

IEEE Guide for Software Verification and Validation Plans

Sponsor

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

Approved December 2, 1993

IEEE Standards Board

Abstract: Guidance in preparing Software Verification and Validation Plans (SVVPs) that comply with IEEE Std 1012-1986 are provided. IEEE Std 1012-1986 specifies the required content for an SVVP. This guide recommends approaches to Verification and Validation (V&V) planning. This guide does not present requirements beyond those stated in IEEE Std 1012-1986.

Keywords: baseline change assessment, life cycle phases, master schedule, V&V tasks

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

Copyright © 1994 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1994. Printed in the United States of America.

ISBN 1-55937-384-9

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 may involve the use of patented technology. Their approval by the Institute of Electrical and Electronics Engineers does not mean that using such technology for the purpose of conforming to such standards is authorized by the patent owner. It is the obligation of the user of such technology to obtain all necessary permissions.

Introduction

(This introduction is not a part of IEEE Std 1059-1993, IEEE Guide for Software Verification and Validation Plans.)

The purpose of this guide is to provide guidance in preparing Software Verification and Validation Plans (SVVPs) that comply with IEEE Std 1012-1986* (referred to throughout as the Standard). The Standard specifies the required content for an SVVP. This guide recommends approaches to Verification and Validation (V&V) planning. This guide does not present requirements beyond those in the Standard.

When describing specific information to be placed in sections of an SVVP, this guide quotes extensively from the Standard. Not all of the Standard is reproduced. The Standard is a prerequisite for the use of this guide.

This guide does not provide a comprehensive view of the verification and validation process itself. It does provide an overview of the process of creating an SVVP. For each section in a plan, it presents a list of topics that could be addressed when preparing that section. The guide is not specific to any application area.

This guide includes the following:

- a) Recommended planning procedures (e.g., criticality analysis) to aid in selecting specific V&V tasks for each phase
- b) General guidance on the topics that shall be addressed in the life cycle phase sections of a plan (e.g., task descriptions, methods and criteria, and inputs and outputs)
- c) Specific guidance in planning the V&V tasks required by the Standard
- d) Samples of V&V planning based on the elevator system, which is described in annex A

As with the Standard, this guide is directed at the development and maintenance of critical software—software in which a failure could affect personal or equipment safety, or could result in a large financial or social loss. The guidance may also be applied to noncritical software. Determination of criticality should be carefully considered.

The Standard and this guide employ a specific life cycle model of the software development process, although plans written in the Standard may make use of a different model.

This guide may be used where software is the system or where software is part of a larger system. When software is part of a larger system, software V&V should be integrated into a system-level V&V effort covering critical hardware, software, and interfaces to operators.

This guide is primarily directed to those preparing SVVPs. This guide may be used by project management, system and software developers, quality assurance organizations, purchasers, end users, maintainers, and those performing V&V.

Neither the Standard nor this guide assumes that the V&V effort is necessarily independent of the development effort. V&V tasks may be performed by personnel doing development tasks, by a separate team within a single organization, by a completely independent organization, or by some combination as assigned within the SVVP.

*Information on references can be found in clause 2.

At the time this guide was completed, the P1059 working group had the following membership:

Jerome W. Mersky, *Chair* **Dolores Wallace, *Co-chair***
Kirby Fortenberry, *Secretary*

François Coallier	Caroline L. Evans	Dennis E. Nickle
Cora Carmody	Robert V. Ebenau	S. J. Pasquariello
Michael S. Deutsch	Eva Freund	Lee Perkins
Michael P. Dewalt	Jack Leavenworth	Andrew Vaughan
David C. Doty	George Lee	Natalie C. Yopconka

The following persons were on the balloting committee:

H. R. Berlack	John Harauz	Dennis E. Nickle
William J. Boll, Jr.	John W. Horch	John D. Pope
Fletcher Buckley	Peter L. Hung	Patricia Rodriguez
Geoff Cozens	Myron S. Karasik	Hans Schaefer
Stewart Crawford	Judy S. Kerner	Gregory D. Schumacher
Bostjan K. Derganc	Robert Kosinski	Robert W. Shillato
Einar Dragstedt	Thomas Kurihara	D. M. Siefert
Robert G. Ebenau	Renee Lamb	A. R. Sorkowitz
Caroline L. Evans	Robert Lane	Vijaya Srivastava
John W. Fendrich	Ben Livson	David Terrell
Kirby Fortenberry	Joseph Maayan	Richard Thayer
Eva Freund	Jukka Marijarvi	George D. Tice
Roger Fujii	Roger Martin	Leonard L. Tripp
David Gelperin	Scott D. Matthews	Dolores Wallace
Yair Gershkovich	Ivano Mazza	William M. Walsh
Julio Ganzalez-Sanz	Michael McAndrew	Natalie C. Yopconka
David A. Gustafson	Sue McGrath	Janusz Zalewski
	Jerome W. Mersky	

When the IEEE Standards Board approved this standard on December 2, 1993, it had the following membership:

Wallace S. Read, *Chair* **Donald C. Loughry, *Vice Chair***
Andrew G. Salem, *Secretary*

Gilles A. Baril	Jim Isaak	Don T. Michael*
José A. Berrios de la Paz	Ben C. Johnson	Marco W. Migliaro
Clyde R. Camp	Walter J. Karplus	L. John Rankine
Donald C. Fleckenstein	Lorraine C. Kevra	Arthur K. Reilly
Jay Forster*	E. G. "Al" Kiener	Ronald H. Reimer
David F. Franklin	Ivor N. Knight	Gary S. Robinson
Ramiro Garcia	Joseph L. Koepfinger*	Leonard L. Tripp
Donald N. Heirman	D. N. "Jim" Logothetis	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 I. Warshaw

Rachel A. Meisel
IEEE Standards Project Editor

Contents

CLAUSE	PAGE
1. Overview.....	1
1.1 Scope.....	1
2. References.....	2
3. Conventions, definitions, and acronyms and abbreviations.....	3
3.1 Conventions	3
3.2 Definitions.....	3
3.3 Acronyms and abbreviations.....	3
4. Software verification and validation.....	4
4.1 Software V&V planning guidance.....	5
4.2 Integrating and continuing V&V tasks	8
5. SVVP guidance.....	16
5.1 Purpose.....	16
5.2 Referenced documents.....	17
5.3 Definitions.....	17
5.4 Verification and validation overview.....	18
5.5 Life cycle verification and validation	20
5.6 Reporting.....	43
5.7 Verification and validation administrative procedures	44
Annex A (informative) Extracts from a sample Software Verification and Validation Plan.....	53
A.1 Overview.....	53
A.2 Elevator system reference material.....	54
A.3 Extracts from a sample SVVP	59
Annex B (informative) Critical software and criticality analysis	79
Annex C (informative) The seven planning topics	81
C.1 V&V Tasks	81
C.2 Methods and criteria	82
C.3 Inputs and outputs.....	83
C.4 Schedule.....	84
C.5 Resources	85
C.6 Risks and assumptions.....	86
C.7 Roles and responsibilities	87

IEEE Guide for Software Verification and Validation Plans

1. Overview

This guide is the explication of IEEE Std 1012-1986¹ (referred to throughout as the Standard). This Standard specified the required content of a Software Verification and Validation Plan (SVVP). This guide provides the guidance and background discussion that a user of the Standard may need in order to write a plan.

