*Proceedings of the 2000 Winter Simulation Conference*
*J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, eds.*

# VERIFICATION AND VALIDATION WITHOUT INDEPENDENCE: A RECIPE FOR FAILURE

James D. Arthur
Richard E. Nance

Systems Research Center and
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061, U.S.A.

## ABSTRACT

Verification and validation (V&V) is a prominent technical area within simulation, attested to by the 12 tutorial sessions (including five advanced) included in the past ten Winter Simulation Conferences (WSCs). In recent WSCs the issue of *Independent* V&V (IV&V) has drawn increased attention, with sessions examining the perceived lack of use and little concern for the technique evinced within the simulation community. The objectives of this paper are four-fold: (1) to examine the current picture in software systems development, (2) to review the rationale, role and expressed need for IV&V, (3) to identify the benefits attendant in the insistence on the "independent" status of the activity, and (4) to respond to the usual criticisms of negative impacts on cost and schedule. While the treatment is couched in the more general software systems context, we contend that simulations represent prime candidates for IV&V application.

## 1 BACKGROUND

In 1972 Fritz Bauer called for drastic improvements in the way software was created, noting the need for research in concepts and techniques in the production of software objects. He characterized the desired approach as "software engineering" (Bauer 1972). Software engineering has now been in existence for 30 years, and has been the subject of more than 25 years of research. Fifteen years ago the Department of Defense created the Software Engineering Institute (SEI) and charged it with the specific task of identifying sound software engineering principles and practices that would lead to the development of a quality product, on time and within budget. The Capability Maturity Models (CMM) and the CMM Assessment Procedures are prominent products of the SEI effort.

Independent verification and validation (IV&V) is a technique of long-standing in software development. Often criticized for imposing excessive cost on a software development process, the temptation is to cast IV&V as unnecessary and wasteful in the much-improved software practices of today. So, with the vastly improved technology of today, how successful are software development organizations in producing quality products on time and within budget*?* Of even greater significance is an answer to the question: *Is IV&V an anachronism that should be eliminated as a risk-mitigating strategy?* The remainder of this report, addressing the objectives cited above, provides some thought-provoking facts supporting a clearly negative answer to that question.

### 1.1 A Statistical Profile of Software-Intensive Projects

Despite the advances in software engineering and the existence of SEI-CMM-driven improvements for both the software and systems engineering domains, *project success is still less likely than failure* (Standish 1995). More specifically, the U.S. spends more than $250 billion annually on information technology applications development of approximately 175,000 projects (Standish 1995, p. 1). Research shows that 31% of those projects are canceled before completion and that 53% incur cost exceeding 189% of their original estimates (Standish 1995, p. 1). The Standish Group estimates that in 1995 American companies and government agencies spent $81 billion for canceled software projects, and that the same organizations paid an additional $59 billion for completed projects that exceed their original time estimates (Standish 1995, p. 2).

The same study reports that in 1995 only 16% of software projects were finished on time and within budget. In large companies, however, the news is even worse: *only nine percent* of their projects were completed on time and within budget. And, among the completed projects, many of the original specification requirements are missing: on the average only 42% of the original features and functions were present. Smaller companies tended to do much

better, completing 78% of their software projects with at least 74% of originally specified features and functions (Standish 1995, p.2).

Kasser and Williams (1998, Part 1, p. 1) report similar failure statistics characterized by project size. In particular they cite a 1995 study by Cuppan (1995) which reports that approximately 80% of large software projects executed within the DOD were 100% over budget and that 90% were at least one year behind schedule. Even more disturbing is the opinion stated by 48% of the IT executives interviewed that *the level of current failures exceeds that of just five years ago*.

The above statistics do not support a claim for improvement in the capability of software development organizations to deliver a quality product on time and within budget. Even with the staffing levels and expertise that the (larger) companies are able to apply, project failures far exceed project successes. While CMM compliance is a necessary step in the right direction, it is not, in and of itself, sufficient. Lewis states that even when working with organizations at CMM levels 4 and 5 one can expect to find "deficiencies in requirements and design, poorly expressed algorithms, errors in code, and testing shortcomings" (Lewis 1992, p. 299). To help mitigate the risks of failure, he and we advocate the use of an additional powerful tool: *independent* verification and validation (IV&V).

