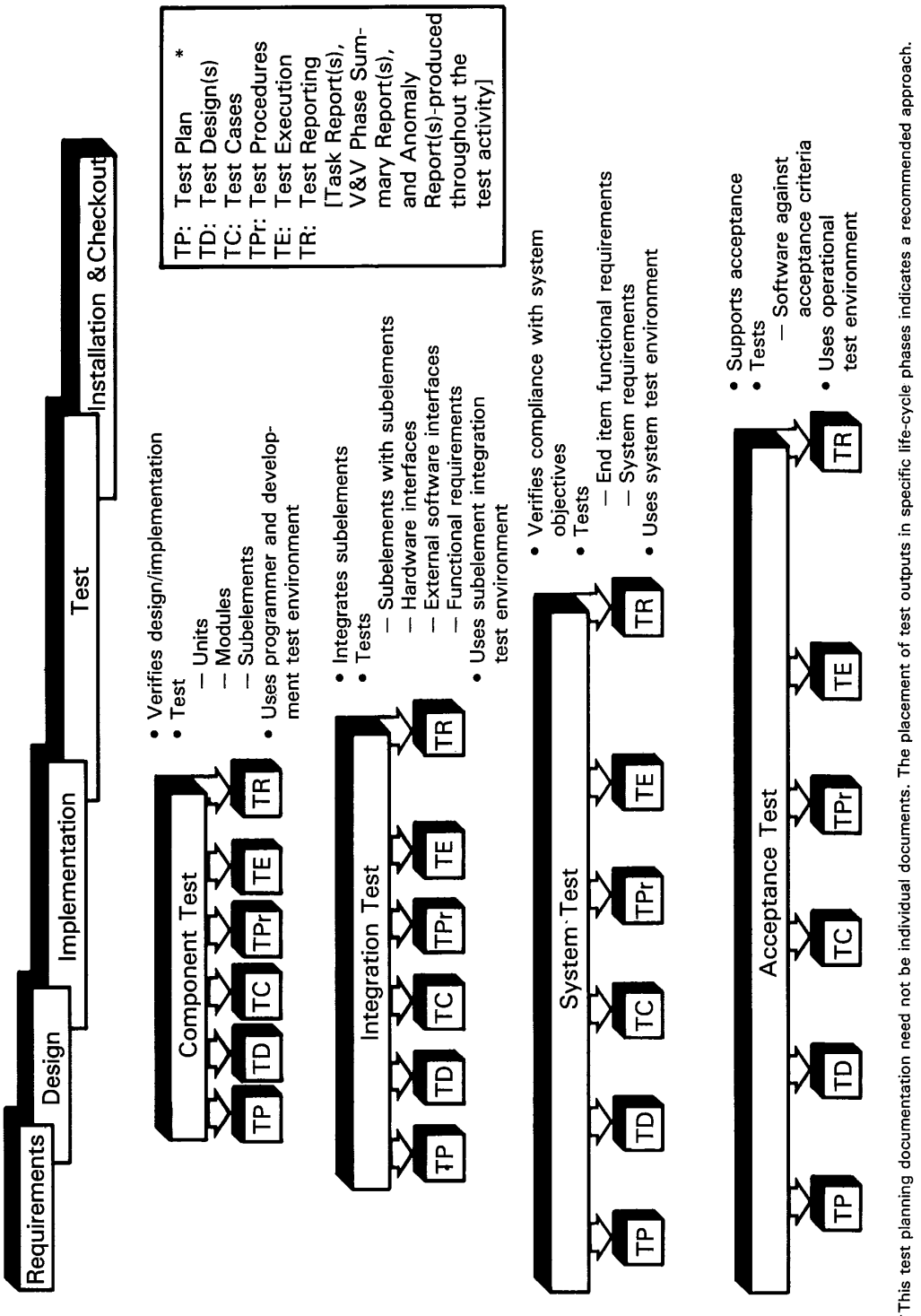

Figure 2 — V&V Test Tasks and Documentation

2) *Methods and Criteria*. Identify the methods and criteria used in performing the V&V tasks. Describe the specific methods and procedures for each task. Define the detailed criteria for evaluating the task results.

3) *Inputs/Outputs*. Identify the inputs required for each V&V task. Specify the source and format of each input. The inputs required for each of the minimum V&V tasks are identified in Table 1. The required inputs are used, as appropriate, by subsequent life-cycle phase V&V tasks. Only the primary inputs are listed in Table 1.

