IEEE Standard for Software Verification and Validation

IEEE Std 1012 – 2004

Revision of IEEE Std 1012-1998

Content

1. **Introduction & Definitions**
2. **V&V Objectives**
3. **Verification Process**
4. **Validation Process**
5. **Integrity Levels**
6. **Techniques**
7. **V&V Plan**
8. **Independent V&V (IV&V)**
9. **V&V Measures**
10. **Limitations**

Definitions

- **Verification**
  - Confirmation by examination and provisions of objective evidence that specified requirements have been fulfilled.
    - In design and development, verification concerns the process of examining the result of a given activity to determine conformity with the stated requirement for that activity.
- **Validation**
  - Confirmation by examination and provisions of objective evidence that the particular requirements for a specific intended use are fulfilled.
    - Validation is normally performed on the final product under defined operating conditions.
    - "Validated" is used to designate the corresponding status.

6/30/2008 3

Introduction

- **Purpose**
  - To help the development organization build quality into the software during the life cycle
- **Field of application**
  - A software system provides a capability to satisfy a stated need or objective by combining one or more of the following: processes, hardware, software, facilities, and people.
    - This relationship between the software and the system requires that software V&V processes consider software interactions with **all** system components.

6/30/2008 4

Introduction

- The V&V process addresses the following interactions with software:
  - **Environment:**
    - Determines that the solution represented in the software correctly accounts for all conditions, natural phenomena, physical laws of nature, business rules, and physical properties and the full ranges of the system operating environment.
  - **Operators/users:**
    - Determines that the software communicates the proper status/condition of the software system to the operator/user and correctly processes all operator/user inputs to produce the required results.
    - Validate that operator/user policies and procedures (e.g., security, interface protocols, data representations, system assumptions) are consistently applied
  - **Hardware:**
    - Determines that the software correctly interacts with each hardware interface and provides a controlled system response (i.e., graceful degradation) for hardware faults.
  - **Other software:**
    - Determines that the software interfaces correctly with other software components in the system in accordance with requirements and that errors are not propagated between software components of the system.

6/30/2008 5

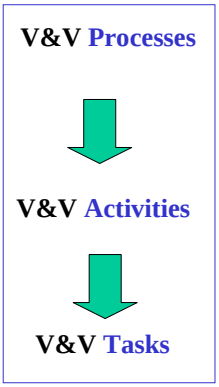
Introduction

- **Goals**
  - Determine if products of a given activity conform to the requirements of this activity
  - Ensure software satisfies the intended use and user needs
- **Execution of V & V Activities**
  - In parallel with the software development, not at the conclusion of development
- **Conformance**
  - *The word shall identifies mandatory requirements to claim compliance with this standard.*
  - *The words should and may indicate optional tasks that are not required to claim conformance to this standard.*

6/30/2008 6

Purpose of the Standard

1. Establish a common framework for **V&V Processes, Activities, and Tasks** in support of **all** software life cycle processes:
   - Acquisition, supply, development, operation, and maintenance processes.
2. Define the V&V tasks, required inputs, and required outputs
3. Identify the minimum V&V tasks corresponding to a **four-level** software integrity scheme
4. Define the content of a Software V&V Plan (SVVP)



6/30/2008 7

V&V Objectives

1. **V&V processes provide an objective assessment of software products and processes throughout the life cycle**
   - The assessment demonstrates that system and software requirements are:
     - correct
     - complete
     - accurate
     - consistent
     - testable

6/30/2008 8

V&V Objectives

2. **Facilitate early detection and correction of errors**
3. **Enhance management insight into**
   - Process risk
   - Product risk
4. **Support life cycle processes to ensure compliance with:**
   - Program performance
   - Schedule
   - Budget

Verification Process
«Are we building the product right ? »

- **Provides supporting evidence that software and its associated products:**
  1. Comply with requirements, e.g. correctness, completeness, for all life cycle activities during each life cycle process (e.g. acquisition, development)
  2. Satisfy standards, practices, conventions
  3. Establish a basis for assessing the completion of activities and initiating other activities