### 1.2 Sources Used

To help ensure the assimilation of a comprehensive picture framing the current status of how effective the software engineering community and development organizations are in managing software-intensive projects, we draw from (and are citing) a wide variety of references. The citations fall into three categories: (1) studies that present objective statistics describing project successes and failures, reinforced by solicited opinions of IT professionals, (2) standards promulgated by a software engineering professional body, and (3) research that compares the impact of IV&V with the conventional use of verification and validation within software quality assurance (SQA).

Reports by the Standish Group and the DOD publication "Software Tech News" are representatives of first category. The IEEE Standard for Software Verification and Validation comprises the second category. The third category is composed of reports from technical journals, such as *IEEE Computer,* and from books citing experiences and lessons learned.

## 2 INDEPENDENT VERIFICATION AND VALIDATION: RATIONALE AND ROLE

Within the software engineering process activities, *verification* is defined as an iterative process aimed at determining whether the product of each step in the development cycle:

- fulfills the requirements levied on it by previous steps, and
- is internally complete, consistent, and sufficiently correct to support the next phase.

*Validation*, on the other hand, is defined as the process of executing the software to exercise the hardware and comparing the test results to the specification requirements (Lewis 1993, p. 7).

*Independent* Verification and Validation is defined as a series of technical and management activities *performed by someone other than the developer of a system* with the objectives of:

- improving the quality of that system, and
- assuming that the delivered product satisfies the user's operational needs.

Similarly, the IEEE Standard for Verification and Validation states that in the classical approach to IV&V "the IV&V responsibility is vested in an organization that is separate from the development organization" (IEEE 1998, p. 58).

### 2.1 Factors Motivating the Need for IV&V

In an extensively cited paper Barry Boehm states that "verification and validation activities produce their best results when performed by a V&V agent who operates independently of the developer or specification agent" (Boehm 1984, p. 76). The advantages of an independent V&V process are many. In particular, the *independence* in V&V:

- provides an objective assessment of the product during its creation,
- adds a new analytical perspective not present in the development environment,
- brings its own set of tools and techniques to bear on ensuring development accuracy and validity,
- introduces "intermediate" users of the system who serve as "beta testers" before the product goes to market, and finally
- significantly enhances testing and the discovery of design flaws and coding errors.

#### 2.1.1 When is *Independent* V&V Needed?

The IEEE Standards for Software Verification and Validation states that classical IV&V is generally required for the development of software systems deemed "critical" nature, i.e., those which can result in loss of life, loss of

mission or significant social or financial loss (IEEE 1998, p. 58).

Mirroring a similar opinion, Lewis (1992, p. 16) states that IV&V should be required for:

- real-time critical software that must work every time,
- programs having a high cost of failure in terms of human life, national security, or money,
- software for which the cost of error detection through operational use exceeds the cost of IV&V, and
- software for which the cost of maintenance and modifications exceeds the costs of IV&V.

### 2.1.2 Why is *Independent* V&V Needed?

Independent V&V is necessary to establish technical, managerial and financial independence (IEEE 1998, p. 57). The following three paragraphs are taken directly from the IEEE Standards for Software Verification and Validation.

*Technical independence* requires the V&V effort to utilize personnel who are not involved in the development of the software. The IV&V effort must formulate its own understanding of the problem and how the proposed system is solving the problem. Technical independence is an important method to detect subtle errors overlooked by those too close to the solution.

*Managerial independence* requires that the responsibility for the IV&V effort be vested in an organization separate from the development and program management organizations. Managerial independence also means that the IV&V effort independently selects the segments of the software and system to analyze and test, chooses the IV&V technique, defines the schedule of IV&V activities, and selects the specific technical issues and problems to act upon. The IV&V effort must be allowed to submit to program management the IV&V results, anomalies, and findings without any restrictions or adverse pressures, direct or indirect, from the development group.

*Financial independence* requires that control of the IV&V budget be vested in an organization independent of the development organization. This independence prevents situations where the IV&V effort cannot complete its analysis or test within the delivery time because funds have been diverted or adverse financial pressures or influences have been exerted.

