The Role of V&V in the Early Stages of the Lifecycle

By
Steven H. Dam, Ph.D., ESEP
SPEC Innovations

Abstract

Verification and Validation (V&V) occur in the later parts of the system lifecycle to ensure that the requirements developed in the early phases of the lifecycle have been met. This paper discusses how to use the techniques of V&V, including simulation, early in the lifecycle to enhance the probability of success of the program by identifying errors early in the development and preparing for the V&V activities later in the lifecycle.

www.specinnovations.com
steven.dam@specinnovations.com

1. Introduction

Having just watched the movie “Deepwater Horizon” [Deepwater] I observed the results when proper testing is skipped. If you have not seen the movie, several routine and critical tests were skipped and another test was performed with mixed results. The mixed results test had the result of people picking the optimistic conclusion, which turned out catastrophic.

We have seen these problems in many other instances. Just look at NASA Mishap Reports. [Larson et al, 2009] They contain example after example of failures all through the lifecycle, often due to skipped tests. Such defects can be identified earlier in the lifecycle if modern modeling and simulation techniques are used. Unfortunately, most systems engineering and architecture work is done using the equivalent of Microsoft Office, which is fantastic for word processing, spreadsheets, and drawing tools, but wholly insufficient for engineering modeling and simulation.

2. The Lifecycle Model

If you are a systems engineer, you are likely familiar with the “V” lifecycle model. [Forsberg, 1991] Our version of the V is shown in Figure 1 below. This diagram shows the phases of the lifecycle as we decompose the system to a level from concept to where we can specify and build the components of the system. Then it shows the integration of the components to build up the system, while verifying and validating that the system meets the requirements.

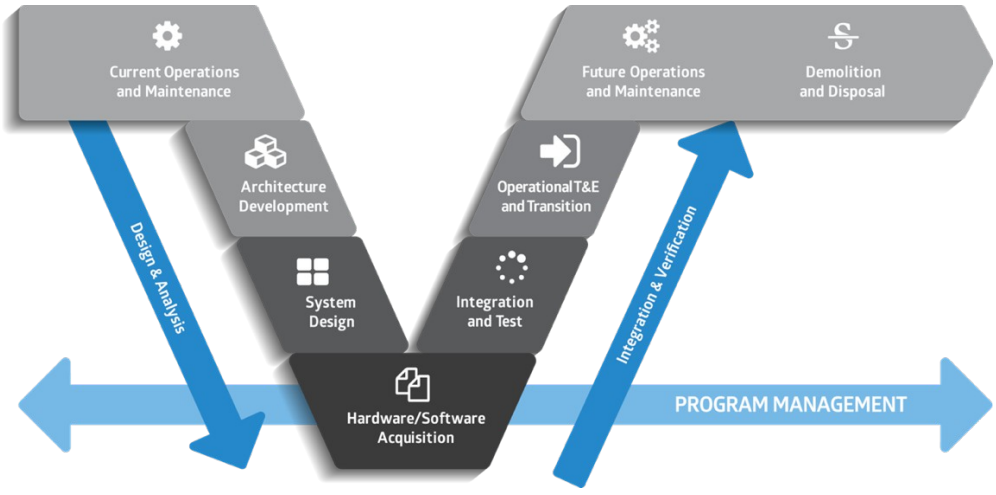

Figure 1. The “V” Lifecycle Model

The “V” shape reminds us of the parallel V&V activities that are affected as we decompose the system. For example, during the architecture development, it is important to derive the acceptance criteria as the basis for the operational test and evaluation (OT&E) and transition activities. A draft test and evaluation master plan (TEMP) can and should be produced during this architecture development phase. Unfortunately, this process often is not done at all or it is done in a way that renders it effectively useless.

In addition to this lack of test planning, often the architecture development takes the form of creating a series of drawings based on a limited language, such as Systems Modeling Language (SysML). The nine SysML drawings are necessary, but not sufficient, to capture all the information about the system, which includes the risks, decisions, costs, and other parameters needed to fully describe the system. The work also usually focuses on operations and ignores maintenance and various potential failure modes of the system. In addition to these problems, the drawings are usually just treated as static depictions of processes and components. Those drawings can contain significant errors in logic that do not become obvious until the equivalent phase coming up the “V.” As is well known, finding errors later in the lifecycle is very expensive and may even lead to program cancellation. [Albrecht, 2017]