Validation Process
« Are we building the right product ? »

- **Provides supporting evidence that software satisfies system requirements allocated to software and solves the right problem**
  - e.g.
    - Correctly models physical laws,
    - Implement system business rules

Software Integrity Levels (SIL)

- A range of values that represent software complexity, criticality, risk, safety level, security level, desired performance, reliability, or other project-unique characteristics that define the **importance** of the software to the user and acquirer.
- Determine the **minimum** V&V tasks to be performed.
- Integrity level assigned to reused software products shall be in accordance with the integrity level scheme adopted for the project
- Tools that insert or translate code (e.g., optimizing compilers, auto-code generators)
  - shall be assigned the same integrity level as the integrity level assigned to the software element that the tool affects.

Example of Integrity Levels Scheme

4	Software element must execute correctly or grave consequences (loss of life, loss of system, economic or social loss) will occur. • No mitigation is possible.
3	Software element must execute correctly or the intended use (mission) of the system/software will not be realized, causing serious consequences (permanent injury, major system degradation, economic or social impact). • Partial to complete mitigation is possible.

Example of Integrity Levels Scheme

2.	Software element must execute correctly or an intended function will not be realized, causing minor consequences. • Complete mitigation possible.
1.	Software element must execute correctly or intended function will not be realized, causing negligible consequences. • Mitigation not required.

Software Integrity Levels
A Risk-Based Approach Example

Consequence	Definitions
Catastrophic	Loss of human life, complete mission failure, loss of system security and safety, or extensive financial or social loss.
Critical	Major and permanent injury, partial loss of mission, major system damage, or major financial or social loss.
Marginal	Severe injury or illness, degradation of secondary mission, or some financial or social loss.
Negligible	Minor injury or illness, minor impact on system performance, or operator inconvenience.

Assignment of Integrity Levels
An Example

Risk-Based Approach = Function of Consequence and Likelihood of Occurrence

Error consequence	Likelihood of occurrence of an operating state that contributes to the error			
	Reasonable	**Probable**	**Occasional**	**Infrequent**
Catastrophic	4	4	4 or 3	3
Critical	4	4 or 3	3	2 or 1
Marginal	3	3 or 2	2 or 1	1
Negligible	2	2 or 1	1	1

Minimum V&V Tasks – A Subset

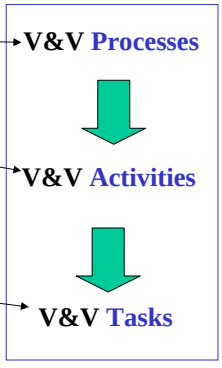
V&V activities	Requirements V&V			
	Levels			
	4	3	2	1
Planning the interface between the V&V effort and supplier				
Proposed/baseline change assessment	X	X	X	
Retirement assessment				
Risk analysis	X	X		
Scoping the V&V effort				
Software design evaluation	X	X		

Software integrity levels

6/30/2008 17

V&V Processes

- V&V processes support the primary processes of ISO/IEC 12207:
  - Management process, acquisition process, supply process, operation process, maintenance process, and Development process.
- **Development Process**
  - Contains the activities and tasks of the developer:
  - V&V Activities are organized into:
    - Concept V&V, Requirements V&V, Design V&V, Implementation V&V, Test V&V, and Installation and checkout V&V.
  - Requirements V&V Tasks
    - Software requirements evaluation, Interface analysis, Criticality analysis, System V&V test plan generation, Acceptance V&V test plan generation, Configuration management assessment, Hazard analysis, Security analysis, Risk analysis, Traceability analysis*.