While the IEEE standard recognizes several forms of "independent" V&V, it also states that only the classical IV&V form embodies all three of the above independence parameters. More specifically, the three forms of independence can only be achieved if the V&V organization is separate and distinct from the software development and project management organizations.

The importance of financial and managerial independence is underscored in the analogical questions raised by Lewis: *"Should a meat inspector work for the packing house?" "Should an auditor work for the company being audited?" "Should a bank examiner work for the bank?"*

### 2.1.3 What Are the Benefits of *Independent* V&V?

The benefits of IV&V are many, and are a direct consequence of maintaining technical, managerial and financial independence. The benefits outlined below are enumerated in two studies that specifically compare the projects developed under the auspices of IV&V to those developed with no IV&V component (Arthur et al. 1999; Radatz 1981). Wallace and Fujii (1989) and Lewis (1992) support the findings by Arthur et al. and Radatz and also add to the characterization of those benefits.

*Benefit 1:* Reflecting the rationale for maintaining technical independence, *IV&V promotes objectivity*, that is, it helps maintain an unbiased technical viewpoint and supports an objective engineering analysis. This objectivity (a) encourages one to consider a wider range of solutions (Arthur et al. 1999, p. 79), (b) aids in the detection of subtle errors often overlooked by those too close to the solution (Wallace and Fujii 1989, p. 14; IEEE 1998, p. 57), and (c) supports an unbiased critique of how well the software performs (Wallace and Fujii 1989, p. 14).

*Benefit 2: IV&V promotes the earlier detection of software and system errors* (Arthur et al. 1999, p. 80; Wallace and Fujii 1989, p. 14). Arthur et al. report errors being detected a full phase earlier in the software development life cycle. Radatz, on the other hand, reports that 50%-89% of the errors were detected before development testing.

*Benefit 3:* Earlier error detection translates into *reduced effort and cost in removing those errors*. In their study, Arthur et al. report that the effort to remove errors is cut by one-half (Arthur et al. 1999, p. 82).

Lewis states that "the greatest cost-benefit ratio in IV&V comes from requirements verification, wherein defects in requirements can be caught before they begin to ripple forward" (Lewis 1992, p. 67). Similarly, Arthur et al. found that IV&V supports the detection of ambiguous and unclear statements in requirements and design documents, and hence, removes the potential introduction of additional errors later in the development effort (Arthur et al. 1999, p. 81).

*Benefit 4: Enhanced operational correctness* is another benefit reported by Arthur et al. (1999, p. 82).

*Benefit 5:* An unexpected but plausible benefit reported by Arthur et al. is the statistically significant *reduced variability in the development process* (Arthur et al. 1999, p. 83). From a software development perspective, the implication of such a finding is that IV&V encourages a more controlled software development process.

The points and observations enumerated in Section 2.1 (and its subsections) affirm the contention that *independent* V&V plays a special role in the software development process. That is, IV&V provides a powerful approach to mitigating risk, and is of particular importance when that risk applies to the development of life-critical and/or extremely costly systems.

### 2.2 What Does IV&V Provide Beyond SQA?

"The chief role of SQA is that of an internal watchdog" (Lewis 1992, p. 282). What is crucial in this statement is the qualifier "internal". Consequently, SQA is (rightfully) viewed as part of the organization that is also providing the software product or the development services. Both SQA and the development organization report (at some point) to the same higher-level management position. This reality creates a natural and significant tension between conflicting objectives: producing software exhibiting the highest quality and delivering a product on time and within budget. The development organization's objective of realizing maximum profit compromises concerns for quality. As such, decisions are often made that adversely impact the quality of the product being developed. *More pointedly, the resolution of internal political issues often overrides technical concerns about the quality of the product or process.*

The Standish Group report cites several representative statements by IT professionals attesting to the above indictment: "Probably 90% of applications project failures is due to politics!" "Sometimes you have to make decisions you don't like. Even against your own nature. You say well, it's wrong but you make the decision anyway." In one of its case studies, the group also makes the observation that "Because of internal state politics, unclear objectives, and poor planning, the (California DMV) project was doomed from the start" (Standish 1995, p. 6). A similar criticism is based on a survey of systems and software personnel. The report shows that political considerations outweighing technical factors places sixth in a priority list of risk factors affecting project success (Kasser and Williams 1998, Part 3, p. 2).

