Independent verification and validation

Ron Krauskopf
Frank Rash

Keeping software systems on the straight and narrow path to success

Mistakes caught early are cheaper and easier to correct. With this maxim in mind, third party checking is on the rise. V&V (verification and validation) is defined by Air Force Regulation 800-14 as "the process of determining that the computer program was developed in accordance with the stated specification and satisfactorily performs in the mission environment, the function(s) for which it was designed." Hence, IV&V (independent verification and validation) is the confirmation of a system by a group, agency, or company other than the original developer.

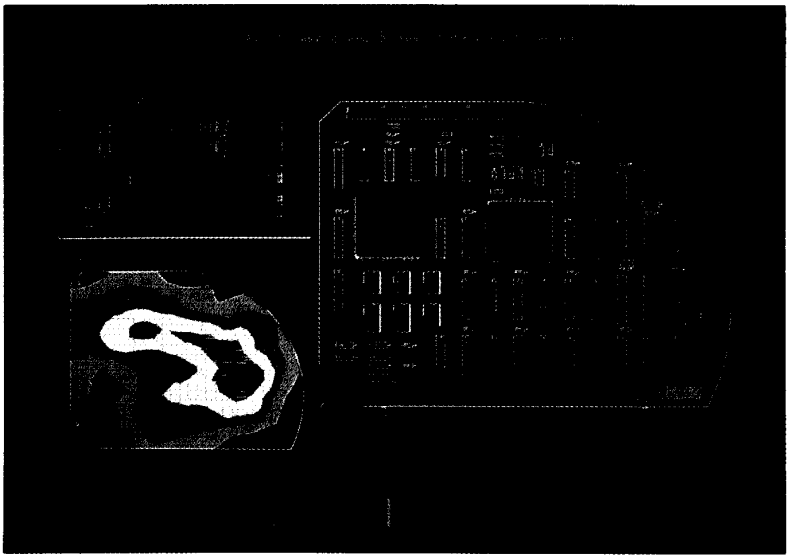
IV&V has three characteristics. They are applied over the entire software lifecycle. Pre-test activities include requirements verification, design walk-throughs, design quality measurements, interface control definition, code inspections, and formal design reviews.

Next, testing is a rigorous activity involving formal plans and procedures. Automated test tools are incorporated to reduce errors in repetitive tests and to increase the amount (paths) of testing.

Third, formal testing is performed by an independent test organization. This organization reports directly to the customer. The customer deals with the developer and with the independent test organization. While this move may seem to add a level of complexity, the net savings are usually worth it.

IV&V is not needed for all projects; just the five types of systems listed below:

- Realtime critical software (and systems) that must work correctly the first time and every time.
- Programs having critical outputs which cannot be verified with every run.
- Programs having a high cost of failure in human life, national security, or dollars.
- Software for which the cost of error detection through operational use exceeds the cost of IV&V.
- Software for which the cost of maintenance and modification exceeds that of IV&V.



Mentor Graphics Corp.

Beginning with the requirements phase, the IV&V team works closely with the customer and with the developers. An ideal relationship among the parties is depicted in figure 1. Continuing through the design, coding, and testing/validation phases, the IV&V team plays an active role in the creation of a system.

Requirements

The requirements phase formally specifies a system and translates the system into hardware, software and human subsystems. IV&V makes sure that the requirements fully meet the original intent of the system, and that the requirements are appropriately partitioned into software components. The objective scrutiny of IV&V leads to a more robust and mature software specification. IV&V concentrates its efforts in the following areas:

Completeness. Are the requirements complete? Is everything specified that is necessary so the original intent can be met?

Quantifiable. Are the performance characteristics simply stated and measurable?

Logical. Are the requirements reasonable? As high level system requirements are translated to lower level components, do they correctly maintain the original intent? Can the requirements be met given current technologies?

Consistent. IV&V ensures that differences among individual viewpoints do not embed themselves as contradictions in the system specification.

Testable. Results of future tests must be evaluated against the requirements stated in this phase of development. Therefore, the requirements must specify observable functionality in the system.

Understandable. Are the requirements meaningful? Are they comprehensible? Do they meet the original intent of the system?

0278-6648/90/0004-0012$01.00 © 1990 IEEE

Traceable. IV&V ensures that no requirements are lost or created unwillingly as the developers decompose the systems from the high level system requirements to the lower level component requirements.

Identify interfaces. The developer must correctly understand how the system fits into the operational environment and reflect this understanding in the system specification. Additionally, the interfaces between components of the system must be consistent and fully specified.

Design

The design phase in the software development lifecycle generally gets broken into two parts: preliminary and detailed design. This provides an orderly expansion of detail from the requirements into the software components. The IV&V troop participates in the design walkthroughs and ensures that the intent of the requirements are completely and correctly captured in the design. IV&V provides an alternative analysis of the developer's allocations of requirements to design.

Traceability. Again, the IV&V team verifies that the requirements are adequately captured and trace back to the customer's original requirements. No requirements have been deleted, and no "extra" requirements are included.

