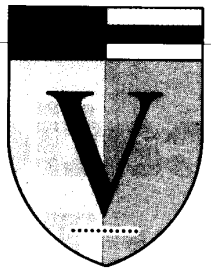

Software Verification and Validation: An Overview

Dolores R. Wallace, *National Institute of Standards and Technology*
Roger U. Fujii, *Logicon*

Properly applied throughout the life cycle, verification and validation can result in higher quality, more reliable programs. This article introduces the V&V method and standards to support it.

Verification and validation is one of the software-engineering disciplines that help build quality into software. V&V is a collection of analysis and testing activities across the full life cycle and complements the efforts of other quality-engineering functions. This overview article explains what V&V is, shows how V&V groups' efforts relate to other groups' efforts, describes how to apply V&V, and summarizes evaluations of V&V's effectiveness. The box on p. 11 describes standards and guidelines for planning and managing a V&V effort.

What is V&V?

V&V comprehensively analyzes and tests software to determine that it performs its intended functions correctly, to ensure that it performs no unintended functions, and to measure its quality and reliability. V&V is a systems-engineering discipline to evaluate software in a systems context. Like systems engineering, it uses a structured approach to analyze and test the software against all system functions and against hardware, user, and other software interfaces.

Verification involves evaluating software during each life-cycle phase to ensure that it meets the requirements set forth in the previous phase. Validation involves testing software or its specification at the end of the development effort to ensure that it meets its requirements (that it does what it is supposed to). While "verification" and "validation" have separate definitions, you can derive the maximum benefit by using them synergistically and treating "V&V" as an integrated definition.

Ideally, V&V parallels software development and yields several benefits:

- It uncovers high-risk errors early, giving the design team time to evolve a comprehensive solution rather than forcing a makeshift fix to accommodate development deadlines.
- It evaluates the products against system requirements.
- It gives management continuous and

0740-7459/89/0500/0010/$01.00 © 1989 IEEE

comprehensive information about the quality and progress of the development effort.

• It gives the user an incremental preview of system performance, with the chance to make early adjustments.

The V&V tasks in Table 1 are the minimum tasks for the development phases required by ANSI/IEEE Std 1012-1986, *Standard for Software Verification and Validation Plans.* The V&V standard specifies minimum input and output requirements for each V&V task. A V&V task may not begin without specific inputs, and is not completed until specific outputs are completed.

You can tailor a V&V effort by adding or deleting tasks to or from the minimum set. Table 2 lists some optional V&V tasks and considerations that you might use to assign the tasks to your V&V group. You can apply these tasks to different life-cycle models by mapping the traditional waterfall phases to the new model. Examples include variations of the traditional waterfall, Boehm's spiral development,[1] rapid prototyping, and evolutionary development models.

Where V&V fits in

Because V&V should occur throughout the life cycle, applying it involves many groups. Furthermore, V&V and other groups complement each other's software-quality responsibilities:

• The software-development group builds the product to satisfy the established quality and performance requirements. The group relies on its quality-assurance group, systems engineers, requirements analysts, designers, programmers, testers, data-management and configuration-management specialists, documentation specialists, and others.

• The quality-assurance group verifies that the development process and products conform to established standards and procedures. Using reviews, audits, inspec-

V&V standards and guidelines

The concepts of V&V emerged in the late 1960s and 1970s as the use of software in military and nuclear-power systems increased. Initially, individual programs' standards addressed the need for V&V. Then government and industry began to develop V&V standards so they would have a specification of this methodology for contract procurements and for monitoring the technical performance of V&V efforts. Today's V&V standards and guidelines serve large, heterogeneous communities and are applicable to many types of software. They include:

• Federal Information-Processing Standards Publication 101, *Guideline for Life-Cycle Validation, Verification, and Testing of Computer Software,*

• FIPS Pub. 132, *Guideline for Software Verification and Validation Plans,* which adopts ANSI/IEEE Std 1012-1986, the *Standard for Software Verification and Validation Plans,*

• the US Air Force's AFSC/AFLC 800-5, *Software Independent Verification and Validation,*

• the American Nuclear Society's ANS 10.4, *Guidelines for the Verification and Validation of Scientific and Engineering Computer Programs for the Nuclear Industry,* and

• the NASA Jet Propulsion Laboratory's JPL D-576, *Independent Verification and Validation of Computer Software: Methodology.*

Table A shows you how to develop a V&V effort based on the strength of the guidance in these standards and guidelines.