This guide is divided into four clauses. Clause 1 provides the overview and scope of this guide. Clause 2 lists references to other standards. Clause 3 defines the terms, conventions, and acronyms used in this guide. Clause 4 provides an overview of software V&V, general planning guidance, and a description of the major V&V activities that continue across more than one life cycle phase (e.g., traceability analysis, evaluation, interface analysis, and testing). Clause 5 provides a sequential review of each of the sections of the SVVP, providing specific guidance for each section. In many cases, the guidance for a section or task is in the form of a list of questions or activities to consider. Note that these lists are not complete because for any given situation there will be other questions or activities that can be addressed. Also, all items in a given list may not be applicable to any given situation.

This guide also contains three annexes. Annex A describes the elevator example that was used to develop sample plan extracts. These sample extracts are also presented in this annex. Annex B describes and overviews software criticality planning analysis. Annex C provides guidance for the seven planning topics that are aimed at V&V tasks in an SVVP.

1.1 Scope

This document provides specific guidance about planning and documenting the tasks required by the Standard so that the user may write an effective plan. Some guidance is in the form of checklists, questions, or activities to consider.

This guide does not present requirements beyond those found in the Standard. Mention of certain methods or activities in this guide does not mean that the user of the Standard must include these in a plan in order for that plan to comply with the Standard. There may be other methods or activities that are relevant to a specific software project or product. The user of this guide is encouraged to survey the standards referenced in clause 2 for additional guidance, particularly guidance for specific tools or techniques.

¹Information on references can be found in clause 2.

2. References

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

IEEE Std 610.12-1990, IEEE Glossary of Software Engineering Terminology (ANSI).²

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 830-1993, IEEE Recommended Practice for Software Requirements Specifications.

IEEE Std 1008-1987, IEEE Standard for Software Unit Testing (ANSI).

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

IEEE Std 1016-1987, IEEE Recommended Practice for Software Design Descriptions (ANSI).

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

²IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

3. Conventions, definitions, and acronyms and abbreviations

3.1 Conventions

Below are a number of conventions adopted within the body of this guide.

- *Standard* refers to IEEE Std 1012-1986.
- *Plan* refers to a Software Verification and Validation Plan (SVVP).
- *Shall* or the imperative form indicates an item or activity required by the Standard.
- *Should* indicates a recommended item or activity.
- *May* indicates an item or activity appropriate under some, but not all, conditions; for which there are a number of acceptable alternatives; or for which there is no professional consensus.
- *Life cycle model* is taken from the Standard. As stated in the Standard, a different model of the life cycle may be used, provided the SVVP written for that model includes cross-references to the original model.
- *System* is used to denote the item that will contain the developed software.
- Material quoted from the Standard is indented and italicized and is followed by a reference to the section (or, in some cases, part of a table) of the Standard.

3.2 Definitions

This guide is consistent with other IEEE standards in its use of terms. For a complete list of definitions, refer to IEEE Std 1012-1986.

Note that definitions for the terms *verification* and *validation* may be used differently by other standards organization.

3.3 Acronyms and abbreviations

The following acronyms and abbreviations are used in this guide:

COTS	Commercial-off-the-shelf
CPM	Critical path method
IV&V	Independent verification and validation
PERT	Program evaluation review technique
QA	Quality assurance
SDD	Software design description
SQA	Software quality assurance
SRS	Software requirements specification
SV&V	Software verification and validation
SVVP	Software verification and validation plan
SVVR	Software verification and validation report
TBD	To be determined
V&V	Verification and validation

4. Software verification and validation

Software Verification and Validation (V&V) is a disciplined approach to assessing software products throughout the product life cycle. A V&V effort strives to ensure that quality is built into the software and that the software satisfies user requirements. V&V provides software management with insights into the state of the software project and products, allowing for timely change in the products or in the development and support processes.

Software verification and validation employs review, analysis, and testing techniques to determine whether a software system and its intermediate products comply with requirements. These requirements include both functional capabilities and quality attributes.

These quality attributes will differ from project to project. Quality attributes that are identified serve the user's need for a software product that is capable of meeting its objectives with adequate performance with no unexpected side effects. The following list enumerates quality attributes that may be specified.

- accuracy
- completeness
- consistency
- correctness
- efficiency
- expandability
- flexibility
- interoperability
- maintainability
- manageability
- portability
- readability
- reusability
- reliability
- safety
- security
- survivability
- testability
- usability

The objectives of the V&V effort are to find defects and to determine if required functions and attributes are built into the software system. The activities of V&V operate on the products of software development and support the following:

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

A V&V effort is typically applied in parallel with software development and support activities. Some V&V tasks may be interleaved with the development and support process. A V&V effort consists of management tasks (e.g., planning, organizing, and monitoring the V&V effort) and technical tasks (e.g., analyzing, evaluating, reviewing, and testing the software development processes and products) to provide information about the engineering, quality, and status of software products throughout the life cycle.

Planning for a V&V effort begins early in the project so as to aid in assessing the scope of the V&V effort as part of the total effort. This is necessary in order to ensure that V&V resource needs are included in the total project planning. The initial SVVP may provide insights to the financial sponsor of the project concerning development or support plans. It also provides enough information for the sponsor to approve the plan and to monitor its implementation.

It is important on each project to clarify how V&V fits into the overall project life cycle and relates to all project entities (e.g., user, developer, buyer, software configuration management, software quality assurance, etc.). Sometimes the V&V effort will be performed by an entirely different organization, often directly for the buyer of the software. This situation is referred to as Independent Verification and Validation (IV&V). V&V tasks may be performed internally by a completely separate organization within the company or perhaps by the systems engineering group or product assurance group. Selection of some tasks may depend on how a project is organized (e.g., consider not only how the V&V tasks serve immediate V&V objectives, but also how they serve the total project).

The first V&V tasks begin early, with concept documentation evaluation, requirements analysis, and acceptance test planning. Planning the V&V should be closely coupled to the planning of the rest of the project. The planner should anticipate and schedule regular updating of the SVVP to reflect changes in the overall development effort.

Software verification and validation tasks support one another and combine to become a powerful tool that does the following:

- Discovers errors as early as feasible in the software life cycle
- Ensures that required software qualities are planned and built into the system
- Predicts how well the interim and final products will result in final products satisfying user requirements
- Assures conformance to standards
- Confirms safety and security functions
- Helps prevent last-minute problems at delivery
- Provides a higher confidence level in the reliability of the software
- Provides management with better decision criteria
- Reduces frequency of operational changes
- Assesses impact of proposed changes on V&V activities

4.1 Software V&V planning guidance

This subclause describes activities required for effective V&V planning. The purpose of planning and plan documentation is to employ the V&V resources efficiently, to monitor and control the V&V process, and to allow the identification of each participant's role and responsibility. Planning results in a well-structured, thorough, and realistic SVVP that provides the foundation for a successful V&V effort. Documentation of the scope and explicit statement of the objectives provides all parties with an understanding of what is going to be achieved and what is not.