   Identify the outputs from each V&V task. Specify the purpose and format for each output. The outputs from each of the minimum V&V tasks are identified in Table 1.

   The outputs of the management of V&V and of the life-cycle phases shall become inputs to subsequent life-cycle phases, as appropriate.

   Anomaly report(s), task report(s), and phase summary report(s) provide feedback to the software development process regarding the technical quality of each life-cycle phase software product. Each critical anomaly shall be resolved before the V&V effort proceeds to the next life-cycle phase.

4) *Schedule*. Identify the schedule for the V&V tasks. Establish specific milestones for initiating and completing each task, for the receipt of each input, and for the delivery of each output.

5) *Resources*. Identify the resources for the performance of the V&V tasks. Specify resources by category (for example, staffing, equipment, facilities, schedule, travel, training). If tools are used in the V&V tasks, specify the source of the tools, their availability, and other usage requirements (for example, training).

6) *Risks and Assumptions*. Identify the risks and assumptions associated with the V&V tasks, including schedule, resources, or approach. Specify a contingency plan for each risk.

7) *Roles and Responsibilities*. Identify the organizational elements or individuals responsible, for performing the V&V tasks. Assign specific responsibilities for each task to one or more organizational element.

3.5.1 Management of V&V

(Section 5.1 of the Plan.) This section of the Plan shall address the seven topics identified in 3.5 of this standard. The management of V&V spans all life-cycle phases. The software development may be a cyclic or iterative process. The V&V effort shall reperform previous V&V tasks or initiate new V&V tasks to address software changes created by the cyclic or iterative development process. V&V tasks are reperformed if errors are discovered in the V&V inputs or outputs.

For all software, management of V&V shall include the following minimum tasks:

1) Software Verification and Validation Plan (SVVP) Generation
2) Baseline Change Assessment
3) Management Review of V&V
4) Review Support

Table 1 describes the minimum management V&V tasks and identifies the required inputs and outputs. The inputs and outputs required for each V&V task shall include, but not be limited to, those listed in Table 1.

3.5.2 Concept Phase V&V

(Section 5.2 of the Plan.) This section of the Plan shall address the seven topics identified in 3.5 of this standard.

For critical software, Concept Phase V&V shall include the following minimum V&V task:

Concept Documentation Evaluation. Table 1 contains a description of the minimum Concept Phase V&V task and identifies the required inputs and outputs. The inputs and outputs required to accomplish the minimum V&V task shall include, but not be limited to, those listed in Table 1.

3.5.3 Requirements Phase V&V

(Section 5.3 of the Plan.) This section of the Plan shall address the seven topics identified in 3.5 of this standard.

For critical software, Requirements Phase V&V shall include the following minimum tasks:

1) Software Requirements Traceability Analysis
2) Software Requirements Evaluation
3) Software Requirements Interface Analysis
4) Test Plan Generation
    a) System Test
    b) Acceptance Test

Table 1 contains a description of the minimum Requirements Phase V&V tasks and identifies the required inputs and outputs. The inputs and outputs required to accomplish the minimum V&V tasks shall include, but not be limited to, those listed in Table 1.

3.5.4 Design Phase V&V

(Section 5.4 of the Plan.) This section of the Plan shall address the seven topics identified in Section 3.5 of this standard. For critical software, Design Phase V&V shall include the following minimum V&V tasks:

1) Software Design Traceability Analysis
2) Software Design Evaluation
3) Software Design Interface Analysis
4) Test Plan Generation
    a) Component Test
    b) Integration Test
5) Test Design Generation
    a) Component Test
    b) Integration Test
    c) System Test
    d) Acceptance Test

Table 1 contains a description of the minimum Design Phase V&V tasks and identifies the required inputs and outputs. The inputs and outputs required to accomplish the minimum V&V tasks shall include, but not be limited to, those listed in Table 1.

3.5.5 Implementation Phase V&V

(Section 5.5 of the Plan.) This section of the Plan shall address the seven topics identified in 3.5 of this standard.

For critical software, Implementation Phase V&V shall include the following minimum V&V tasks:

1) Source Code Traceability Analysis
2) Source Code Evaluation
3) Source Code Interface Analysis
4) Source Code Documentation Evaluation
5) Test Case Generation
   a) Component Test
   b) Integration Test
   c) System Test
   d) Acceptance Test
6) Test Procedure Generation
   a) Component Test
   b) Integration Test
   c) System Test
7) Component Test Execution

Table 1 contains a description of the minimum Implementation Phase V&V tasks and identifies the required inputs and outputs. The inputs and outputs required to accomplish the minimum V&V tasks shall include, but not be limited to, those listed in Table 1.