Table A.
Planning V&V with guidance from V&V documents.

Activity	Procedure	Guidance
Scope the V&V effort	Criticality assessment	AFSC 800-5
	Organization	AFSC 800-5, ANS 10.4
	Cost estimation	AFSC 800-5
Plan the V&V effort	Planning preparation	FIPS 132/IEEE 1012, ANS 10.4, FIPS 101
	Objectives	FIPS 132/IEEE 1012, FIPS 101, ANS 10.4, JPL D-576
	General V&V task selection	all
	Minimum, required	FIPS 132/IEEE 1012
	Optional	FIPS 132/IEEE 1012
	Criticality levels	FIPS 101, AFSC 800-5
	Test management	FIPS 132/IEEE 1012
	Test types	FIPS 132/IEEE 1012, FIPS 101, JPL D-576
	Objectives	FIPS 132/IEEE 1012
	Documentation	FIPS 132/IEEE 1012
	Coverage	FIPS 101, FIPS 132/IEEE 1012, ANS 10.4
	Planning	all
	Planning V&V for maintenance	ANS 10.4, FIPS 132/IEEE 1012
Manage the V&V effort	V&V management tasks	FIPS 132/IEEE 1012
	Reporting	FIPS 132/IEEE 1012, ANS 10.4

Table 1.
Minimum set of recommended V&V tasks.

Phase	Tasks	Key issues
Concept	Concept-documentation evaluation	Satisfy user needs; constraints of interfacing systems
Requirements definition	Traceability analysis	Trace of requirements to concept
	Requirements validation	Correctness, consistency, completeness, accuracy, readability, and testabililty; satisfaction of system requirements.
	Interface analysis	Hardware, software, and operator interfaces
	Begin planning for V&V system testing	Compliance with functional requirements; performance at interfaces; adequacy of user documentation; performance at boundaries
	Begin planning for V&V acceptance testing	Compliance with acceptance requirements
Design	Traceability analysis	Trace of design to requirements
	Design evaluation	Correctness, consistency, completeness, accuracy readability, and testability ; design quality.
	Interface analysis	Correctness, consistency, completeness, accuracy readability, and testability; data items across interface
	Begin planning for V&V component testing	Compliance to design; timing and accuracy; performance at boundaries
	Begin planning for V&V integration testing	Compliance with functional requirements; timing and accuracy; performance at stress limits
Implementation	Traceability analysis	Trace of source code to design
	Code evaluation	Correctness, consistency, completeness, accuracy, readability, and testability; code quality
	Interface analysis	Correctness, consistency, completeness, accuracy, readability, and testability; data/control access across interfaces
	Component test execution	Component integrity
Test	V&V integration-test execution	Correctness of subsystem elements; subsystem interface requirements
	V&V system-test execution	Entire system and at limits and user stress conditions
	V&V acceptance-test execution	Performance with operational scenarios
Installation and checkout	Installation-configuration audit	Operations with site dependencies; adequacy of installation procedure
	V&V final-report generation	Disposition of all errors; summary of V&V results

tions, and walkthroughs, it acts as a formal check and balance to monitor and evaluate software as it is being built.

- The systems-engineering group ensures that the product satisfies system requirements and objectives. It uses techniques like simulation to gain reasonable assurance that the requirements are satisfied.
- The configuration-management and data-management groups monitor and control the program versions and data during product development, using techniques like formal audits, change-control records, requirements traceability, and sign-off records. The user organization must provide assurance that the software satisfies user requirements and operational needs. Typically, it uses techniques like formal design reviews and acceptance testing.
- Like the systems-engineering group, the V&V group is responsible for verifying that the product at each life-cycle phase satisfies quality attributes (like correctness) and that at each phase it satisfies the requirements of the previous phase. The V&V group is also responsible for validating that the software satisfies system requirements and objectives.