Each project needs its own SVVP, based on the generic plan identified in the Standard. The specific plan will be tailored to the project. The plan may be modified at times in response to major changes. While the basic objectives of the V&V effort are not likely to change over time, there may be major changes that require response. These changes may be associated with development activities (e.g., significant delays in schedules, unanticipated technological changes) or V&V tasks (e.g., loss of resources, inability to develop tools in time). Therefore, procedures for modifying the plan should be included in the plan.

V&V planning, which should be thought of as an integrated part of overall project planning, may be broken down into the following steps:

- a) *Identify the V&V scope.* Define a set of software tasks as early as possible. The tasks should be tailored and take into consideration several software factors (e.g., criticality, complexity, available resources including development environment and tools). Determining which V&V tasks to perform requires a decision-making methodology. One such methodology, criticality analysis, is presented in annex B.

Before beginning the next step, the recommended V&V tasks should be reviewed by all participants, especially the user and customer. The next step should begin only after agreement is achieved.

- b) *Establish specific objectives from the general project scope.* Detailed, measurable, and achievable objectives establish the conditions of satisfaction with the V&V effort. These conditions of satisfaction provide the basis for assessing and enforcing V&V performance. Measurable criteria, such as defined load limits and response time, are desirable because of the objectivity they afford. Also, the discipline and effort required to formulate measurable objectives reveal ambiguities and differences of expectations early in the V&V effort.

It is important to identify only achievable objectives. The presence of unachievable objectives reduces the credibility of a plan, and of the V&V effort itself, before the execution of the plan begins. A realistic plan enhances the motivation of all parties in a project and increases the value of the V&V effort.

- c) *Analyze the project input prior to selecting the V&V tools and techniques and preparing the plan.* Input analysis identifies all the specific products, information, and other documentation that will be available during the project phases. Examples of such products are given in figure 1 of the Standard and are reproduced here in figure 1. Input can be divided into two types—the products and information available at project start and the products and information that will become available in the later phases of the development cycle. The items that are available during the planning stage are either project- or product-related. Project-related information includes the budget, manpower and other resources, development schedules and milestones, and any other project-specific considerations, such as contractual requirements. Product-related information includes concept documentation, requirements specifications, or other formal specifications for the product being developed. Note that terminology for these products varies according to the user or developer environment.

Historical defect data from previous versions of the same or similar software, if available, can be a great aid. Such historical data will be available only if planned for (e.g., by archiving and analyzing anomalies). When planning the V&V effort, prediction and analysis of the nature and number of expected defects aid in the selection of V&V tasks and help to determine the resources required to perform the tasks.

- d) *Select techniques and tools.* Identify those tools and techniques (such as those that support traceability analysis) that are available and applicable to the project, and gather sufficient information in order to choose among alternatives. For techniques, this information includes input needed, range of applicability, and resources and skills needed. For tools, this information includes availability of tool and associated documentation, applicability, and resource requirements including cost of operation and training.

Specific information about the software development environment, methodology, and tools should be considered when selecting the tools and techniques to be used in V&V. Techniques and tools need to be selected with the format, structure, and schedule of the development products in mind. Also, while in some cases it is appropriate to have separately developed and qualified tools, it may be more cost effective to use a common set of tools for similar functions.

Obtain a subset of tools and techniques that are tailored to the particular constraints and needs of the project. When doing so consider the skills and training of the personnel and the availability of the other necessary resources.

- e) *Develop the plan.* Review the results of the previous steps and prepare a detailed set of tasks (addressing the seven topics discussed in 3.5 of the Standard and annex C of this guide) to meet the V&V goals, objectives, and constraints.