3.5.6 Test Phase V&V

(Section 5.6 of the Plan.) This section of the Plan shall address the seven topics identified in 3.5 of this standard.

Testing activities and their interrelationships with previous V&V phases are shown in Figure 2.

For critical software, Test Phase V&V shall include the following minimum V&V tasks:

1) Acceptance Test Procedure Generation
2) Test Execution
   a) Integration Test
   b) System Test
   c) Acceptance Test

Table 1 contains a description of the minimum Test Phase V&V tasks and identifies the required inputs and outputs. The inputs and outputs required to accomplish the minimum V&V tasks shall include, but not be limited to, those listed in Table 1.

3.5.7 Installation and Checkout Phase V&V

(Section 5.7 of the Plan.) This section of the Plan shall address the seven topics identified in Section 3.5 of this standard.

For critical software, Installation and Checkout Phase V&V shall include the following minimum V&V tasks:

1) Installation Configuration Audit
2) Final V&V Report Generation

Table 1 contains a description of the minimum Installation and Checkout Phase V&V tasks and identifies the required inputs and outputs. The inputs and outputs required to accomplish the minimum V&V tasks shall include, but not be limited to, those listed in Table 1.

3.5.8 Operation and Maintenance Phase V&V

(Section 5.8 of the Plan.) This section of the Plan shall address the seven topics identified in Section 3.5 of this standard.

Any modifications, enhancements, or additions to software during this phase shall be treated as development activities and shall be verified and validated as described in 3.5.1 through 3.5.7. These modifications may derive from requirements specified to correct software errors (that is, corrective), to adapt to a changed operating environment (that is, adaptive), or to respond to additional user requests (that is, perfective).

If the software was verified under this standard, the standard shall continue to be followed in the Operation and Maintenance Phase. If the software was not verified under this standard, the V&V effort may require documentation that is not available or adequate. If appropriate documentation is not available or adequate, the SVVP shall comply with this standard within cost and schedule constraints. The V&V effort may generate the missing documentation.

For critical software, Operation and Maintenance Phase V&V tasks shall include the following minimum V&V tasks:

1) Software Verification and Validation Plan Revision
2) Anomaly Evaluation
3) Proposed Change Assessment
4) Phase Task Iteration

Table 1 contains a description of the minimum Operation and Maintenance Phase V&V tasks. The inputs and outputs required to accomplish the minimum V&V tasks shall include, but not be limited to, those listed in Table 1.

3.6 Software Verification and Validation Reporting

(Section 6 of the Plan.) This section shall describe how the results of implementing the Plan will be documented. V&V reporting shall occur throughout the software life cycle. This section of the Plan shall specify the content, format, and timing of all V&V reports. These V&V reports shall constitute the Software Verification and Validation Report (SVVR).

3.6.1 Required Reports

The following reports shall be generated for each software V&V effort.

1) *Task Reporting*. These shall report on the individual V&V phase tasks and shall be issued as necessary. They may document interim results and status. They may be in a format appropriate for technical disclosure (for example, technical reports or memos).
2) *V&V Phase Summary Report*. A Phase Summary Report shall summarize the results of V&V tasks performed in each of the following life-cycle phases: Concept, Requirements, Design, Implementation, Test, and Installation and Checkout. For the Operation and Maintenance Phase, V&V phase summary reports may be either updates to previous V&V phase summary reports or separate documents. Each V&V Phase Summary Report shall contain the following:
   a) Description of V&V tasks performed
   b) Summary of task results
   c) Summary of anomalies and resolution
   d) Assessment of software quality
   e) Recommendations

3) *Anomaly Report*. An Anomaly Report shall document each anomaly detected by the V&V effort. Each Anomaly Report shall contain the following:
   a) Description and location
   b) Impact
   c) Cause
   d) Criticality
   e) Recommendations
4) *V& V Final Report*. The Verification and Validation Final Report shall be issued at the end of the Installation and Checkout phase or at the conclusion of the V&V effort. The Final Report shall include the following information:
   a) Summary of all life-cycle V&V tasks
   b) Summary of task results
   c) Summary of anomalies and resolutions
   d) Assessment of overall software quality
   e) Recommendations

3.6.2 Optional Reports

The following reports are optional.