Equation/algorithm analysis. The IV&V team should analyze the developer's equations and verify that these equations do indeed address the requirements of the system. Additionally, the team needs to provide alternatives to the algorithms utilized in the design. The team needs to be cognizant of the target hardware timing and sizing budgets for the system. Modeling the equations and algorithms, and simulating the run time environment are useful when performing this analysis.

Data base definition. All the necessary data must be captured in the data base design. In large systems where data is produced and utilized by various subsystems, the IV&V team should confirm that the data is complete and satisfies the needs of all the design components. For example, a typical problem that arises is when the same data set is defined and created multiple times by different groups working on different portions of the system. IV&V can catch this occurrence.

Compatibility. The design of systems, particularly larger systems, requires that many groups of developers design various portions of the system. IV&V provides a broad, system level analysis of the design paying particular attention to the interface and compatibility aspects.

Interfaces. IV&V verifies that the developer's interface design provides access for each of the design components to the necessary data and functionality. The IV&V team again provides an overview approach of the entire system in doing this, thus, verifying that no interfaces are omitted and that all designed interfaces are necessary.

Independent allocation. A popular approach for IV&V is to independently allocate the requirements which were produced in the requirements analysis portion of the development. The IV&V team thus provides an alternative to the developer's allocation of requirements to the design. If the two allocations are extremely different, an investigation is warranted.

Timing/sizing budgets. IV&V establishes timing and sizing budgets for against the functional threads in the system and compared with the performance requirements of the system. These budgets must be maintained to reflect the current performance characteristics of the system. Thus, early detection of failure in meeting performance requirements can be accomplished. Also, areas which approach the design. These budgets are analyzed the timing or sizing limits of the system can be pin-pointed.

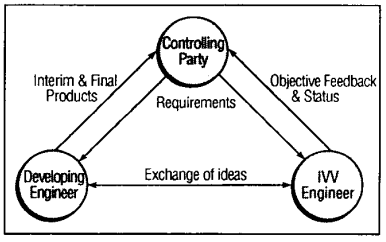


Fig. 1. Ideal IVV relationship.

Test procedures. During design, the IV&V team begins monitoring the developer's development of test plans and procedures. The team also develops their own independent test plans and procedures. The independent plans and procedures should not duplicate the developer's efforts. Instead, they should concentrate on stressing the system and finding its limitations.

Code

As the developers transcribe the software design into code, the IV&V team is there. They ensure that the code adheres to the standards and procedures established and that sound programming practices are being followed. Generally, automated tools are used for this phase. Code analysis tools, automated metrics capturing tools, and reverse engineering tools are a few which are available.

Traceability. The requirements defined for the system and captured in the design of the software must now be implemented in the code. The IV&V team must report any instances where the code does not adequately address the design or where the code provides auxiliary functionality not called for.

Standards adherence. Generally, the software developed for a system must be developed to a defined standard. Maximum lines of code per module guidelines, comment to line of code ratios, capitalization and indentation guidelines, and if/then/else nesting levels are typical standards which must be followed. IV&V reports where standards are not followed.

Efficiency. IV&V addresses efficiency from both a static and dynamic analysis of the code. For instance, assignment statements within a loop should be moved outside of the loop whenever possible. Functionality which is repeated many times should possibly be captured in a subroutine. In general, space versus time tradeoff issues are addressed by the IV&V team. Input from this analysis is then provided to the code developers.

Maintainability. The customer wants the code to be as inexpensive to maintain as possible. IV&V ensures that the structure of the code is smartly laid out (e.g., modular, clean interfaces). Additionally, the code should be well documented.

Unit/module test. The IV&V team should monitor the developer's unit and module tests. The testing of the most critical units and modules should be monitored when the size of a procurement precludes the team from monitoring all testing. All code is executed and a sufficient percentage of the paths are executed to reasonably assure no side effects are present.

Test/validation

The ultimate responsibility of the IV&V team is to validate the software. Throughout the entire software development process, the IV&V troop has been monitoring and providing alternatives to the developers so that the software satisfies the original intent. The final activity is to prove that the software meets this intent.