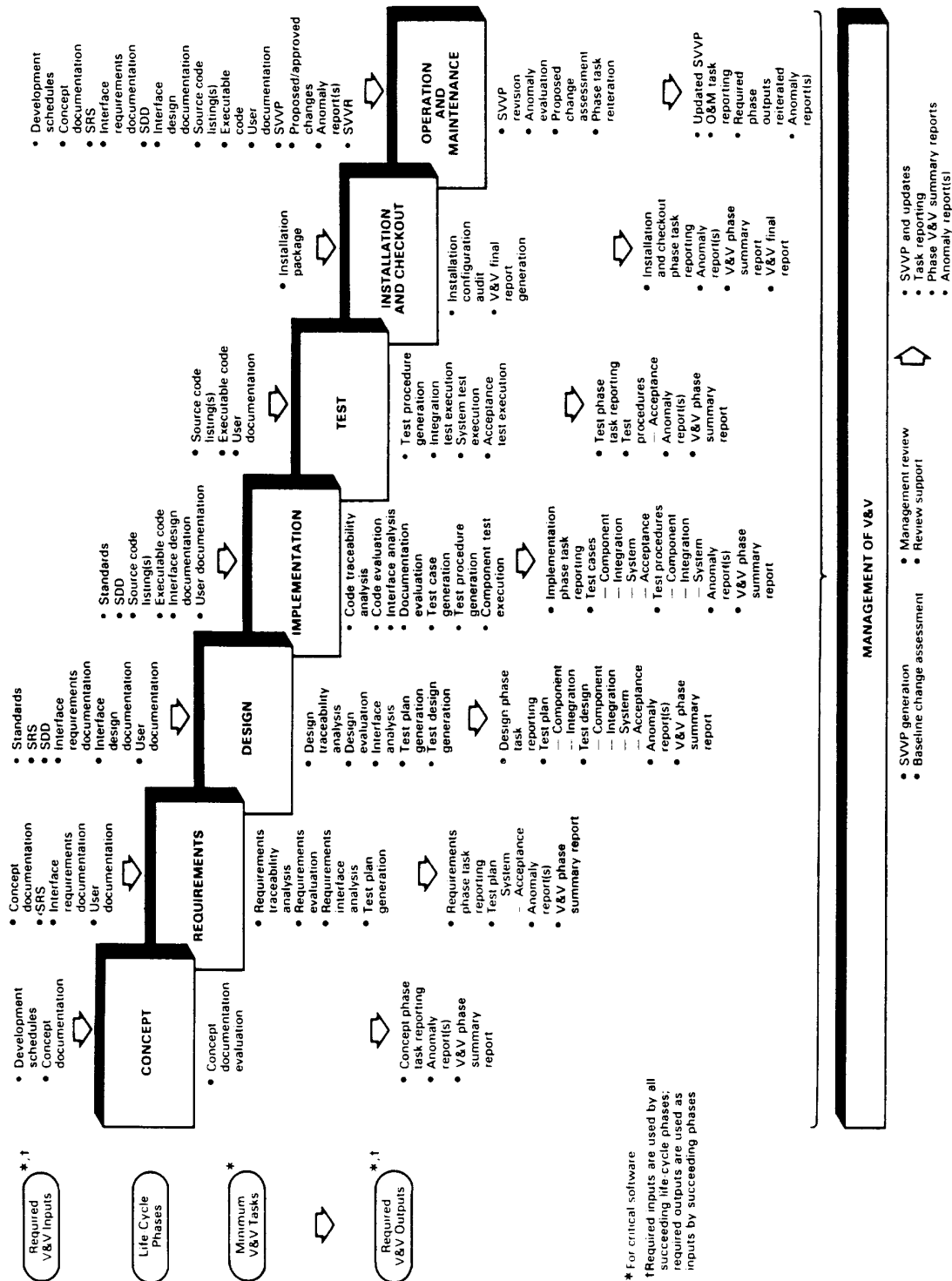


Figure 1—Software verification and validation plan overview

4.2 Integrating and continuing V&V tasks

Several tasks, as described by the Standard, collectively make up continuing activities that go across the different life cycle phases. These general activities are traceability analysis, evaluation, interface analysis, and testing. Two other activities also discussed in the Standard are management and reporting. Management is described in 5.5.1 of this guide. Reporting is described in 5.6.

These activities are horizontal threads that tie together the subsequent phase activities and allow verification to be more effectively conducted. The V&V planner should plan to integrate and share resources throughout the life cycle so that planning and implementation of all tasks of a continuing activity are more effectively performed. See table 1.

In order to avoid redundancy, only the general characteristics of these activities are described in this sub-clause. Phase-specific applications are described in 5.5. Each activity is described in terms of the functions performed, the kinds of problems it will uncover, the significance of the problems, and the relationship of this activity to other activities.

Table 1—Relationships between tasks

Contribution of Task	Task			
	<i>Traceability Analysis</i>	<i>Evaluation</i>	<i>Interface Analysis</i>	<i>Testing</i>
Criticality Analysis	Identifies critical features for tracing.	Identifies critical areas for evaluation.	Identifies critical interfaces.	Establishes priorities for testing.
Traceability Analysis		Maps the location of each feature to be evaluated.	Identifies where interfaces are required between dependent components, and the information to be passed.	Traces test documents. Assists in developing a test strategy by determining how functions are distributed.
Evaluation	Assures the structural validity of the distribution of function.		Establishes that interfaces are appropriate and are correctly implemented.	Identifies attributes that require dynamic validation, and the extent of testing that will be needed.
Interface Analysis	Establishes the need for traceability, and assures that dependent elements can communicate.	Identifies explicit relationships between elements, and their input and output requirements.		Establishes the framework for integration test planning.
Testing	Confirms the traceability and the distribution of functions.	Confirms the evaluation.	Confirms that interfaces are operational.	

4.2.1 Traceability analysis

The Standard requires traceability analysis in the requirements, design, and implementation phases. Traceability is the ability to identify the relationships between originating requirements and their resulting system features. It permits tracking forward or backward through the network of interrelationships that are created as requirements are decomposed and refined throughout a system's life cycle. Traceability provides the

thread that links one element to another. When an element is traced to another, and that element is traced on to another, a chain of cause and effect is formed.

Traceability allows verification of the properties set forth in the concept and that requirement specifications have been

- a) Carried forward to the design specifications
- b) Implemented in the code
- c) Included in the test plan and cases
- d) Provided to the customer and user in the resulting system

Traceability, with the aid of forward and backward tracing, facilitates the construction of efficient test plans and permits verification that the resulting test cases have covered the permutations of functional and design requirements/features.

When the trace from all software requirements back to the concept documentation and system requirements is verified, then successive traceability analysis occurs, beginning from the software requirements through all the development representations, user documentation, and test documentation. Each trace is analyzed for consistency, completeness, and correctness to verify that all software requirements are implemented in the software and are associated with the correct design, code, and test information.

In the course of performing traceability analysis, some errors are readily apparent, such as a requirement with no design element, or a design element with no source code, or the converse of these. The analyst examines each trace path to ensure that the connected pieces are the proper ones. To be able to do this the analyst has to understand the intent of both requirements and design.

Traceability analysis can be used to support configuration management, test coverage analysis, analysis of V&V results, regression testing, criticality assessment, and V&V management decisions. Also, traceability analysis is useful in evaluating the software development effort for good software engineering practices.

Traceability analysis is performed to assure that

- Every requirement in the specification is identified correctly
- All traces are continuous from the prior phase through the current phase
- Forward and backward traces between adjoining phases are consistent
- The combined forward traces originating from each specification, fully support that specification
- Each current specification or feature is fully supported by traceable predecessor specifications

4.2.2 Evaluation

Evaluations ascertain the value or worth of an item and help to assure that a system meets its specifications. Evaluations are performed by many persons across all life cycle phases, on both interim and final software products, and may be either a comprehensive or selective assessment of a system.

Evaluations uncover problems in the different products and their relationships. These problems relate to the basic user need for the system to be fit for use in its intended setting. For a product to be fit for use, an evaluation may be used to assure that

- The product conforms to its specifications.
- The product is correct.
- The product is complete, clear, and consistent.
- Appropriate alternatives have been considered.
- The product complies with all appropriate standards.
- The product meets all specified quality attributes.

Evaluations not only identify problems, but may help to determine the progress of software development by recommending that the project

- Continue on to the next stage
- Correct specific items first, then continue
- Go back several steps and correct some problems
- Perform additional evaluations, such as a simulation, before that part of the system is further refined
- Monitor the progress of an item whose quality may be doubtful
- Make changes in methodology or software tools
- Make changes in schedules
- Make staffing decisions, such as additional resources, additional training, or changes in assignment

Evaluations are used through all phases and for all types of software products, including user documents, manuals, and other project documents. These may be of many forms, such as text or graphic representations, and in various media, such as paper, magnetic tape, diskette, and computer files. This range of product types and forms requires a large variety of techniques for performing and managing software evaluations.

When evaluating system products, there will be a change in emphasis from one development stage to the next. As the product progresses from a concept to a final product, its functional requirements are first defined. Next, the organization and structure of design elements are established. Then, all executable procedures are prepared. Questions of “what and why,” “what-if,” “alternatives,” and “better-or-best” will predominate during the earlier development stages, while issues of “how,” “correctness,” and “conformance” will be more frequent in the later development stages.

When selecting evaluation strategies, the V&V planner first uses the features and characteristics of the system as a guide in the selection of the evaluation techniques. The planner shall question the system’s functions, how they will be addressed, and what types of problems might be encountered. Second, the V&V planner derives additional evaluation techniques from quality attributes that may be specified for the system. Third, the V&V planner considers other evaluation methods, based on special V&V concerns and experience.