In bottom-line parlance, while SQA might be the "watchdog", because the SQA group is part of the development organization, it has very little influence in advancing the consideration of quality when confronted by pressures of schedule and cost. *This untenable situation is exacerbated by the fact that in neither the ISO 9001 Standard nor the Software-CMM (at any of the five levels) is the issue of political concerns overriding sound technical decisions addressed.*

*Independent* V&V, on the other hand, is ideally positioned to identify and ensure that the negative impact of political concerns are minimal. This claim is derived from recognition that the IV&V agent (a) operates independently of the development organization, and (b) owes allegiance only to the customer and not to the development organization. These characteristics to a large degree insulate the IV&V agent from the whims of politics, and thereby, force greater accountability on the development organization and increased interaction with the customer.

## 3 THE EFFECT OF IV&V ON SOFTWARE DEVELOPMENT COSTS

Lewis estimates that an IV&V effort starting at the requirements phase and continuing through deployment would increase the development costs approximately 10 to 18 percent. He also states and justifies that the larger the project, the lower the percentage – that is, as total project costs increase the percentage is reduced to close to ten (Lewis 1992, pp. 270, 280).

The study by Radatz (as reported by Wallace and Fujii (1989, p. 4)), however, indicates that a significant portion of the costs of IV&V can be recouped. When examining development efforts that initiate IV&V at the beginning of the coding phase, he estimates that 20%-28% of the cost of IV&V is saved. When examining development efforts that initiate IV&V at the beginning of the requirements phase, he reports a savings of 92%-180% of the costs of IV&V! In effect, these statistics imply that the estimated costs of a software development effort containing no IV&V component is comparable to (if not more than) the "same" development effort containing an IV&V component. Furthermore, because of the IV&V component, a higher quality, more robust product is a reasonable expectation.

Several factors point to the validity of the above observations and conclusions. First, as reported by Lewis "… the cost (of IV&V) is offset by a reduction in reported problems and latent user errors and much higher level of user satisfaction…" (Lewis 1992, p. xxiii). Secondly, in a survey of systems and software personnel the following risk factors are ranked among the highest in contributing to (among other things) costs (Kasser and Williams 1998, Part 2, p. 1). From high to low they are:

- poor requirements,
- lack of, or, poor plans,
- failure to validate original specifications and requirements, and
- failure to communicate with the customer.

Independent V&V addresses and has a positive influence on each of the above.

## 4 THE IMPACT OF IV&V ON PROJECT SCHEDULE

The Standish Group survey includes 365 IT professionals who report on 8,380 projects. Over 7000 of those projects

(84%) are either cancelled or exceeded budget and/or time estimates. The latter projects (termed "impaired") typically delivered less functionality than originally specified. Of the 7000+ impaired projects:

- 11% exceeded their time estimates by more than 200%,
- 46% exceeded their time estimates by more than 100%, and
- 66% exceeded their time estimates by more than 50% (Standish 1995, pp. 2-4).

Both the Standish Group survey and the Kasser and Williams report indicate "poor requirements" as clearly the strongest indicator ofthese project failures (Standish 1995, p. 5, Kasser and Williams 1998, Part 3, p. 1). As indicated by *all* of the references cited in this report, a major activity of IV&V focuses on producing well-defined requirements, i.e., requirements that are complete, non-ambiguous, consistent, testable and traceable. Because of its independence, the IV&V activity is unfettered by the events occurring during the generation of requirements. Moreover, because it has no responsibility to defend the generation process, the IV&V agent can bring a fresh perspective and more detached view that allows the agent to be objectively critical during requirements analysis.