1) *Special Studies Report*. This report shall describe any special studies conducted during any life-cycle phase. The report shall document technical results and shall include, at a minimum, the following information:
   a) Purpose and objectives
   b) Approach
   c) Summary of results
2) *Other Reports*. These reports shall describe the results of tasks not defined in the SVVP. These other activities and results may include quality assurance results, end user testing results, or configuration and data management status results.

3.7 Verification and Validation Administrative Procedures

(Section 7 of the Plan). This section of the Plan shall describe, at a minimum, the V&V administrative procedures described in 3.7.1 through 3.7.5.

3.7.1 Anomaly Reporting and Resolution

(Section 7.1 of the Plan.) This section shall describe the method of reporting and resolving anomalies, including the criteria for reporting an anomaly, the anomaly report distribution list, and the authority and time lines for resolving anomalies. The section shall define the anomaly criticality levels. Each critical anomaly shall be resolved satisfactorily before the V&V effort can formally proceed to the next life-cycle phase.

3.7.2 Task Iteration Policy

(Section 7.2 of the Plan.) This section shall describe the criteria used to determine the extent to which a V&V task shall be reperformed when its input is changed. These criteria may include assessments of change, criticality, and cost, schedule, or quality effects.

3.7.3 Deviation Policy

(Section 7.3 of the Plan.) This section shall describe the procedures and forms used to deviate from the Plan. The information required for deviations shall include task identification, deviation rationale, and effect on software quality. The section shall define the authorities responsible for approving deviations.

3.7.4 Control Procedures

(Section 7.4 of the Plan.) This section shall identify control procedures applied to the V&V effort. These procedures shall describe how software products and results of software V&V shall be configured, protected, and stored.

These procedures may describe quality assurance, configuration management, data management, or other activities if they are not addressed by other efforts. At a minimum, this section shall describe how SVVP materials shall comply with existing security provisions and how the validity of V&V results shall be protected from accidental or deliberate alteration.

3.7.5 Standards, Practices, and Conventions

(Section 7.5 of the Plan.) This section shall identify the standards, practices, and conventions that govern the performance of V&V tasks, including internal organizational standards, practices, and policies.

Table 1—Required V&V Tasks, Inputs, and Outputs for Life-Cycle Phases[3]

Minimum V&V Tasks	Required Inputs	Required Outputs*
5.1 MANAGEMENT OF V&V		
(1) Software Verification and Validation Plan (SVVP) Generation. Generate SVVP (during Concept Phase) for all life-cycle phases in accordance with this standard based upon available documentation. Include estimate of anticipated V&V activities for Operation and Maintenance Phase. Update SVVP for each life-cycle phase, particularly prior to Operation and Maintenance. Consider SVVP to be a *living document,* and make changes as necessary. A baseline SVVP should be established prior to the Requirements Phase.	Development Schedules Concept Documentation SRS Interface Requirements Documentation SDD Interface Design Documentation Source Code Listing(s) Executable Code User Documentation Proposed Changes	SVVP and Updates
(2) Baseline Change Assessment. Evaluate proposed software changes (for example, anomaly corrections, performance enhancements, requirement changes, clarifications) for effects on previously completed V&V tasks. When changes are made, plan iteration of affected tasks which includes reperforming previous V&V tasks or initiating new V&V tasks to address the software changes created by the cyclic or iterative development process.	Proposed Changes	Updated SVVP
(3) Management Review. Conduct periodic reviews of V&V effort, technical accomplishments, resource utilization, future planning, risk management. Support daily management of V&V phase activities, including technical quality of final and interim V&V reports and results. Review the task and V&V phase summary reports of each life-cycle phase. Evaluate V&V results and anomaly resolution to determine when to proceed to next life-cycle phase and to define changes to V&V tasks to improve the V&V effort.	Development Schedules V&V Outputs	Task Reporting V&V Phase Summary Reports
(4) Review Support. Correlate V&V task results to support management and technical reviews (for example, Software Requirements Review, Preliminary Design Review, Critical Design Review). Identify key review support milestones in SVVP. Schedule V&V tasks to meet milestones. Establish methods to exchange V&V data and results with development effort.	V&V Outputs	Task Reporting Anomaly Report(s)
5.2 CONCEPT PHASE V&V		
(1) Concept Documentation Evaluation. Evaluate concept documentation to determine if proposed concept satisfies user needs and project objectives (for example, performance goals). Identify major constraints of interfacing systems and constraints or limitations of proposed approach. Assess allocation of functions to hardware and software items, where appropriate. Assess criticality of each software item.	Concept Documentation (for example, Statement of Need, Advance Planning Report, Project Initiation Memo, Feasibility Studies, System Definition Documentation, Governing Regulations, Procedures, Policies, and customer acceptance criteria/ requirements)	Task Reporting Anomaly Report(s)
5.3 REQUIREMENTS PHASE V&V		
(1) Software Requirements Traceability Analysis. Trace SRS requirements to system requirements in concept documentation. Analyze identified relationships for correctness, consistency, completeness, accuracy.	Concept Documentation SRS Interface Requirements Documents	Task Reporting Anomaly Report(s)
(2) Software Requirements Evaluation. Evaluate SRS requirements for correctness, consistency, completeness, accuracy, readability, and testability. Assess how well SRS satisfies software system objectives. Assess the criticality of requirements to identify key performance or critical areas of software.	Concept Documentation SRS Interface Requirements Documentation	Task Reporting Anomaly Report(s)