6/30/2008 18

V&V for the Management Process

- **Activity: Management of the V&V effort**
  - Monitors and evaluates all V&V outputs
  - **Tasks**:
    - SVV Plan generation
    - Proposed/baseline change assessment
    - Management review of the V&V effort
    - Management and technical review support
    - Interface with organizational and supporting processes
    - Identify process improvement opportunities in the conduct of V&V

V&V for the Acquisition Process

- The Acquisition Process begins with the definition of the need to acquire a system, software product, or software service.
- Continues with:
  - preparation and issuance of a request for proposal,
  - selection of a supplier,
  - management of the acquisition process through to the acceptance of the system, software product, or software service.
- **Activity: Acquisition support V&V**
  - Addresses project initiation, RFP, contract preparation, supplier monitoring, and acceptance and completion
  - **Tasks:**
    - Scoping* the V&V effort
    - Planning the interface between the V&V effort and supplier
    - System requirements review
    - Acceptance support

Scoping the V&V effort

V&V tasks	Required inputs	Required outputs
(1) Scoping the V&V effort a) Determine the software characteristics (e.g., complexity, criticality, risk, safety level, security level, desired performance, reliability, or other project-unique characteristics) that define the importance of the software to the user. b) Adopt the system integrity scheme assigned to the project. If no system integrity level scheme exists, then one is selected. c) Assign a software integrity level to the system and the software. d) Establish the degree of independence (see Annex C), if any, required for the V&V. e) Determine the minimum V&V tasks for the software integrity level using Table 2 and the selected software integrity level scheme. f) Determine the extent of V&V on reuse software selected for the program (see Annex D). g) Determine the extent of V&V for tools that insert or translate code (e.g., optimizing compilers, auto-code generators). h) Augment the minimum V&V tasks with optional V&V tasks, as necessary. i) Provide an estimate of the V&V budget, including test facilities and tools as required.	Preliminary system description Statement of need Draft RFP or tender System integrity level scheme	SVVP

V&V for the Supply Process

- The Supply process is initiated by either a decision to prepare a proposal or by negotiating, finalizing, and entering into a contract with the acquirer to provide the system, software product, or software service.
- **Activity: Planning V&V**
  - Addresses the initiation, preparation of response, contract, planning, execution and control, review and evaluation, and delivery and completion activities.
  - **Tasks:**
    - Planning the interface between the V&V effort and supplier
    - Contract verification

V&V for the Development Process

- The development process contains the activities and tasks of the developer.
- **V&V activities**
  - **Concept V&V**
    - System architecture is selected
    - System requirements are allocated to hardware, software, and user interface components
  - **Requirements V&V**
    - Ensure the correctness, completeness, accuracy, testability, and consistency of the system software requirements.
    - **Tasks:**
      - Traceability analysis*, Software requirements evaluation, Interface analysis, Criticality analysis, System V&V test plan generation, Acceptance V&V test plan generation, Configuration management assessment, Hazard analysis, Security analysis, Risk analysis
  - **Design V&V**
  - **Implementation V&V**
  - **Test V&V**
  - **Installation and checkout V&V**

Traceability Analysis Task

V&V tasks	Required inputs	Required outputs
(1) Traceability analysis Trace the software requirements (SRS and IRS) to system requirements (concept documentation) and system requirements to the software requirements. Analyze identified relationships for correctness, consistency, completeness, and accuracy. The task criteria are a) Correctness Validate that the relationships between each software requirement and its system requirement are correct. b) Consistency Verify that the relationships between the software and system requirements are specified to a consistent level of detail. c) Completeness 1) Verify that every software requirement is traceable to a system requirement with sufficient detail to show conformance to the system requirement. 2) Verify that all system requirements related to software are traceable to software requirements. d) Accuracy Validate that the system performance and operating characteristics are accurately specified by the traced software requirements.	Concept documentation (system requirements) SRS IRS	Task Report(s)—Traceability analysis Anomaly report(s)

V&V for the Operation Process

- The operation process involves the use of the software system by the end user in an operational environment.
- **Activity: Operation V&V**
  - Evaluates the impact of changes in the operating environment
  - **Tasks:**
    - Evaluation of new constraints
    - Operating procedures evaluation
    - Hazard analysis
    - Security analysis
    - Risk analysis