While its activities are directed at the software, the V&V group must also consider how the software interacts with the system, including hardware, users, other software, and other external systems. The V&V group maintains its own configuration- and data-management functions on program, data, and documentation received from the development team to ensure that V&V discrepancy reports are made against controlled documents and to repeat V&V tests against controlled

Table 2.
Optional V&V tasks and suggested applications.

Tasks	Phases* M C R D I T X O	Considerations
Algorithm analysis	□ □ ■ ■ ■ ■ □ ■	Numerical and scientific software using critical equations or models
Audit performance		When the V&V group is part of the quality-assurance or user group, for large developments to help quality-assurance or f or user-group staff audits
Configuration control	□ □ □ □ ■ ■ ■ ■	
Functional	□ □ □ □ ■ ■ ■ ■	
In-process	□ □ ■ ■ ■ ■ ■ ■	
Physical	□ □ □ □ □ ■ ■ ■	When the V&V group is part of a systems-engineering group or is independent, for large software developments
Audit support		
Configuration control	□ □ □ □ ■ ■ ■ ■	
Functional	□ □ □ □ ■ ■ ■ ■	
In-process	□ □ ■ ■ ■ ■ ■ ■	
Physical	□ □ □ □ □ ■ ■ ■	
Configuration management	■ ■ ■ ■ ■ ■ ■ ■	When the V&V group is part of the user group
Control-flow analysis	□ □ ■ ■ ■ □ □ □	Complex, real-time software
Database analysis	□ □ ■ ■ ■ ■ □ ■	Large database applications; if logic is stored as parameters
Dataflow analysis	□ □ ■ ■ ■ □ □ ■	Data-driven real-time systems
Feasibility-study evaluation	□ ■ □ □ □ □ □ ■	High-risk software using new technology or concepts
Installation and checkout test	□ □ ■ ■ ■ ■ ■ ■	When the V&V group is part of the systems-engineering or user group
Performance monitoring	□ □ □ □ □ □ □ ■	Software with changeable man-machine interfaces
Qualification testing**	□ □ ■ ■ ■ ■ ■ ■	When the V&V group is part of the systems-engineering or user group
Regression analysis and testing	□ □ ■ ■ ■ ■ ■ ■	Large, complex systems
Reviews support		When the V&V group is part of the systems-engineering or user group
Operational readiness	□ □ □ □ □ □ ■ ■	
Test readiness	□ □ □ □ ■ ■ ■ ■	
Simulation analysis	□ □ □ ■ ■ ■ □ ■	No system-test capability or the need to preview the concept for feasibility or the requirements for accuracy
Test certification	□ □ □ □ □ ■ ■ ■	For critical software
Test evaluation	□ □ ■ ■ ■ ■ ■ ■	When the V&V fgroup is part of the quality-assurance or user group
Test witnessing	□ □ □ □ □ ■ ■ ■	When the V&V group is part of the quality-assurance, user, or systems-engineering group
User-documentation evaluation	□ ■ ■ ■ ■ ■ ■ ■	Interactive software requiring user inputs
V&V tool-plan generation	■ □ □ □ □ □ □ ■	When acquiring or building V&V analysis and test tools
Walkthroughs		When the V&V group is part of the quality-assurance or systems-engineering group, for large software developments to staff walkthroughs
Design	□ □ □ ■ □ □ □ ■	
Requirements	□ □ ■ □ □ □ □ ■	
Source code	□ □ □ □ ■ □ □ ■	
Test	□ □ □ □ ■ ■ ■ ■	

*Phase codes: M=management, C=concept, R=requirements, D=design, I=implementation, T=test, X=installation/checkout, and O=operations/maintenance
**Test plan, test design, test cases, test procedures, and test execution

software releases.

The V&V group's documentation evaluation and testing are different from those conducted by other groups. The quality-assurance group reviews documents for compliance to standards and performs a logical check on the technical correctness of the document contents. The V&V group may perform in-depth evaluations by activities like rederiving the algorithms from basic principles, computing timing data to verify response-time requirements, and developing control-flow diagrams to identify missing and erroneous requirements. The V&V group may suggest more optimal approaches.