* Outputs from phase tasks are used to develop corresponding V&V phase summary reports and are ongoing inputs to the SVVR. Outputs of V&V tasks become inputs to subsequent life-cycle V&V tasks.

[3] The section numbers referred to in this Table refer to Section 5 of the Plan.

Table 1— (Continued)

Minimum V&V Tasks	Required Inputs	Required Outputs*
5.3 REQUIREMENTS PHASE V&V *(Continued)*		
(3) Software Requirements Interface Analysis. Evaluate SRS with hardware, user, operator, and software interface requirements documentation for correctness, consistency, completeness, accuracy, and readability.	SRS Interface Requirements Documentation	Task Reporting Anomaly Report(s)
(4a) System Test Plan Generation. Plan system testing to determine if software satisfies system objectives. Criteria for this determination are, at a minimum: (a) compliance with all functional requirements as complete software end item in system environment (b) performance at hardware, software, user, and operator interfaces (c) adequacy of user documentation (d) performance at boundaries (for example, data, interface) and under stress conditions. Plan tracing of system end-item requirements to test design, cases, procedures, and execution results. Plan documentation of test tasks and results.	Concept Documentation SRS Interface Requirements Documentation User Documentation	System Test Plan Anomaly Report(s)
(4b) Acceptance Test Plan Generation. Plan acceptance testing to determine if software correctly implements system and software requirements in an operational environment. Criteria for this determination are, at a minimum: (a) compliance with acceptance requirements in operational environment (b) adequacy of user documentation. Plan tracing of acceptance test requirements to test design, cases, procedures, and execution results. Plan documentation of test tasks and results.	Concept Documentation SRS Interface Requirements Documentation User Documentation	Acceptance Test Plan Anomaly Report(s)
5.4 DESIGN PHASE V&V		
(1) Design Traceability Analysis. Trace SDD to SRS and SRS to SDD. Analyze identified relationships for correctness, consistency, completeness, and accuracy.	SRS SDD Interface Requirements Documentation Interface Design Documentation	Task Reporting Anomaly Report(s)
(2) Design Evaluation. Evaluate SDD for correctness, consistency, completeness, accuracy, and testability. Evaluate design for compliance with established standards, practices, and conventions. Assess design quality.	SDD Interface Design Documentation Standards (standards, practices, conventions)	Task Reporting Anomaly Report(s)
(3) Design Interface Analysis. Evaluate SDD with hardware, operator, and software interface requirements for correctness, consistency, completeness, and accuracy. At a minimum, analyze data items at each interface.	SDD Interface Design Documentation	Task Reporting Anomaly Report(s)
(4a) Component Test Plan Generation. Plan component testing to determine if software elements (for example, units, modules) correctly implement component requirements. Criteria for this determination are, at a minimum: (a) compliance with design requirements (b) assessment of timing, sizing, and accuracy (c) performance at boundaries and interfaces and under stress and error conditions (d) measures of test coverage and software reliability and maintainability. Plan tracing of design requirements to test design, cases, procedures, and execution results. Plan documentation of test tasks and results.	SRS SDD Interface Requirements Documentation Interface Design Documentation	Component Test Plan Anomaly Report(s)
(4b) Integration Test Plan Generation. Plan integration testing to determine if software (for example, subelements, interfaces) correctly implements the software requirements and design. Criteria for this determination are, at a minimum: (a) compliance with increasingly larger set of functional requirements at each stage of integration (b) assessment of timing, sizing, and accuracy (c) performance at boundaries and under stress conditions (d) measures of functional test coverage and software reliability. Plan tracing of requirements to test design, cases, procedures, and execution results. Plan documentation of test tasks and results.	SRS SDD Interface Requirements Documentation Interface Design Documentation	Integration Test Plan Anomaly Report(s)