3. Deriving Requirements from Scenarios

Often in the architecture development phase, requirements, particularly functional requirements, have not been developed to a sufficient level that we can derive the verification requirements needed to begin test planning. In view of this fact, scenario analysis is often used to develop models of the operations (and maintenance) of the system, from which the functional requirements for the system can be identified. The challenge in using scenario analysis is evolving a set of scenarios that will ensure that the breadth and depth of the functionality will be developed. A number of years ago, I developed an approach that applied the idea of a test matrix to the development of scenarios for architecture analysis. The basic strategy was to develop a list of scenarios and a set of characteristics of these scenarios [Scenarios, 2007]. These two pieces of information would then form a matrix, similar to a test matrix where you have the test scenarios juxtaposed with the parameters you want to measure. An example is shown in Figure 2. The example shown is a fairly simplistic example, but this can be expanded and used for much larger problems.

Scenario #	Number of Targets	Friendly/Enemy	Communicating/Not Communicating	Zone	Nature of Activity
1. Detect, Locate and Kill TEL	Single	Friendly	Communicating	Single	Track
2. Defend Against Missile Attack	Single	Friendly	Communicating	Many	Track
3. Defense and Kill TEL	Single	Unknown	Not Communicating	Many	Track
4. Find Threat Before Launch	Single	Friendly	Communicating	Single	Search and Rescue
5. Find and Destroy Threat Before Launch	Single	Friendly	Communicating	Single	Environmental Issue
6. Find and Destroy Multiple Threats	Many	Unknown	Not Communicating	Single	Track
7. Find and Destroy Threats Theater-wide	Many	Mix	Mix	Many	Track

Figure 2. An Example of a Scenario Matrix

Once we have identified these scenarios, then we can build models of each. Figure 3 shows an Action Diagram of the first scenario for the table above. [Dam]

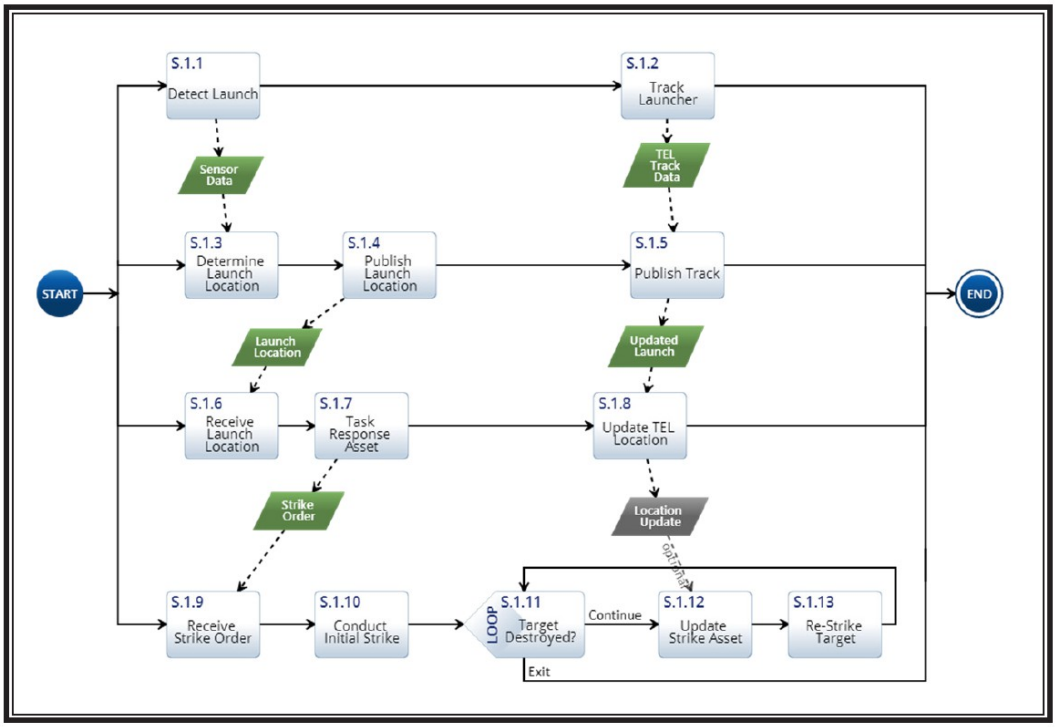

Figure 3. Example of Scenario 1 Action Diagram Model

These models can then be used to derive the functional requirements for the system. Several tools automate this process by reading the functional elements of the models and their relationships to physical entities, such as the system, and then produce requirements and/or documents that can be used as the basis for the system specification.