V&V for the Maintenance Process

- The maintenance process is activated when the software system or associated documentation must be changed in response to a need for system maintenance.
- **Activity: Maintenance V&V**
  - System modifications may be derived from requirements specified to correct software errors; to adapt to a changed operating environment; or to respond to additional user requests or enhancements
  - **Tasks**
    - SVV Plan revision
    - Anomaly evaluation
    - Criticality analysis
    - Migration assessment
    - Retirement assessment
    - Hazard analysis
    - Security analysis
    - Risk analysis
    - Task iteration

V & V Techniques for Software: Three Major Classes

1. **Static analysis**
   - analyze the form and structure of a product without executing the product
2. **Dynamic analysis**
   - involve execution, or simulation, of a development activity product to detect errors by analyzing the response of a product to sets of input data
3. **Formal analysis**
   - use of rigorous mathematical techniques to analyze the algorithms of a solution

Wallace, D., et al, 'Reference Information for the Software Verification and Validation Process', NIST Special Publication 500-234, 1996.

6/30/2008 27

V&V Techniques

Algorithm analysis	Finite state machines(FSM)	Sensitivity analysis
Analytic modeling	Functional testing	Simulation
Boundary value analysis	Inspections	Sizing and timing analysis
Code reading	Interface analysis	Slicing
Control flow analysis	Interface testing	Software failure mode, effects and criticality analysis
Coverage analysis	Mutation analysis	Software fault tree analysis
Critical timing/flow analysis	Performance testing	Stress testing
Database analysis	Petri-nets model	Structural testing
Data flow analysis	Proof of correctness	Symbolic execution
Decision (truth) tables	Prototyping	Test certification
Desk checking	Regression analysis and testing	Walkthroughs
Error seeding	Requirements parsing	
Event tree analysis	Reviews	

Wallace, D., et al, 'Reference Information for the Software Verification and Validation Process', NIST Special Publication 500-234, 1996.

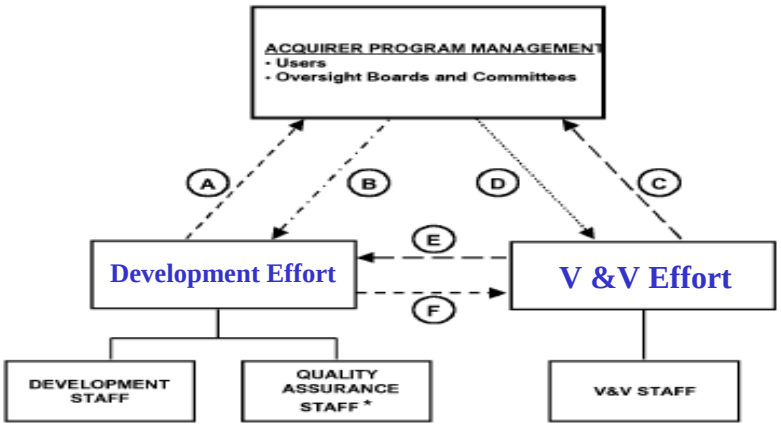
6/30/2008 28

V&V Techniques

- Algorithm analysis
- Analytic modeling
- Boundary value analysis
- Code reading
- Control flow analysis
- Coverage analysis
- Critical timing/flow analysis
- Database analysis
- Data flow analysis
- Decision (truth) tables
- Desk checking
- Error seeding
- Event tree analysis
- Finite state machines(FSM)
- Functional testing
- Inspections
- Interface analysis
- Interface testing
- Mutation analysis
- Performance testing
- Petri-nets model
- Proof of correctness
- Prototyping
- Regression analysis and testing
- Requirements parsing
- Reviews
- Sensitivity analysis
- Simulation
- Sizing and timing analysis
- Slicing
- Software failure mode, effects and criticality analysis
- Software fault tree analysis
- Stress testing
- Structural testing
- Symbolic execution
- Test certification
- Walkthroughs

Wallace, D., et al, 'Reference Information for the Software Verification and Validation Process', NIST Special Publication 500-234, 1996.