* Outputs from phase tasks are used to develop corresponding V&V phase summary reports and are ongoing inputs to the SVVR. Outputs of V&V tasks become inputs to subsequent life-cycle V&V tasks.

Table 1—(Continued)

Minimum V&V Tasks	Required Inputs	Required Outputs*
5.4 DESIGN PHASE V&V *(Continued)*		
(5) Test Design Generation. Design tests for: (a) component testing (b) integration testing (c) system testing (d) acceptance testing. Continue tracing required by the Test Plan.	SDD Interface Design Documentation User Documentation	Component Test Design(s) Integration Test Design(s) System Test Design(s) Acceptance Test Design(s) Anomaly Report(s)
5.5 IMPLEMENTATION PHASE V&V		
(1) Source Code Traceability Analysis. Trace source code to corresponding design specification(s) and design specification(s) to source code. Analyze identified relationships for correctness, consistency, completeness, and accuracy.	SDD Interface Design Documentation Source Code Listing(s)	Task Reporting Anomaly Report(s)
(2) Source Code Evaluation. Evaluate source code for correctness, consistency, completeness, accuracy, and testability. Evaluate source code for compliance with established standards, practices, and conventions. Assess source code quality.	Source Code Listing(s) Standards (standards, practices, conventions) SDD Interface Design Documentation User Documentation	Task Reporting Anomaly Report(s)
(3) Source Code Interface Analysis. Evaluate source code with hardware, operator, and software interface design documentation for correctness, consistency, completeness, and accuracy. At a minimum, analyze data items at each interface.	Source Code Listing(s) User Documentation	Task Reporting Anomaly Report(s)
(4) Source Code Documentation Evaluation. Evaluate draft code-related documents with source code to ensure completeness, correctness, and consistency.	Source Code Listing(s) User Documentation	Task Reporting Anomaly Report(s)
(5) Test Case Generation. Develop test cases for: (a) component testing (b) integration testing (c) system testing (d) acceptance testing. Continue tracing required by the Test Plan.	SDD Interface Design Documentation Source Code Listing(s)	Component Test Cases Integration Test Cases System Test Cases Acceptance Test Cases Anomaly Report(s)
(6) Test Procedure Generation. Develop test procedures for: (a) component testing (b) integration testing (c) system testing. Continue tracing required by the Test Plan.	SDD Interface Design Documentation Source Code Listing(s) User Documentation	Component Test Procedures Integration Test Procedures System Test Procedures Anomaly Report(s)
(7) Component Test Execution. Perform component testing as required by component test procedures. Analyze results to determine that software correctly implements design. Document and trace results as required by the Test Plan.	Source Code Listing(s) Executable Code SDD Interface Design Documentation	Task Reporting Anomaly Report(s)
5.6 TEST PHASE V&V		
(1) Test Procedure Generation. Develop test procedures for acceptance test. Continue tracing required by the Test Plan.	SDD Interface Design Documentation Source Code Listing(s) User Documentation	Acceptance Test Procedures Anomaly Report(s)

* Outputs from phase tasks are used to develop corresponding V&V phase summary reports and are ongoing inputs to the SVVR. Outputs of V&V tasks become inputs to subsequent life-cycle V&V tasks.