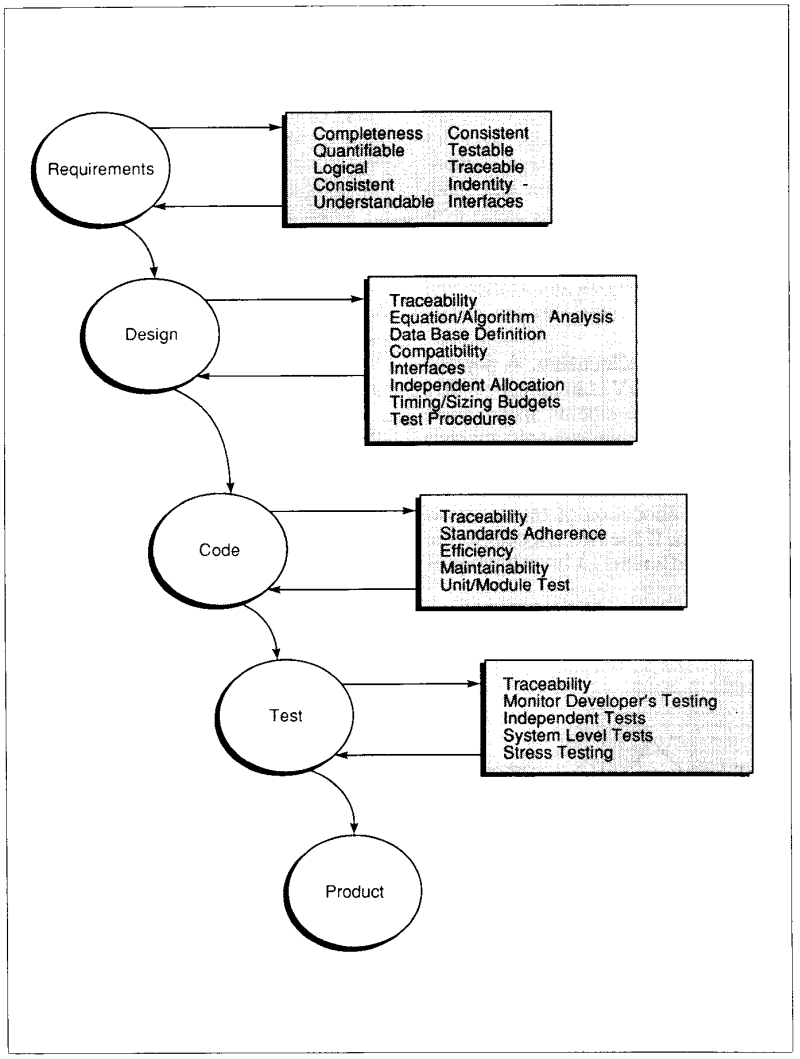


Fig. 2. IVV responsibilities.

Traceability. For the last time, IV&V must validate that the original requirements levied on the system by the customer have been fully satisfied and no additional functionality has crept into the system. The requirements must be allocated to test plans and procedures for independent testing. The successful execution of these tests confirms this fact.

Monitoring developer's testing. IV&V should monitor the developer's testing which is usually focused on proving that the requirements are satisfied. The IV&V team concentrates on areas they feel are weak in the developer's testing and perform additional tests.

Independent tests. IV&V tests should focus on critical areas in the system. Criticality is defined based on the customer's concerns and might include areas where failure could lead to: threat to human life, threat to completion of a mission, and so forth. Again, IV&V testing should elaborate on the developer's testing and provide additional assurance that the software performs the required functionality in the given time constraints.

System level tests. IV&V tests are generally system level tests. They execute threads that begin with a system input (e.g., an operator pressing a button) and end with a system output (e.g., a message displayed on the screen). The system's requirements are mapped to these threads and, thus, the successful execution of a system level test validates that the requirement is met.

Stress testing. Whereas the developer's testing focussed on proving that the requirements are satisfied, the IV&V should test to find the limitations of the software. Tests should be designed to bombard the software with inputs. For example, at what point does the amount of data introduced into the system push the performance below an acceptable rate? Again, the IV&V team is not duplicating the developer's efforts but providing additional information concerning the performance of the system.

Summary

Independent verification and validation is becoming more and more prevalent both at a customer level, and within companies. The personnel required to successfully perform IV&V on a software procurement must themselves be capable of developing that software. The personnel must be knowledgeable and have experience in developing similar systems and environments they are going to verify and validate. Thus, educated assessments and alternative approaches can be provided which eventually lead to a better system.

Read more about it

- Dobson, J., and Randell, B., "Program verification: public image and private reality," *Communications of the ACM*, 32:420–2, Apr. 1989.
- Fetzer, J.H., "Program verification: the very idea," *Communications of the ACM*, 32:506–12, Apr. 1989.
- Irland, E.A., "Assuring quality and reliability of complex electronic systems: hardware and software," *Proceedings of the IEEE*, 76:5–18, January 1988.
- Musa, J.D., "Tools for measuring software reliability," *IEEE Spectrum*, 26:39–42, February, 1989.

About the authors

Ron Krauskopf is a Software Engineer for Dynamics Research Corporation. He is currently involved in the IV&V effort for the Joint STARS project. He is pursuing his Ph.D. in Computer Science from Florida Institute of Technology. His interests include artificial intelligence, software engineering methodologies, real-time distributed computer systems, and IV&V.

Frank Rash is a Software Engineer for Infotec Development, Inc. He is currently involved in the IV&V effort for the Joint STARS (Surveillance Target Attack Radar System). Frank received his BS in Computer Information Systems from the College of Engineering at Ohio State University. He is currently pursuing a Masters in Operations Research from Florida Institute of Technology. □