V&V Organizational Relationships



A: **Submittal** of program documentation
B: **Approval, denial, and recommendations** on development issues and deliverables listed in A.
C: **Submittal** of SVVP, V&V task results, anomaly reports, activity reports, and other special reports.
D: **Approval, denial, and recommendations** on V&V issues and deliverables listed in C.
*: The quality assurance staff may report directly to the Quality Assurance Director rather than through the development organization.

Software V&V Plan - Outline

1. **Purpose**
2. **Reference documents**
3. **Definitions**
4. **V&V overview**
   - Integrity Level Scheme
5. **V&V processes**
   - Management, acquisition, supply, development, operation, maintenance.
6. **V&V reporting requirements**
7. **V&V administrative requirements**
8. **V&V documentation requirements**

Independent V&V (IV&V)

- **Function of 3 parameters**
  - Technical, Managerial and Financial independence

1. Technical independence

- Utilize personnel who are not involved in the development of the software.
- Must formulate its own understanding of the problem and how the proposed system is solving the problem.
- "Fresh viewpoint" is an important method to detect subtle errors overlooked by those too close to the solution.

Independent V&V (IV&V)

2. Managerial independence

- Organization separate from the development and program management organizations.
  1. Selects the segments of the software and system to analyze and test,
  2. Chooses the IV&V techniques,
  3. Defines the schedule of IV&V activities,
  4. Selects the specific technical issues and problems to act upon.
- Submit to program management the IV&V results, anomalies, and findings without any restrictions
  - e.g., **without** requiring prior approval from the development group or adverse pressures, direct or indirect, from the development group.

Independent V&V (IV&V)

3. Financial independence

- Control of the IV&V budget be vested in an organization independent of the development organization.
- Prevents situations where the IV&V effort cannot complete its analysis or test or deliver timely results because funds have been diverted or adverse financial pressures or influences have been exerted.

V&V Measures

- The V&V measures should consider the software integrity level assigned to the software and system, application domain, project needs, and current industry practices.

- **Three categories of measures:**
  1. For evaluating anomaly density
     - e.g. Requirements anomaly density (# requirements anomalies found /# requirements reviewed)

  2. For evaluating V&V effectiveness
     - Characterize the added benefits of V&V to discover anomalies in software products and processes.

     - Delineate the percentage of the total anomalies found by the V&V effort.
       - e.g. Requirements V&V effectiveness (# anomalies found by V&V / # anomalies found by all sources)

V&V Measures

3. For evaluating V&V efficiency
   - Characterize the capability of the V&V effort to discover anomalies in software products and processes in the development activity in which they are injected

     - e.g. Requirements V&V efficiency (# Req Anomalies found by V&V in Req Activities / # Req anomalies found by V&V in all activities) X 100%

V&V Limitations

1. **Impracticality of Testing All Data**
   - For most programs, it is impractical to attempt to test the program with all possible inputs, due to a combinatorial explosion.
2. **Impracticality of Testing All Paths**
   - For most programs, it is impractical to attempt to test all execution paths through the product, due to a combinatorial explosion
3. **No Absolute Proof of Correctness**
   - Howden* claims that there is no such thing as an absolute proof of correctness.
     - Unless a formal specification can be shown to be correct and, indeed, reflects exactly the user's expectations, no claim of product correctness can be made.

* Schulmeyer, G., 'Verification & Validation of Modern Software Intensive Systems', Prentice Hall, 2000.

6/30/2008 37

Summary

1. **Introduction**
2. **Key Concepts**
3. **V&V Objectives**
4. **Verification Process**
5. **Validation Process**
6. **Integrity Levels**
7. **Techniques**
8. **V&V Plan**
9. **IV&V**
10. **V&V Measures**
11. **Limitations**

6/30/2008 38