The V&V group's testing is usually separate from the development team's testing. In some cases, the V&V group may use development test plans and results and supplement them with additional tests.

A major influence on the responsibilities of a V&V group and its relationship to other groups is to whom the V&V group reports. There are four ways to organize a V&V effort:

- Independent. The traditional approach is that the V&V group is independent of the development team and is called "independent V&V" or "IV&V." In this relationship, the V&V group establishes formal procedures for receiving software releases and documentation from the development group. The V&V group sends all its evaluation reports and discrepancy reports to both the user and

development groups. To maintain an unbiased technical viewpoint, the V&V group does not use any results or procedures from the quality-assurance or systems-engineering groups.

The independent V&V group's tasks are oriented toward engineering analysis (like algorithm analysis and control-flow and dataflow analysis) and comprehensive testing (like simulation). The objective is to develop an independent assessment of the software's quality and to determine whether the software satisfies critical system requirements.

The advantages of this approach are detailed analysis and test of software requirements, an independent determination of how well the software performs, and early detection of high-risk software and system errors. The disadvantages are higher costs and additional development interfaces.

- Embedded in the system-engineering group. When the V&V group is embedded in the systems-engineering group, its tasks are to review the group's engineering analyses (like algorithm development, sizing, and timing) and testing (like test evaluation and review of the adequacy of the development test-planning document). In some instances, the V&V group may be the independent test team for the systems-engineering group, sharing its data. The V&V group's results are reviewed and monitored by the systems-engineering and quality-assurance groups. An independent V&V group reporting to the systems-engineering group is another form of this organizational approach.

The advantages to using systems-engineering personnel in the V&V tasks are minimum cost to the project, no system learning for the staff, and no additional development interfaces. A disadvantage is the loss of objective engineering analysis.

- Embedded in the quality-assurance group. When the V&V group is embedded in the quality-assurance group, its tasks are monitoring, auditing, and reviewing content (through tasks like audit performance, audit support, test witnessing, walkthrough support, and documentation review). In these tasks, the V&V group works as part of the quality-assurance group and maintains its relationship to the systems-engineering and other development groups in the same way that the quality-assurance group does.

The advantages of embedding the V&V group in the quality-assurance group are low cost to the project and the entry of V&V analysis capabilities into reviews, audits, and inspections. A disadvantage is the loss of independent systems analysis and testing.

- Embedded in the user group. When the V&V group is embedded in the user group, its tasks are an extension of the user group's responsibilities. Its tasks are configuration-management support of products under development, support of formal reviews, user-documentation evaluation, test witnessing, test evaluation of the development test-planning documents, and user-testing support (like user-acceptance testing and installation and checkout testing). As an extension of the user group, the V&V group would receive formal software deliverables and provide comments and data to the development's project management that distributes the information to its own development team.

V&V should focus on identifying and eliminating the high risks a project is likely to encounter. V&V management must define and use methods of risk management.

An advantage of this approach is the strong systems-engineering and user perspectives that can be brought to bear on the software during development. The disadvantages are loss of detailed analysis and test of incremental software (because these incremental versions typically are not formal deliverables) and loss of error detection and feedback to the development group (because of the constraints caused by the frequency of formal product deliverables). If the user group has an independent V&V group reporting to it, these disadvantages can be overcome — but the price is an additional development interface.

Applying V&V

V&V tasks span the entire development effort, and several of these tasks affect the selection of development techniques and tools. The V&V group's management plans the V&V process, coordinates and interprets its performance and quality, reports discrepancies promptly to the development group, identifies early problem trends and focuses its activities on them, provides a technical evaluation of the product's performance and quality at each major program review, and assesses the full effects of proposed product changes. The group's management produces a V&V plan, task reports, phase summary reports, a final report, and a discrepancy report.

Boehm and Papaccio[2] have reported that Pareto analysis (which shows that 20 percent of the problems cause 80 percent of the rework costs) applies to software and have recommended that V&V "focus on identifying and eliminating the specific high-risk problems to be encountered by a software project." Part of the V&V management activities is to define and use methods to address these problems of rework and risk management.