It may be helpful for the V&V planner to consider these three approaches when selecting techniques for evaluating the product:

- a) Static analysis of the specification (e.g., reviews, inspections, structure analysis)
- b) Dynamic analysis of expected software behavior (e.g., simulation, prototyping, screen emulation, branch execution analysis)
- c) Formal analysis (e.g., mathematical proofs)

All these evaluation types can be used in conjunction with one another to provide a powerful synergism. Selection of a particular type of evaluation depends on the results required, tool availability, and cost trade-offs.

Static analysis detects deficiencies and errors through the examination of the software product, manually or automatically, without executing the software code. For example, static analysis can be applied to evaluate the form and content of the specification. It is usually straightforward. It frequently employs various types of reviews. Applicable to all levels of specification, it is able to detect flaws that could preclude or make meaningless the other forms of analysis, or prevent inadequate testing of the product. Static analysis focuses on the form, structure, and content of the product; its interfaces and dependencies; and on the environment in which it is to be used. Static analysis may be applied at all phases of development, and to all documented deliverables.

Dynamic analysis requires the execution of the software itself, or a model or simplified version of the software, to determine the validity of some of its attributes. Dynamic analysis can produce results that are not

available or are more time consuming with the other techniques. Dynamic analysis may also be more tractable than formal analysis, as there are a variety of tools expressly designed to support it.

Formal analysis uses rigorous mathematical proof techniques to analyze the algorithms or properties of the software. It can provide a strong conclusion regarding certain properties of an approach, but it is limited by the difficulty of its application and the general scarcity of automated support. Formal techniques generally require both a formal specification and a formal implementation. For example, the correctness of a natural, English-language requirement could not be proven. Certain assertions could be drawn from the requirement though, and proved to satisfy a set of conditions, thereby increasing the confidence that could be placed in the requirements for those conditions. Formal analysis is frequently used to verify sections of a specification that handle security requirements.

4.2.3 Interface analysis

When information is passed across a boundary (e.g., hardware to software, software to software, software to user) there is always the possibility of losing some information or altering the information content. The task of interface analysis serves to ensure the completeness, accuracy, and consistency of these interfaces. Interface requirements at the design and implementation phases should be identified and analyzed at the functional, physical, and data interface level. The goal of interface analysis is to evaluate the specific software deliverable (e.g., requirements, design, code) for correct, consistent, complete, and accurate interpretation of the interface requirements.

Interface analysis is concerned with data that flows from one part of the system to another. The sources of this information are important in assuring that the intended flow is feasible. Information about the recipient of the interface data is needed to assure that each interface is indeed necessary and sufficient. The correctness of the interface will rely on the use to which it is put, and its form and content will be part of the analysis.

Interface analysis should focus on three interface areas:

- a) *User interface.* This should analyze what the human interfaces to the software product are, such as required screen format, protection mechanisms, page layout and content of the reports, relative timing of inputs and outputs, etc.
- b) *Hardware interface.* This should analyze what the logical characteristics of each interface are between the software product and the hardware components of the system. Identify electronic devices, firmware, communication devices, and output devices. Then identify applicable standards for these interfaces and verify the current application's interface.
- c) *Software interface.* This should analyze what the interfaces to other required software products are (e.g., data management system, operating system, or a library package), and interfaces with other application systems. Identify applicable standards for any other software products and interfaces to other application systems. Verify correct software interfaces to them.

In planning interface analysis consider the following:

- *Are the interface objectives technically adequate and well understood?*
- *Are all data elements well defined?*
- *Are all restrictions and constraints clearly defined?*
- *Were all the different kinds of interfaces taken into account and properly described?*
- *Are all hardware-to-software functional interfaces specified in quantitative terms such as units, bits per second, message formats, priority rules, word length, timelines, and protocols?*
- *Are all software-to-software interfaces functional, and are all data interface levels specified in quantitative terms, such as data timeliness, data definition, data formats, priority rules, message content, and communication protocols?*
- *Are the interface performance requirements well defined, and are the limits specified?*

- *Is the criticality of the interface taken into consideration?*
- *What are impacts if the interface is degraded?*
- *Are the interfaces testable and maintainable?*
- *Have all appropriate standards been identified for the interfaces, including those from the current software application to its environment?*

4.2.4 Testing

Testing is the process of analyzing a software item to detect the differences between existing and required conditions (that is, bugs) and to evaluate the features of the software item.

In the context of software verification and validation, testing can be defined as the testing that is performed in support of the V&V objectives. These objectives may differ from those of the developer. For example, V&V testing may focus on a narrow or critical area or on ensuring compliance to planned testing by the developers.

The scope and organizational responsibilities for the testing defined in an SVVP will depend on project-specific considerations. Once V&V objectives are established, the testing performed is in support of those objectives. Recognize there may be other testing activities beyond those of software verification and validation and there may be other organizations involved. Testing performed as part of V&V may be performed by personnel who generated the software or by personnel independent of the original developers.

The remainder of this subclause provides a description of testing activities performed during software verification and validation. It is not meant to be a guide to testing, but to clarify the concepts and terminology of the Standard, and to be a basis for V&V test planning.

4.2.4.1 Levels of testing

Testing is performed at several points in the life cycle as the product is constructed component by component into a functioning system. These points in the life cycle can be viewed as levels of capabilities that need to be tested. The levels progress from the smallest or single component through combining or integrating the single units into larger components or stages. The Standard describes these levels as component, integration, system, and acceptance. Table 2 further describes these levels of testing.

4.2.4.2 Test planning

V&V test planning is a continuing activity performed throughout the life cycle, starting from the requirements phase up to the test phase. Test plans are developed for each level of testing. These plans correspond to the developed products of the phase. The requirements phase produces the system requirements specification and acceptance and system test plans. The design phase produces the system design document and the component and integration test plans. The implementation phase produces the software product and test cases, test procedures, and component level execution. In the test phase, acceptance test procedures are produced and integration system and acceptance tests executed. An additional benefit of test planning is the additional review of the requirements, design, and interface descriptions.

The Standard provides for the creation of the test plan components in stages at appropriate points and times in the life cycle. Table 3 shows the stages and sequences. Following the table is an explanation of the various test activities.