But, how do we know this model works? Could we be building errors in the logic in these diagrams? The answer to these questions comes from further analysis using simulation techniques.

4. Application of Simulation

We are very familiar with simulation as one of the methods for V&V, along with analysis, inspection, demonstration, and test. We can apply simulation in the early phases of the lifecycle by applying various simulation techniques to the models developed in the previous section. Discrete Event Simulation (DES) uses “a mathematical/logical model of a physical system that portrays state changes at precise points in simulated time” [Albrecht]. Unfortunately, many of the modeling tools do not have an embedded DES capability, so models developed using the drawing technique of choice often have to be redeveloped in the simulation tool. The question becomes “Is this the same model?” In general, the answer is probably no. Just as when we had a diagram to a programmer and receive code back, the simulation modeler will make changes that often are not documented to fix problems with the original model. Fortunately, several tools [Tools] have both the modeling capability and a full DES. An example of the execution of the model is shown below in Figure 4.

Recently, I have discovered that given the uncertainty in the timing of each step of the process, the probabilities of failure paths, the variability of resources, and many other factors, DES is insufficient to estimate the potential range of capabilities of the systems being modelled. However, if we apply another simulation technique in conjunction with DES, we can easily include these uncertainties in the modeling. This technique is called Monte Carlo simulation. Monte Carlo simulation, the name given by John van Neumann and Stanislaw M. Ulam to reflect its gambling similarity, utilizes models of uncertainty where representation of time is unnecessary. The term originally attributed to a situation in which a difficult non-probabilistic problem is solved through the invention of a stochastic process that satisfies the relations of

the deterministic problem. A more recent characterization is that Monte Carlo is the method of repetitive trials. This latter definition fits our interest in that what we want to do is to repeat the simulation produced by the DES, but vary the points within each distribution to see the impacts of the uncertainties in the timing, resources, branching, and other probabilistic elements of the simulation. Figure 5 shows an example of the Monte Carlo results of executing the same scenario shown in Figure 3.