Table 1 —(Continued)

Minimum V&V Tasks	Required Inputs	Required Outputs*
5.6 TEST PHASE V&V *(Continued)*		
(2a) Integration Test Execution. Perform integration testing in accordance with test procedures. Analyze results to determine if software implements software requirements and design and that software components function correctly together. Document and trace results as required by the Test Plan.	Source Code Listing(s) Executable Code	Task Reporting Anomaly Report(s)
(2b) System Test Execution. Perform system testing in accordance with test procedures. Analyze results to determine if software satisfies system objectives. Document and trace all testing results as required by the Test Plan.	Source Code Listing(s) Executable Code User Documentation	Task Reporting Anomaly Report(s)
(2c) Acceptance Test Execution. Perform acceptance testing in accordance with test procedures under formal configuration control. Analyze results to determine if software satisfies acceptance criteria. Document and trace all testing results as required by the Test Plan.	Source Code Listing(s) Executable Code User Documentation	Task Reporting Anomaly Report(s)
5.7 INSTALLATION AND CHECKOUT PHASE V&V		
(1) Installation Configuration Audit. Audit installation package to determine that all software products required to correctly install and operate the software are present, including operations documentation. Analyze all site-dependent parameters or conditions to determine that supplied values are correct. Conduct analyses or tests to demonstrate that installed software corresponds to software subjected to V&V.	Installation Package (for example, Source Code Listing(s), Executable Code, User Documentation, SDD, Interface Design Documentation, SRS, Concept Documentation, Installation Procedures, and Installation Tests)	Task Reporting Anomaly Report(s)
(2) V&V Final Report Generation. Summarize all V&V activities and results, including status and disposition of anomalies in the V&V final report (see 3.6.1 of this standard).	All V&V Phase Summary Report(s)	V&V Final Report
5.8 OPERATION AND MAINTENANCE PHASE V&V		
(1) Software V&V Plan Revision. For software verified and validated under this standard, revise SVVP to comply with new constraints based upon available documentation. For software not verified and validated under this standard, write new SVVP.	Development Schedules Concept Documentation SRS Interface Requirements Documentation SDD Interface Design Documentation Source Code Listing(s) User Documentation Installation Package Proposed Changes	Updated SVVP
(2) Anomaly Evaluation. Evaluate severity of anomalies in software operation. Analyze effect of anomalies on system.	Anomaly Report(s)	Task Reporting
(3) Proposed Change Assessment. Assess all proposed modifications, enhancements, or additions to determine effect each change would have on system. Determine extent to which V&V tasks would be iterated.	Proposed Changes	Task Reporting
(4) Phase Task Iteration. For approved software changes, perform V&V tasks necessary to ensure that: planned changes are implemented correctly; all documentation is complete and up to date; and no unacceptable changes have occurred in software performance.	Approved Changes	Task Reporting from Iterated Tasks Anomaly Report(s) Required Phases Outputs of Iterated Tasks

* Outputs from phase tasks are used to develop corresponding V&V phase summary reports and are ongoing inputs to the SVVR. Outputs of V&V tasks become inputs to subsequent life-cycle V&V tasks.

Table 2 — Optional V&V Tasks and Suggested Applications

Optional V&V Tasks	Life-Cycle Phases							
	Management	Concept	Requirements	Design	Implementation	Test	Installation and Checkout	Operation and Maintenance
Algorithm Analysis			•	•	•	•		•
Audit Performance								
Configuration Control					•	•	•	•
Functional					•	•	•	•
In-Process			•	•	•	•	•	•
Physical						•	•	•
Audit Support								
Configuration Control					•	•	•	•
Functional					•	•	•	•
In-Process			•	•	•	•	•	•
Physical						•	•	•
Configuration Management	•	•	•	•	•	•	•	•
Control Flow Analysis			•	•	•			•
Database Analysis			•	•	•	•		•
Data Flow Analysis			•	•	•			•
Feasibility Study Evaluation		•						•
Installation and Checkout Testing*			•	•	•	•	•	•
Performance Monitoring								•
Qualification Testing*			•	•	•	•	•	•
Regression Analysis and Testing			•	•	•	•	•	•
Reviews Support								
Operational Readiness							•	•
Test Readiness					•	•	•	•
Simulation Analysis			•	•	•	•	•	•
Sizing and Timing Analysis				•	•	•		•
Test Certificate						•	•	•
Test Evaluation			•	•	•	•	•	•
Test Witnessing						•	•	•
User Documentation Evaluation		•	•	•	•	•	•	•
V&V Tool Plan Generation	•							•
Walkthroughs								
Design					•			•
Requirements				•				•
Source Code					•			•
Test					•	•	•	•

*Test plan, test design, test cases, test procedures, and test execution

Appendix A

(informative)

Description of Optional V&V Tasks

(This Appendix is not a part of IEEE Std 1012-1986, IEEE Standard for Software Verification and Validation Plans, but is included for information only.)