Table 2—Levels of testing

Testing	Definition in the Standard	Purpose	Traceability
Component Testing	Testing conducted to verify the implementation of the design for one software element (e.g., unit, module) or a collection of software elements.	Ensures program logic is complete and correct. Ensures component works as designed.	From each test to detailed design
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.	Ensures that design objectives are met.	From each test to the high-level design
System Testing	The process of testing an integrated hardware and software system to verify that the system meets its specified requirements.	Ensures that the software as a complete entity complies with its operational requirements.	From each test to the requirements
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.	Ensures that customers' requirements objectives are met and that all components are correctly included in a customer package.	From each test to customer-specific requirements

Table 3—Test activities in life cycle phases

Activities	Requirements	Design	Implementation	Test
Test Plan Generation	- System - Acceptance	- Component - Integration		
Test Design Generation		- Component - Integration - System - Acceptance		
Test Case Generation			- Component - Integration - System - Acceptance	
Test Procedure Generation			- Component - Integration - System	- Acceptance
Test Execution			- Component	- Integration - System - Acceptance

4.2.4.2.1 Test plans

A test plan spells out the scope, approach, resources, and schedule of the testing activity. It indicates what is to be tested, what is not to be tested, testing tasks to perform, people responsible, and risks. Key points in planning include

- Transitioning from one phase or level to another
- Estimating the number of test cases and their duration
- Defining the test completion criteria
- Identifying areas of risks
- Allocating resources

The testing tasks performed during the requirements phase involve planning for and generating the system and acceptance test plans. Since these tests attempt to demonstrate that both the operational concept and the requirements are satisfied, planning may begin as soon as the requirements are written, although completion of the test plans shall await finishing other requirements phase V&V tasks. More detailed levels of preparation of test procedures and test cases will be performed in later phases.

The system design document is a key input to planning for component and integration test in the design phase. By developing the test plans early, there is adequate time to put test resources in place and to ensure that any untestable design features are identified prior to implementation.

Identifying methodologies, techniques, and tools is part of planning for testing. Methodologies are derived from the overall testing strategy. Techniques and tools are identified according to the specified methodologies. The specific methodologies, techniques, and tools selected will be determined by the type of software (e.g., application, operating system) to be tested, the test budget, the risk assessment, the skill level of available staff, and the time available.

In planning for verification and validation testing, the V&V planner should answer the following questions:

- *Who is responsible for generating the different levels of test designs, cases, and procedures?*
- *Who is responsible for executing the different levels of tests?*
- *Who is responsible for building and maintaining the test bed?*
- *Who is responsible for configuration management?*
- *What are the criteria for stopping test effort?*
- *What are the criteria for restarting test effort?*
- *When will source code be placed under change control?*
- *Which test designs, cases, and procedures will be placed under change control?*
- *For which level of tests will anomaly reports be written?*

It is often difficult to determine when to stop testing or when a reasonable number of defects have been detected. In fact, it may be unreasonable to expect that all defects will be detected. Therefore, completion criteria should be provided as a guideline for test completion. There are at least two common types of completion criteria. One criterion, test case completion, requires that all resultant test cases are executed without observing defects. A second criterion is fault prediction. For example, since the objective of testing is to find defects, the second criterion might be to test until a predefined number of defects have been found (or, conversely, a number of successful tests has been executed).

4.2.4.2.2 Test design

The test plan is followed by the development of test designs in the design phase. Test designs refine the test plan's approach, identify specific features to be tested by the design, and define its associated test cases and procedures. Whenever possible, tests should be designed to be suitable for regression testing; that is, tests

previously executed and verified can then be repeated at a later point in development or during maintenance of the installed system.

Consider the following when developing test designs:

- Required features to be tested
- Specified quality attributes
- Load limits
- Stress tests
- Configurations that the software may need to support
- Compatibility with existent or planned system components
- Security of the system
- Storage limits of the software, and related data
- Performance (such as accuracy) response times, and throughput
- Installability, primarily in the user environment
- Reliability/availability to the specifications
- Recovery from software and data failures
- Serviceability requirements
- Users' guides
- Human factors for usability and acceptability
- Interfaces with other system components
- Hardware interfaces

4.2.4.2.3 Test case

The test cases and test procedures are developed in the implementation phase. A test case specifies actual input values and expected results. The goal in the generation of test cases is to exercise the component's logic and to set up testing scenarios that will expose errors, omissions, and unexpected results. In developing test cases, one desire is to produce the smallest set of cases possible that will still meet the goal. Use of a matrix relating requirements to test cases aids in determining completeness and overlap.

4.2.4.2.4 Test procedure

A test procedure identifies all steps required to operate the system and exercise the specified test cases in order to implement the associated test design. Refer to IEEE Std 829-1983 for a discussion of test procedures.

4.2.4.2.5 Test execution

Test execution is the exercising of the test procedures. Test execution begins at the component level of testing in the implementation phase. The remaining levels of testing (e.g., integration, system, acceptance) are executed in the test phase.

5. SVVP guidance

The following provides detailed information about each section of an SVVP. The information here is organized in the same order as an SVVP and written in accordance with the Standard. In most cases, the following subclauses begin by quoting the requirements of the Standard for that section. Immediately following is the section number or subsection number where this quote can be found in the Standard.

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. [3]

When preparing an SVVP, include the following additional information prior to the first section of the plan, when appropriate to the particular project.

- a) *Cover or title page.* This will typically include the title of the plan, a document or configuration control identifier, the date, and the identification of the organizations by whom and for whom the SVVP is prepared.
- b) *Signature lines or page.* In many cases, the SVVP will require the signature or signatures of the persons responsible for writing or approving the plan.
- c) *Revision history.* If the SVVP is updated by issuing change pages, it is helpful to include a list of such pages at the front of the SVVP. A brief history of the document and change bars or some other form of attention to changed material are useful regardless of the distribution method.
- d) *Preface or foreword.* This will typically include a brief description of the project and identification of related documents (e.g., project management plan, software quality assurance plan, configuration management plan).
- e) *Table of contents.* A table of contents is required if the reader is to make efficient use of the plan.
- f) *Lists of figures, tables, or illustrations.* If there are more than a few figures, tables, or illustrations, or if these are referenced at widely-scattered points in the text, reference lists similar to a table of contents should be included. These items are typically listed separately.
- g) *Assumptions.* Indicate any assumptions on which the plan is predicated, such as anticipated deliveries, availability of input, or intermediate products.

Include the following material after the last section of the SVVP, when appropriate to the project:

- *Glossary.* Define project-specific terms and acronyms.
- *Procedures.* Provide a clear understanding of the SVVP. In many cases, it is useful to separate the high-level planning information from the detailed procedures. These procedures may also be provided in a separate document referenced in the body of the SVVP.
- *Report formats and standard forms.* Furnish samples of reporting forms that will be useful. If large numbers of standard forms and report formats are to be included, these are best placed in an appendix.

5.1 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. [3.1]

See annex A, A.3.1.

The purpose statement of the SVVP provides the highest level description of the software V&V effort. The following topics should be addressed in the purpose statement:

- Identify the project to which the SVVP applies and describe why the plan is written.
- State the goals to be satisfied by the SVVP. For example, a specific V&V effort may intend to validate that all security requirements are satisfied. Another plan may be concerned only with validating performance requirements.
- Clearly summarize the V&V effort and the responsibilities conveyed under this plan.
- Define the extent of the application of the SVVP to the software. Explicitly cite each part of the software to which the SVVP applies and each part not covered by the SVVP. For example, perhaps only two of three subsystems of a computer system will fall under the SVVP (the third component may be unchanged from a previous system), or the SVVP may be invoked only for design verification.
- Identify waivers and changes if the Standard's requirements have been tailored for this SVVP. Examples include items such as addition or removal of sections or use of a different life cycle model. Sufficient detail should be included to demonstrate compliance with the Standard if the SVVP is audited. Descriptions of extensive changes may be placed in an appendix or a separate tailoring document.