The Standish Group survey and the Kasser and Williams report also identify "failure to communicate with the customer" and "poor allocation of resources" as the second and third most prominent risk factors, both of which have a detrimental impact on project scheduling (Standish 1995, p. 5, Kasser and Williams 1998, Part 3, p. 1). Clearly, *independent* V&V has a positive influence on each of these. First, IV&V generates periodic reports citing progress as well as anomalies. Such reports provide the impetus and basis for additional communication between the developer and customer. Secondly, one of the IV&V defined tasks is to examine the staffing, schedule and funding profiles to establish their feasibility and conformance. Again, noted inadequacies or discrepancies are brought to the attention of both the developer and customer.

Finally, both the survey and report identify "unrealistic deadlines" as another prominent risk factor leading to schedule slips (Standish 1995, p. 5, Kasser and Williams 1998, Part 2, pp. 1, 3). In this instance, the need for IV&V to assist in mitigating this risk factor is even more crucial because neither the ISO 9001 Standards nor the Software-CMM (at any level) have any provisions relating to it (Kasser and Williams 1998, Part 3, p. 1).

## 5 IV&V AND PRODUCT QUALITY: EXAMINING THE LIFE-CYCLE IMPLICATIONS

True project success must be examined from a life-cycle perspective, and in particular, how well the product meets the needs of the customer. Once delivered, the more prominent concerns for the customer are that the product (a) provides the required functionality and (b) embodies those necessary characteristics that facilitate post-deployment sustainment (maintenance).

Customer involvement is crucial to achieving the required functionality. In their survey of IT executives and technical managers, the Standish Group lists ten factors supporting project success; at the top of the list is user involvement (Standish 1995, p. 4). Similar to the project scheduling concerns noted in the previous section, IV&V is charged with specific tasks that focus on identifying the presence and absence of required functionality, e.g., requirements verification and product validation. In performing those activities the *independence* of V&V ensures that no political issues or concerns internal to the development organization can hinder the recognition of inadequate or missing functionality.

From a maintenance perspective, one of the more crucial support items is documentation. In particular, to facilitate effective post-deployment maintenance the delivered documentation must reflect the actual product design, implementation and operational profile. In his study comparing two development efforts, both having a V&V component but only one being independent, Arthur et al. note the substantial difference in the quality of documentation produced (Arthur et al. 1999, p. 82). That difference is attributed to the constant oversight of the IV&V group and the group's insistence that noted anomalies be addressed and resolved.

One final life-cycle implication is the cost of a product over its lifetime. An accepted fact (supported by numerous studies) is that the post-deployment maintenance accounts for approximately 60%-80% of the true life-cycle cost of a product. In Section 3 of this report we note that the cost of IV&V is approximately 10%-18% of the development costs. Being conservative in our computation (accepting the lower percentage maintenance costs and higher percentage IV&V costs) the *lifecycle* costs of IV&V is computed to be only 7.2% of the total life-cycle costs of the product. This estimate does not include the additional savings due to maintaining a better quality product.

## 6 CONCLUSIONS

The IEEE Standards for Software Verification and Validation provides a refined definition for Independent Verification and Validation. It states that Independent V&V is defined by three parameters: Technical Independence, Managerial Independence and Financial Independence. The standard also states that to achieve all three of these highly desirable aspects of independence, one must employ classical IV&V, that is, when the IV&V responsibility is vested in an organization *separate and distinct* from that of the development organization.

The benefits derived from employing Independent V&V are substantial. As outlined in this report those benefits span the product life cycle, starting with improved requirements specifications and ending with a complete, more maintainable product that meets the user's needs. In effect, IV&V can be viewed as an effective risk mitigation strategy that significantly increases the probability of producing a quality product, on time and within budget.

The Standish Group report and the Kasser and Williams survey reveal the continuing disarray in software-intensive projects and lack of success by software development organizations. They also underscore the point that significant software-intensive efforts undertaken by large contractors are the most susceptible to project failure. The reports imply that while compliance with the ISO 9001 Standards and the Software-CMM is highly desirable, it is not, in and of itself, sufficient. The survey and report point out that political concerns or internal organizational objectives override sound technical decisions all too often.

The IEEE Standards for Verification and Validation as well as the studies by Arthur et al., Boehm, Lewis, Radatz, and Wallace and Fujii emphasize the need for *Independent* V&V as a cost-effective way to mitigate the many risks inherent in large software-intensive efforts. The common theme that is pervasive among those studies is that IV&V brings objectivity and oversight, and focuses on effectively resolving those technical issues that can cause project failure or which might compromise the stated needs of the customer.