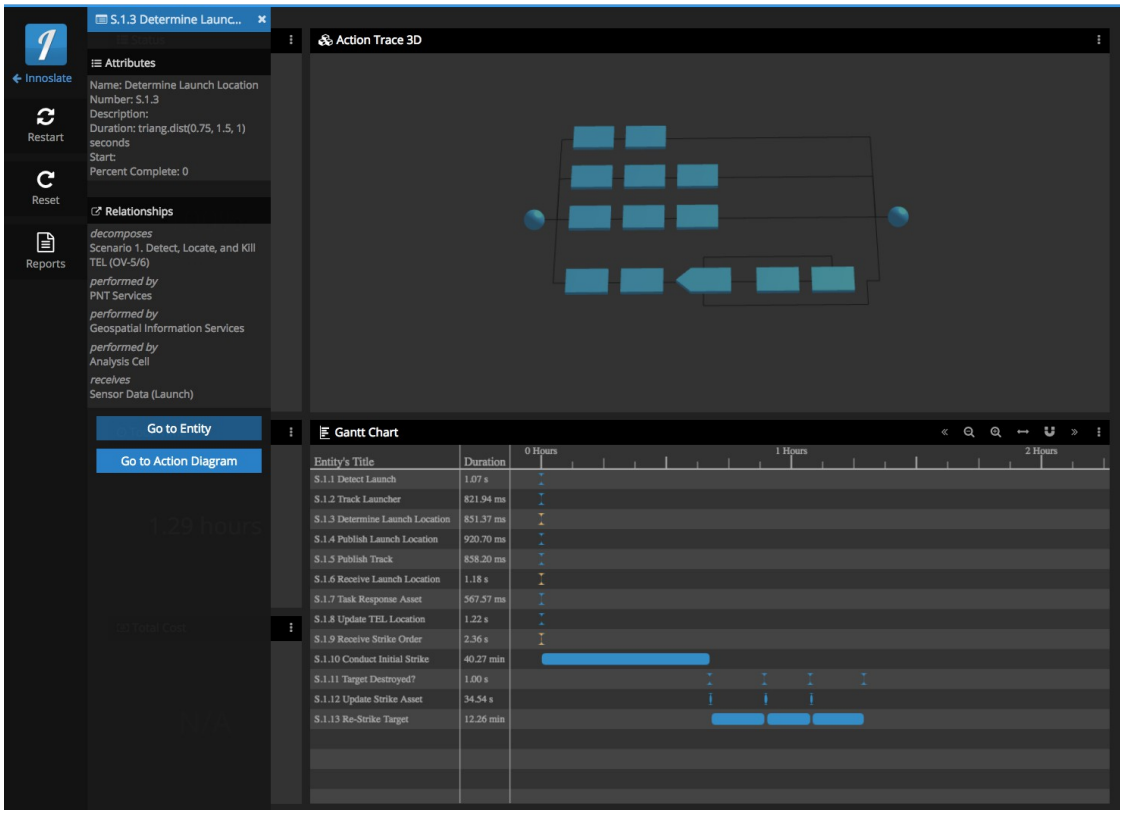

Figure 4. Example of Discrete Event Simulation Output from Scenario 1

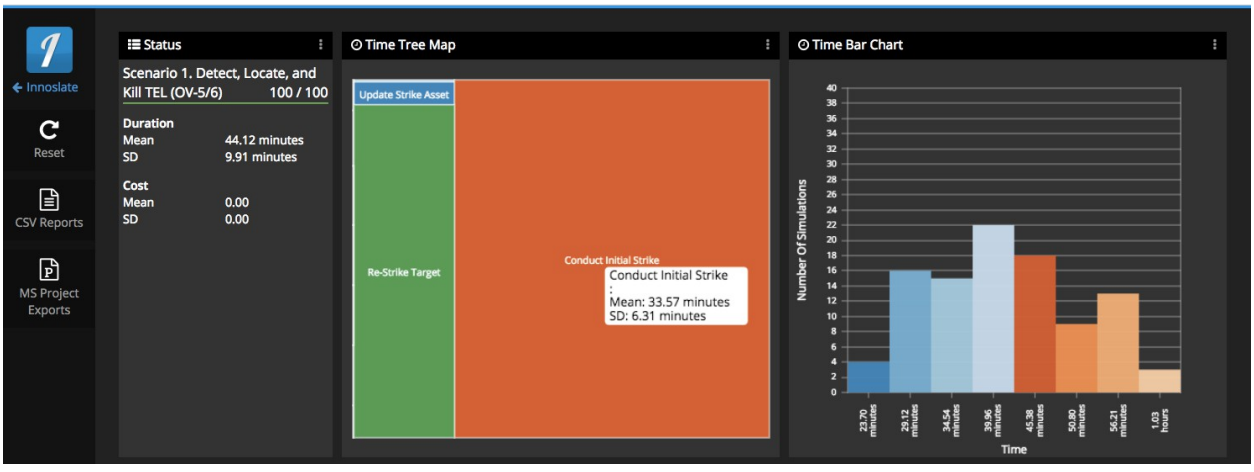

Figure 5. Example of Monte Carlo Simulation Output from Scenario 1

Figure 5 is a Time Tree Map, which provides the mean and standard deviation of each step of the process. The Time Bar Chart shows the distribution in time of the number of runs that occur within the

specified time bins. This example was only executed in the model 100 times. To get a more accurate estimate (higher confidence interval), you can run the simulation many more times [Driels, 2004].

5. Test Planning

Another V&V activity that can at least begin early in the lifecycle is the test planning. As you can imagine, many final operational tests must be conducted at specialized ranges that provide the necessary test equipment and telemetry capabilities needed for final test. These ranges often must be scheduled years in advance. In addition, you may have to develop specialized test equipment and/or arrange for experts/users to participate in this kind of testing. Such tests are also expensive and must maximize the value. These reasons alone drive the need for early test planning.

Figure 6 shows an example of how to capture test expectations and results in a modeling tool.