The descriptions of optional V&V tasks listed in Table 2 of this standard are defined in this Appendix. These V&V tasks are not mandatory for all V&V projects because they may apply to only selected software applications or may force the use of specific tools or techniques. These optional V&V tasks are appropriate for critical and noncritical software. By selecting V&V tasks from these optional V&V tasks, one can tailor the V&V effort to project needs and also achieve a more effective V&V effort.

algorithm analysis Ensure that the algorithms selected are correct, appropriate, and stable, and meet all accuracy, timing, and sizing requirements.

audit performance Conduct independent compliance assessment as detailed for configuration control audit, functional audit, in-process audit, or physical audit.

audit support Provide documentation for, or participate in, any audits performed on the software development (for example, configuration control, functional, in-process, physical).

configuration control audit Assess the configuration control procedures and the enforcement of these procedures.

configuration management Control, document, and authenticate the status of items needed for, or produced by, activities throughout the software life cycle.

control flow analysis Ensure that the proposed control flow is free of problems, such as design or code elements that are unreachable or incorrect.

database analysis Ensure that the database structure and access methods are compatible with the logical design.

data flow analysis Ensure that the input and output data and their formats are properly defined, and that the data flows are correct.

design walkthrough Participate in walkthroughs of the preliminary design and updates of the design to ensure technical integrity and validity.

feasibility study evaluation Evaluate any feasibility study performed during the concept phase for correctness, completeness, consistency, and accuracy. Trace back to the statement of need for the user requirements. Where appropriate, conduct an independent feasibility study as part of the V&V task.

functional audit Prior to delivery, assess how well the software satisfies the requirements specified in the Software Requirements Specifications.

in-process audit Assess consistency of the design by sampling the software development process (for example, audit source code for conformance to coding stan-

	dards and conventions and for implementation of the design documentation).
installation and checkout testing	Generate the test plan, test design, test cases, and test procedures in preparation for software installation and checkout. Place the completed software product into its operational environment, and test it for adequate performance in that environment.
operational readiness review	Examine the installed software, its installation documentation, and results of acceptance testing to determine that the software is properly installed and ready to be placed in operation.
performance monitoring	Collect information on the performance of the software under operational conditions. Determine whether system and software performance requirements are satisfied.
physical audit	Assess the internal consistency of the software, its documentation, and its readiness for delivery.
qualification testing	Generate the test plan, test design, test cases, and test procedures in preparation for qualification testing. Perform formal testing to demonstrate to the customer that the software meets its specified requirements.
regression analysis and testing	Determine the extent of V&V analysis and testing that must be repeated when changes are made to any software products previously examined.
requirements walkthrough	Ensure that the software requirements are correct, consistent, complete, unambiguous, and testable by participating in a walkthrough of the requirements specification.
review support	Provide the results of applicable V&V tasks to support any formal reviews. The results may be provided in written form or in a presentation at the formal review meeting (for example, operational readiness, test readiness).
simulation analysis	Simulate critical aspects of the software or system environment to analyze logical or performance characteristics that would not be practical to analyze manually.
sizing and timing analysis	Obtain program sizing and execution timing information to determine if the program will satisfy processor size and performance requirements allocated to software.
source code walkthrough	Ensure that the code is free from logic errors and complies with coding standards and conventions by participating in a walkthrough of the source code.
test certification	Ensure that reported test results are the actual findings of the tests. Test-related tools, media, and documentation shall be certified to ensure maintainability and repeatability of tests.
test evaluation	Confirm the technical adequacy of test plans, test design, test cases, test procedures, and test results.

test readiness review	Evaluate the code, software documentation, test procedures, test reporting, error detection, and correction procedures to determine that formal testing may begin.
test walkthrough	Ensure that the planned testing is correct and complete and that the test results are properly interpreted by participating in walkthroughs of test documentation.
test witnessing	Observe testing to confirm that the tests are conducted in accordance with approved test plans and procedures.
user documentation evaluation	Examine draft documents during the development process to ensure correctness, understandability, and completeness. Documentation may include user manuals or guides, as appropriate for the project.
V&V tool plan generation	Produce plans for the acquisition, development, training, and quality assurance activities related to tools identified for support of V&V tasks (for example, test bed software used in validation).
walkthrough	Participate in the evaluation processes in which development personnel lead others through a structured examination of a product. See specific descriptions of requirements walkthrough, design walkthrough, source code walkthrough, and test walkthrough.