5.2 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. [3.2]

This section of the SVVP should

- Specify documents completely, using any serial or publication numbers available.
- Make certain the correct version of the document is cited.
- Note clearly changes to referenced document list when SVVP is revised.
- List all documents referenced in the plan.
- Be consistent with other referenced document lists in other project documentation or explicitly indicate differences and reasons.

5.3 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. [3.3]

For this section of the SVVP

- Include only terms necessary for understanding the SVVP.
- Keep the list short by referring to lists of definitions in other project documentation.
- If separate lists are maintained in different project documentation, be sure to use the same definitions in each list. It is helpful, to this end, to have a single database of project definitions and acronyms.
- Use definitions from IEEE Std 610.12-1990 and the series of IEEE glossaries where possible.

5.4 Verification and validation overview

(Section 4 of the Plan.) This section shall describe the organization, schedule, resources, responsibilities, tools, techniques, and methodologies necessary to perform the software verification and validation. [3.4]

5.4.1 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. [3.4.1]

See annex A, figure A.3.

The specific organizational structure of the V&V effort will depend on the nature of the system under development, the developing and acquiring organizations, and the contractual arrangements. Each organization responsible for a V&V effort may want to arrange the V&V effort differently. A V&V effort is simply the set of V&V tasks defined by the SVVP to be performed to achieve a quality product. Organization of the effort is then the assignment of each task to some appropriate person or organizational entity. There is no single, correct organization for all projects.

When planning for the organization of a V&V effort, consider the following:

- Assigning specific responsibility for each task (e.g., accepting the input, performing the task, analyzing the results, reporting the results, making decisions based on the results) where responsibility may be shared
- In the case of overlapping responsibilities, being precise about these assignments
- Using diagrams to show the control and data flow of V&V efforts to clarify responsibilities

5.4.2 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. [3.4.2]

The master schedule summarizes the various V&V tasks and their relationships within the overall project environment. The objective is to spell out the orderly flow of materials between V&V activities and project tasks. This helps to ensure that V&V tasks are appropriately placed and their deliverables are identified within the larger project environment. Be aware that the development of the master schedule will be an iterative process.

In developing the master schedule, the V&V planner should focus on V&V tasks and their placement within the project schedule and highlight the key V&V tasks, deliverables, and completion dates. If an independent V&V effort is being conducted, the interfaces for delivery of materials, reviews, completion meetings, etc. need to be highlighted. There are many formats for schedule presentation (e.g., Gantt Charts, PERT, CPM) and in some cases analysis of schedule flow. The approach used should be consistent with other project elements.

5.4.3 Resource 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. [3.4.3]

An overview of the resources needed for the V&V effort should be presented. The planner should not repeat the resource requirements of the individual tasks; instead, overall requirements should be summarized and potential conflicts for resources, long-lead items, inefficient uses of resources, and alternative resource options should be identified. For very small plans, all resource information could be placed in this section, rather than in the task discussions. It may be valuable to collect the detailed resource information in an appendix.

Resource types include labor or staffing, facilities, equipment, laboratories, the configuration of the laboratories, tools (both software and hardware), budget and financial requirements, documentation, special procedures and conditions (such as, security, access rights, and/or controls).

- Use graphs and tables as an effective means of presenting resource use. Graphs of resource use versus calendar time or life cycle phase give a quick grasp of the relative amounts of effort involved in the different tasks or phases. Tables of resource use, again by calendar time or life cycle phase, aid in comparing resource use with a budget or with the requirements of other project efforts.
- Include in the equipment and laboratories summary the type of equipment needed, duration needed, particular configurations, and other peripheral facilities that will be needed to perform the total V&V operations.
- In the tools section of the summary list the various tools that are to be used throughout the V&V effort. The tools can be subdivided into software and hardware.
- In the budget and financial requirements, take all the resources into account and allow for additional tools and staff to cope with contingencies.

5.4.4 Responsibilities

(Section 4.4 of the Plan.) This section shall identify the organizational element(s) responsible for performing each V&V task. 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. [3.4.4]

See annex A, table A.1.

There are two levels of responsibility for the V&V tasks—general responsibilities assigned to different organizational elements throughout the project and specific responsibilities for the tasks to be performed. A summary of the general responsibilities may be described in this section of the SVVP or in another project-level plan (e.g., a project management plan). If described in another document, this section should contain a summary and a reference to the other document. The specific responsibilities may be described in this section, or this section may summarize the responsibilities defined in the life cycle phase sections of the SVVP.

5.4.5 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. [3.4.5]

Describe the V&V approach, tools, and techniques and their roles in the V&V effort. This may be in narrative or graphic form. References to technique or tool descriptions should be included. A separate tool plan

may be developed for a software tool acquisition, development, or modification. In this case this section of the SVVP should refer to the tool plan. If a tool is to be acquired or developed, its acquisition or development schedule should be included in the V&V schedule. Determine whether sufficient time and appropriate tasks are allowed for tool acquisition or development.

When planning the use of tools, techniques, and methodologies, consider the following:

- a) A description of, or reference to, the methodology selected for the V&V approach
- b) Staff experience and training needed
- c) Special tools and specific techniques for the methodology
- d) How each tool and technique enhances the methodology
- e) Risks associated with a tool or technique
- f) Status of each tool
 - 1) *Is it a new acquisition?*
 - 2) *Are changes needed or is it completely ready for use?*
 - 3) *Is the required quantization available?*
 - 4) *Is its documentation acceptable?*
 - 5) *Is the tool proprietary?*
- g) Acquisition or development schedule
- h) Necessary support (hardware, other software)
- i) Alternate approach for a high-risk tool

5.5 Life cycle verification and validation

This subclause parallels 3.5 of the Standard, using the same life cycle model. The Standard points out that any life cycle model can be employed for development and V&V activities. The Standard can then be adapted to it as long as there is a clear mapping from one life cycle model to the other.

While many tasks can be thought of as going across life cycle phases, most SVVPs are likely to be written and managed with tasks assigned to a particular life cycle phase. Each of the following subclauses amplifies the requirements of the Standard. (See 4.2 of this guide for further guidance on planning the tasks that go across life cycle phases.)

Additionally, the Standard defines management as one of the phases. Of course, management is not a single life cycle phase activity, but rather an activity that goes across all of the individual phases and that ties all of the activities together into a coherent, useful program.

5.5.1 Management of V&V

(Section 5.1 of the Plan.) This section of the Plan shall address the seven topics identified in section 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, the 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 [3.5.1]*

The primary tasks of V&V management are planning, review, and control. The Standard maps these responsibilities into four fundamental tasks for V&V management—SVVP Generation, Baseline Change Assessment, Management Review, and Review Support. Conceptually, management's responsibility is to ensure the positive, successful interaction between the V&V activities and other software development activities to ensure the creation of defect-free software. The planning of the management tasks shall recognize the iterative nature of V&V tasks (e.g., determining when to reanalyze amended software products, when to revise V&V plans to reflect changes in the development process).