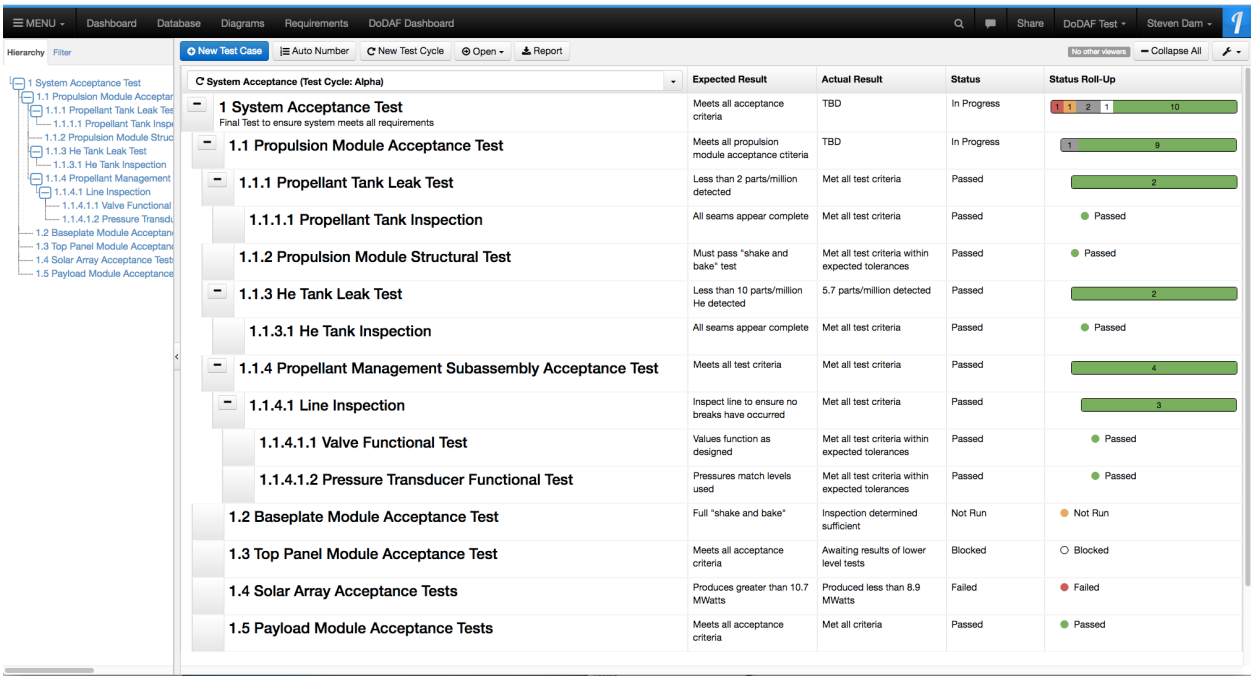

Figure 6. Example of Capturing Test Plans and Results in a Modeling Tool

6. Future V&V

Having the requirements, combined with the models and V&V test plans, enables the potential for automating the V&V process. We can run a simulation against the test case using well defined performance parameters, including time, resource usage, and cost. Using the Monte Carlo simulation provides a confidence interval that can be improved over time as better data become available through the lifecycle process. This capability can then track the progress in improvement of the performance parameters from the initial estimation into final test results. The software industry has had the capability to conduct automated testing for some time due to their use of integrated development environments (IDEs). What is needed is an IDE for systems engineering. The capabilities discussed above come very close to this desired future state.

Summary

Using V&V techniques and effective planning for V&V early in the lifecycle provide many benefits to any project. Although it costs a little more up front, by discovering errors early we save problems later in the lifecycle and even save lives once the system is deployed. Applying these techniques and planning requires understanding the needs of the V&V personnel and thus should encourage the involvement of people with these skills early and throughout the lifecycle.

References

[Deepwater] For an in-depth explanation see https://www.washingtonpost.com/news/achenblog/wp/2016/09/29/deepwater-horizon-movie-gets-the-facts-mostly-right-but-simplifies-the-blame/?utm_term=.34ca084b11e5 (accessed 2/8/2017).

[ASSE] Applied Space Systems Engineering, Wiley J. Larson, et. al., editors, McGraw-Hill Companies, Inc., 2009, p. 387 provides a good summary of NASA Mishap reports and the V&V contributing factors.

[Forsberg] Forsberg, K. and Mooz, H., "The Relationship of Systems Engineering to the Project Cycle," First Annual Symposium of the National Council on Systems Engineering (NCOSE), October 1991.

[Albrecht] Introduction to Discrete Event Simulation, M.C. Albrecht, P.E., January 2010, p. 11 (accessed at http://www.albrechts.com/mike/DES/Introduction%20to%20DES.pdf on 2/9/2017).

[Scenarios] “Intelligent Operational Scenarios – a Strategy for Cost-Saving Scenario Selection,” Steven H. Dam, Ph.D., presented at the July 2007 INCOSE International Symposium.

[Dam] DoD Architecture Framework 2.0, Steven H. Dam, Ph.D., ESEP, SPEC Innovations, 2014, p. 144.

[Tools] Two such tools are Vitech’s CORE and SPEC Innovations’ Innoslate.

[Driels] “Determining the number of iterations for Monte Carlo simulations of weapon effectiveness,” Morris R. Driels, Naval Postgraduate School Thesis, April 2004.