We return to the question initially posited: *Is IV&V an anachronism that should be eliminated as a risk-mitigating strategy?* We find sufficient and compelling evidence to support an emphatically negative answer.

## ACKNOWLEDGMENT

The research leading to the development of this paper has been funded by the Naval Surface Warfare Center Dahlgren Division, Dahlgren, Virginia.

## REFERENCES

Arthur, J.D., M.K. Groener, K.J. Hayhurst, and C.M. Holoway 1999. Evaluating the Effectiveness of Independent Verification and Validation, *IEEE Computer* 32(10): 79-83.

Bauer, F.L. 1972. Software Engineering, *Information Processing* 71: 530.

Boehm, B.W. 1984. Verifying and Validating Software Requirements and Design Specifications, *IEEE Software*, January: 75-88.

Cuppan, C.D. 1995. Capability Maturity Model (CMM) Characteristics and Benefits, Tutorial Presentation and the Defense Mapping Agency, June 1995.

IEEE 1998. *IEEE Standards for Software Verification and Validation*, IEEE Standard 1012-1998.

Kasser, J. and V. Williams 1998. What Do You Mean You Can't Tell Me If My Project Is In Trouble? *Software Tech News,* 2(2) <www.dacs.dtic.mil/awareness/newsletters/technews2-2/trouble>.

Lewis, R.O. 1992. *Independent Verification & Validation: A Life Cycle Engineering Process for Quality Software*, Wiley Series in New Dimensions in Engineering, ed. R. D. Stewart. New York: John Wiley & Sons.

Radatz, J.W. 1981. Analysis of IV&V Data, Technical Report RADC-TR-81-145, Rome Air Development Center, Griffiss AFB, NY, June 1981.

Standish Group 1995. Chaos, Standish Research Paper, <www.standishgroup.com/chaos.html>.

Wallace, D.R. and R.U. Fujii 1989. Software Verification and Validation: An Overview, *IEEE Computer* 6(3): 10-17.

## AUTHOR BIOGRAPHIES

**JAMES D. ARTHUR** is an Associate Professor of Computer Science at Virginia Tech. He received B.S. and M.A. degrees in Mathematics from the University of North Carolina at Greensboro in 1972 and 1973, and M.S. and Ph.D. degrees in Computer Science from Purdue University in 1981 and 1983. His research interests include Software Engineering (Methods and Methodologies supporting Software Quality Assessment and IV&V Processes), Parallel Computation, and User Support Environments. Dr. Arthur is the author of over 30 papers on software engineering, software quality assessment, IV&V, and user/machine interaction. He has served as: participating member of IEEE Working Group on Reference Models for V&V Methods; Chair of Education Panel for National Software Council Workshop; and Co-Guest Editor for *Annals of Software Engineering* special volume on Process and Product Quality Measurement. His e-mail and web addresses are <arthur@vt.edu> and <http://vtopus.cs.vt.edu/~arthur>.

**RICHARD E. NANCE** is the RADM John Adolphus Dahlgren Professor of Computer Science and the Director of the Systems Research Center at Virginia Tech. Dr. Nance is also Chairman of the Board of Orca Computer, Inc. He held a distinguished visiting honors professorship at the University of Central Florida for the spring semester, 1997. Dr. Nance has held research appointments at the Naval Surface Weapons Center and at the Imperial College of Science and Technology (UK). He has held a number of editorial positions and was the founding Editor-in-Chief of the *ACM Transactions on Modeling and Computer Simulation*, 1990-1995. Currently, he is a member of the

Editorial Board, Software Practitioner Series, Springer. He served as Program Chair for the 1990 Winter Simulation Conference. Dr. Nance received a Distinguished Service Award from the TIMS College on Simulation in 1987. In 1995 he was honored by an award for "Distinguished Service to SIGSIM and the Simulation Community" by the ACM Special Interest Group on Simulation. He was named an ACM Fellow in 1996. His e-mail and web addresses are <nance@vt.edu> and <www.cs.vt.edu/info/people/vitae/Nance.html>.