One method for reducing rework costs is to provide early delivery of information (like draft portions of incremental documents) and software builds to the V&V group. (A software build represents a basic program skeleton containing portions of the full software capabilities. Each successive build integrates additional functions into the skeleton, permitting early software deliveries to the V&V group in an orderly development process.) Based on discrepancy or progress reports, program managers can make the technical and management decisions to refocus the V&V and development groups onto the product's problem areas.

Another method is criticality analysis, which can reduce high-risk problems. You perform it at the beginning of a project to identify the functions and modules required to implement program functions or to identify quality requirements whose failure would cause a safety or security

hazard or a large financial or social loss. You usually identify and trace these through a block or control-flow diagram of the system and its software; each block or control-flow box represents a system or software function (module).

You repeat the analysis for each lifecycle phase to observe whether the implementation details shift the emphasis of the criticality. You can combine the criticality analysis with the cross-reference matrix of Table 1 to identify which V&V techniques are needed for a project.

Techniques. There are many V&V techniques that you can use. But how do you know which ones to use at each phase? Table 3 matches 41 techniques against some of they V&V issues they address. Which phases you apply these techniques to depends on each project's characteristics and V&V objectives.[3] Among the table's 41 techniques are five likely, nonexclusive techniques (highlighted in gray) for determining the feasibility of a software concept and its requirements:

- Requirements parsing separates the desired performance requirements from other requirements data.
- Analytical modeling assesses the desired performance capability.
- Simulations of the proposed operating environment let you execute test data to determine whether the resulting performance matches the desired performance.
- Criticality analysis identifies the critical functions and their distribution in the system architecture.
- Test-data generation defines the performance limits of the proposed system requirements. You can verify predicted performance by using simulation to execute the test scenario.

A forthcoming report[4] will include the full matrix on which Table 3 is based.

Table 3.
Matrix of selected V&V issues and V&V techniques and tools. Gray rows are techniques described in the text.

Techniques and tools	Acceptance tests	Bottlenecks	Environment interaction	Execution characteristics	Execution support	Feasibility	Portability	Processing efficiency	Requirements evaluation	System-performance prediction
	V&V issues									
Algorithm analysis								■		■
Analytical modeling		■				■		■	■	■
Assertion generation										
Assertion processing										
Cause-effect graphing		■								
Code auditor							■	■		■
Comparator										
Control-flow analyzer		■	■					■		■
Criticality analysis	■					■	■		■	
Cross-reference generator										
Database analyzer								■		
Dataflow analyzer		■	■					■		■
Design-compliance analyzer										
Execution-time estimator		■		■				■		■
Formal review										
Formal verification										
Functional testing				■						■
Inspections (Fagan)										
Interactive test aids		■	■	■	■			■		
Interface checker			■						■	
Metrics				■				■	■	
Mutation analysis										
Program-description-language processor										
Peer review										
Physical-unit testing										
Regression testing	■									
Requirements parsing	■		■			■			■	
Round-off analysis										
Simulations			■	■	■	■				■
Sizing										
Software monitors		■	■	■	■			■		■
Specification base	■	■	■	■						■
Structural testing		■		■						
Symbolic execution				■						
Test drivers	■		■	■	■					
Test-coverage analyzer				■						
Test-data generator	■		■		■	■				■
Test-support facilities	■		■	■						■
Timing		■						■		
Tracing	■									
Walkthroughs										

Testing. V&V test activities span many phases, from requirements through installation and checkout. V&V testing activities continue into the operations and maintenance phase to address any changes made to the software after initial delivery.

A comprehensive test-management approach recognizes the differences in objectives and strategies of different types of testing. Each of four test-planning activities — component, integration, system, and acceptance testing — produces test-plan, test-design, test-case, and test-procedure documents:

- Component testing verifies the design and implementation of software units, modules, or subelements.
- Integration testing verifies functional requirements as the software subelements are integrated, directing attention to in-

ternal software interfaces and external hardware and operator interfaces.

- System testing validates the entire program against system requirements and performance objectives.
- Acceptance testing validates the software against V&V acceptance criteria, defining how the software should perform with other completed software and hardware.

V&V system testing uses a laboratory environment in which some system features are simulated or performed by hardware or software that will not be in the final user environment. Acceptance testing uses the operational environment.