5.5.1.1 Software Verification and Validation Plan (SVVP) generation

Generate SVVP (during Concept Phase) for all life cycle phases in accordance with the 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. [Table 1, 5.1 (1)]

See annex A, A.3.4.

The Standard specifies the required content and format for an SVVP. From this starting point, V&V planners should tailor these requirements to the operating environment and the development environment. Tailoring considerations include the following:

- Software development environment and methodology
- Size and complexity of software being verified and validated
- Risk management considerations
- Organization and number of personnel performing V&V tasks
- Relationship of personnel performing V&V tasks to other development personnel, management, and user
- Approvals required for the SVVP
- Method of configuration control, maintenance, and updating of SVVP
- Special focus of V&V required (e.g., safety or security)

V&V planners may be dependent on things outside of their control. As a consequence, the plan may need to be able to respond to occasional, controlled changes to support the overall project milestones and objectives. At the same time it should remain constant in its quality goals and V&V objectives. Some of the uncertainties that should be considered include the following:

- Timely delivery of products
- Whether the quality of the products will be sufficient to allow meaningful verification and validation
- Availability of staff and resources needed to perform the planned tasks

V&V planning is most effectively performed in conjunction with the overall software development planning. Revisions to the plan are prepared regularly throughout the software development life cycle, usually at the completion or initiation of each life cycle phase. The plan, to be effective as a complete specification of the tasks required to achieve the established V&V objectives, shall be thorough, comprehensive, and specific.

The SVVP is often best developed incrementally. An SVVP will start small, based on an early, perhaps incomplete, view of the project. The SVVP will grow and incorporate changes as the software is developed. Eventually, the SVVP will be completed and serve as a record of the complete and detailed plan for the project's V&V activities.

This should not be taken to mean that the plan should be continually revised throughout the life cycle phases. Only those changes needed to provide further or more detailed task management for the future or to make corrections should be made. Revisions to the plan should not be made retrospectively; that is, as a way of removing deviations from the plan. Deviations that arise during the execution of the plan, necessary for project changes or external factors, are addressed in the Software Verification and Validation Reports (SVVRs).

Incremental planning may be prudent for at least two reasons.

- a) *It allows for a more realistic view of what is possible in planning in the early stages of a project.* Too often, much time and attention is spent overplanning future life cycle phases when there is the likelihood of significant changes in those phases. For example, certain design and implementation questions may be unanswerable in the concept phase and, as a consequence, precise planning for design and implementation of V&V may not be possible. As decisions are made and results and products obtained throughout the life cycle, more careful planning of subsequent phases is meaningful.
- b) *If a plan is to be effective (e.g., if people are to use it to identify responsibilities, to prepare for future tasks, to measure past performance) it should be realistic and achievable.* It should reflect the changes that are inherent in the other development process activities. It cannot be a static view of the desires of management at the outset of the V&V effort without regard to actual developments. The plan should represent reasonable and achievable objectives.

The SVVP is thus referred to as a living document because it may undergo changes and expansion over time. At its first publication (in most cases at the beginning of the life cycle) it will contain detailed plans about the near-term activities and only cursory descriptions of the later phases. When generating input for the SVVP, be prepared to revise and update the SVVP at fixed points. Most effectively, plan for review and revision at the end of each life cycle phase.

Additionally, during life cycle phases it sometimes may be advisable to incorporate revisions to the plan in response to significant changes in other project plans, and to include references to those revised plans. Plan for the maintenance of the SVVP under configuration identification and control. This can be accommodated by maintaining the SVVP on electronic media or in a hard copy format that can be easily updated.

The SVVP should be an item under configuration management. Plan to establish the baseline SVVP prior to the Requirements Phase if V&V begins early enough. In other cases, baseline the SVVP no later than at the end of the current life cycle phase. The use of a configuration identifier and change notations (such as change bars) will allow effective control of the configuration and status of the document.

5.5.1.2 Baseline change assessment

Evaluate proposed software changes (for example, anomaly corrections, performance enhancements, requirements 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. [Table 1, 5.1 (2)]

Baseline change assessment may be the most dynamic task performed during a V&V effort. Any change proposal could affect an unknown amount of previously completed development and V&V work. A V&V task should examine the change to determine the nature and extent of the V&V rework required by the change. Because changes will be proposed asynchronously, the SVVP cannot, before the fact, contain detailed planning for any given change assessment; rather, it shall contain general guidance for allocating resources to change proposals as they occur.

The request for baseline change assessment should be documented in a change proposal. The change proposal should indicate the perceived severity of the anomaly (if any), the immediacy of the change, and the systems affected so that the criticality of the software can be determined.

5.5.1.3 Management review of V&V

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. [Table 1, 5.1 (3)]

See annex A, A.3.5.

Management review comprises all of the general responsibilities of management for the monitoring, controlling, and reporting of the V&V effort and, in effect, the managing of this plan. In this task, V&V management is responsible for creating the positive interaction between the V&V activities and the other development activities. V&V management is responsible, in particular, for reviewing on-going efforts, accomplishments, and the use of resources. This section of the SVVP addresses only V&V-specific issues in project management review. While not stated earlier, a manager sufficiently technically expert in the system under development and in the process of software V&V is necessary to make informed and useful decisions throughout the project. This need is most keenly felt in the task of management review.

The results and findings of V&V tasks will have a significant impact on the development effort. (See also 5.7 of this guide for control and administrative procedures relating to the distribution of V&V products.) Early identification and correction of defects is one obvious benefit. The monitoring and reporting of levels and types of defects may identify other quality engineering issues in the development process.

Careful management attention to the technical quality and accuracy of the V&V products is particularly important. It is important that inaccurate or incorrect reports are not distributed to development or management personnel lest unnecessary effort be expended responding to illusory or imprecisely stated anomalies.

When reviewing reports, consider the following:

- a) *Is the report complete?*
- b) *Is the report technically accurate?*
- c) *Is the report judicious in its use of language?*
 - 1) *Does it restrict itself to the technical issues under discussion?*
 - 2) *Is it restricted to the tasking direction?*
 - 3) *Is its tone positive?*
 - 4) *Does it support development success?*
- d) *Does the report support effective management decision-making?*
- e) *Does it discriminate between levels of severity?*
- f) *Does it clearly identify impact?*
- g) *Does it propose alternate approaches?*

Prepare summary documents to make them as useful as possible to development and acquiring organization management. When organizing the results, consider the following:

- Summary assessment
- Itemization of critical outstanding problems and the impact of their not being resolved
- Recommendations about corrective course of action
- Summary of rescheduling impact, V&V, and resource requirements
- Distribution of anomalies according to some predefined categorization scheme and the significance of this distribution