In addition to testing against system and software requirements, effective testing requires a comprehensive understanding of the system. You develop such understanding from systematically analyzing the software's concept, requirements, design, and code.

V&V testing includes structural (white-box) testing that requires knowledge of internal software details. It is effective when probing for errors and weaknesses to reveal hidden faults in regions where some test cases for functional (black-box) testing can produce "correct" output despite internal errors.

Another V&V test technique is to develop test cases that violate software requirements. This approach is effective at uncovering basic design-assumption errors and unusual operational-use errors. These types of errors escape initial detection because they are obscure or so simple that you fail to design for them. V&V test planning is as effective for detecting errors as test executions are for uncovering software faults.

Operations and maintenance. For each software change made in the operations and maintenance phase, you repeat all development V&V activities in Table 1 to ensure that nothing is overlooked. You should add or delete V&V activities to address the type of software change made, but take care in doing so because small changes may have subtle but significant side effects. In many cases, an examination of the proposed software change shows that the V&V group must repeat its activities on only a small part of the product. Some V&V activities, like concept-documentation evaluation, may require little or no effort to verify a small change.

How effective is V&V?

Two studies that evaluated the effectiveness of V&V used different data and reported on different factors. While they can't be directly compared, the studies do provide insights on V&V's effectiveness.

One 1982 study by McGarry[5] reported that V&V was *not* an effective approach on three projects at the Software Engineering Laboratory at the National Aeronautics and Space Administration's Goddard Space Flight Center. In the study, three flight-dynamics projects ranging in size from 10,000 to 50,000 lines of code were selected. A V&V group was involved in requirements and design verification, separate system testing, and validation of consistency from start to finish. The project lasted 18 months and used an average of 1.1 people, peaking at three people. Results included:

A comprehensive test-management approach recognizes the differences in objectives and strategies of different types of testing.

- Productivity of the development groups was the lowest of any previously monitored SEL project (due to the cost of the V&V interface).
- Rates of errors found early in the development cycle were better than usual.
- The V&V effort found 2.3 errors per thousand lines of code.
- The cost of fixing all discovered errors was no less than in any other SEL project.
- The software's reliability (defined as the error rate during acceptance and maintenance and operations) was no different from other SEL projects.

However, a 1981 study by Radatz[6] for the Air Force's Rome (N.Y.) Air Development Center reported that V&V *was* effective for four large independent V&V projects ranging from 90,000 to 176,000 lines of code. The projects were real-time command-and-control, missile-tracking, and avionics programs, as well as a time-critical, batch trajectory-computation program. The projects took from 2.5 to four years to develop. Two projects started using V&V at the requirements phase; two others started at the coding phase. The V&V groups used five to 12 people per project. Results included:

- Errors were detected early in the development — 50 percent to 89 percent were detected before development testing began.
- Many discrepancies (1,259) were reported — an average of more than 300 per program.
- The V&V effort found an average of 5.5 errors per thousand lines of code.
- More than 85 percent of the errors affected reliability and maintainability.
- Programmer productivity improved (as measured by subtracting the time required to evaluate the V&V error reports from the programming time saved by the programmers' not having to find the error). The savings per error was 1.3 to 6.1 hours of programmer time and more than seven minutes of computer time.
- For the projects starting at the coding phase, the savings from early error detection were 20 to 28 percent of independent V&V costs. For the projects starting at the requirements phase, the savings from early error detection were 92 to 180 percent of independent V&V costs.

These studies showed that V&V can improve quality, cause more stable requirements, cause more rigorous development planning (at least to interface with the V&V group), catch errors earlier, promote better schedule compliance and progress monitoring, make project management more aware of interim quality and progress, and result in better criteria and results for decision making at formal reviews and audits.

But V&V has several negative effects: It adds 10 to 30 percent to the development cost, requires additional interfaces between project groups, can lower developer productivity if programmers and engineers spend time explaining the system

to V&V analysts when trying to resolve invalid anomaly reports, adds to the documentation requirements if the V&V group is receiving incremental program and documentation releases, requires the sharing of computing facilities and classified data with the V&V group, and increases the paperwork to provide written responses to the V&V group's error reports and other V&V data requirements.

As the Radatz study showed, you are more likely to recover V&V costs when you start using it early in the requirements phase. You should consider the interface activities between development and V&V groups for documentation, data, and software deliveries an inherently necessary step to evaluate intermediate development products. This is a necessary by-product of doing what is right from the outset. The cost of the development interface is minimal, and sometimes nonexistent, when the V&V assessment is independent of the development group.

To offset unnecessary costs, the V&V group must organize its activities to focus on the software's critical areas so it can uncover critical errors and thus significantly save development costs. The V&V group must use criticality analysis to identify critical areas and it must scrutinize each discrepancy report before release to ensure that no inaccurate information is released so the development group will not waste time on inaccurate or trivial reports.

To eliminate the need for the development group to train the V&V group, you must select a V&V staff that is experienced and knowledgeable about the software and its application. When V&V engineers and computer scientists reconstruct the details and idiosyncrasies of the software to reconfirm the correctness of the product's engineering and programming assumptions, they often find subtle errors. They gain detailed insight into the development process and an ability to spot critical errors early.

Finally, the number of discrepancies detected in software and the improvement in documentation quality resulting from error correction suggest that V&V costs are offset by the resulting more reliable and maintainable software. In many application areas, the costs of V&V are offset by either increased reliability during operation or by reduced maintenance when additions or changes are made. Many companies rely on their software systems for their daily operations. For them the costs of system or data loss outweigh the costs of preventing the losses. ❖

References

1. B.W. Boehm, "A Spiral Model of Software Development and Enhancement," *Computer*, May 1988, pp. 61-72.
2. B.W. Boehm and P.N. Papaccio, "Understanding and Controlling Software Costs," *IEEE Trans. Software Eng.*, Oct. 1988, pp. 1462-1467.
3. P.B. Powell, "Software Validation, Verification, and Testing Technique and Tool Reference Guide," NBS Special Pub. 500-93, Nat'l Inst. Standards and Technology, Gaithersburg, Md., 1982.
4. D.R. Wallace and R.U. Fujii, "Software Verification and Validation: Its Role in Computer Assurance and Its Relationship with Software Project-Management Standards," NIST Special Pub., Nat'l Inst. Standards and Technology, Gaithersburg, Md., to be published, 1989.
5. F. McGarry and G. Page, "Performance Evaluation of an Independent Software Verification and Integration Process," Tech. Report SEL 81-110, NASA Goddard Space Flight Center, Greenbelt, Md., Sept. 1982.
6. J.W. Radatz, "Analysis of IV&V Data," Tech. Report RADC-TR-81-145, Rome Air Development Center, Griffiss AFB, N.Y., June 1981.

Dolores R. Wallace is project leader for software quality and safety at the National Computer Systems Laboratory at the US National Institute of Standards and Technology (formerly the National Bureau of Standards). Her research interests include software V&V, acceptance, testing, management, quality assurance, and safety.

Wallace received an MA in mathematics from Case Western Reserve University. She is secretary for ANSI/IEEE Std 1012-1986, *Standard for Software Verification and Validation Plans*, and co-chairman of the IEEE P1059 working group to develop a guide for V&V plans. She is a lecturer for the IEEE standards seminar on software V&V. She is a member of ACM and the IEEE Computer Society.

Roger U. Fujii is the manager of the Systems Technology Operation at Logicon in San Pedro, Calif. His organization performs V&V on critical US Army and Air Force systems. He has also led the software nuclear-safety analysis on several missile and weapons systems. His research interests include software V&V, simulation, artificial intelligence, and expert systems.

Fujii received a BS in engineering mathematics and an MS in electrical engineering and computer science from the University of California at Berkeley. He is chairman of the US JTC1/SC7 Technical Activities Group and had been chairman of ANSI/IEEE Std 1012-1986, *Standard for Software Verification and Validation Plans*. Fujii is the principal instructor for the IEEE standards seminar for software V&V. He is a member of the IEEE Computer Society and ACM.

Address questions about this article to Wallace at National Institute for Standards and Technology, Technology Bldg., Rm. B-266, Gaithersburg, MD 20899.