Verification and Validation of Simulation Models

Verification vs. Validation (Sokolowski & Banks, 2009)

Verification and validation are both important aspects of any simulation project. Both are essential in obtaining credible and useful results. Simulations are often considered accurate up to a certain degree that justifies a rather larger risk. Verification and validation are performed using methods suited to the model and to the application that measures its accuracy.

A **simuland** pertains to an object, process, or a phenomenon to be simulated. It may include real-world aspects that affect the object of interest in a significant way, such as the weather condition that affects the flight of an aircraft. Simulands need not exist in the real world; for example, in a combat simulation, hypothetical nonexistent weapons are often modeled to analyze how a postulated capability would affect the battlefield and the outcomes.

The **result** refers to the output produced by a model during a simulation. Results may be available to the user during the simulation, such as the out-of-the-window views generates in real-time by a flight simulator. Results can also be in the form of statistical information produced by discrete events

A **conceptual model** contains all the details regarding the simuland which needs proper representation or must be omitted. The information in a conceptual model may include objects, environmental phenomena, and/or use cases. Different documentation or forms can be used in a conceptual model, such as mathematical equations, flowcharts, diagrams, data tables, expository texts, and unified modeling language (UML).

An **executable model** pertains to a model that can be implemented. A primary example of an executable model is a computer program, which contains series of codes that can carry out the designed specification in a conceptual model.

Verification

- It is the process of determining the consistency of a model based on its specifications.
- It focuses on how well the model satisfies the requirements of the intended application.
- It is concerned with *transformational accuracy* in modeling and simulation, such as transforming model requirements into a conceptual model and the transformation of the conceptual model into an executable model.
- General software engineering methods or applications that can compare the implemented model with the design specification are often applicable.

Validation

- It is the process of determining the degree of accuracy of a model based on the defined simuland in a simulation.
- It is concerned with *representational accuracy*.
- The required accuracy in the validation process must be considered with respect to its intended usages. The degrees of required accuracy may vary depending on the method that was used for validation.
- In validation, specialized methods that can compare the results of executing the model with the simuland are commonly needed.

Important Points

- Verification and validation both involve the process of comparison, and the defining difference between the two are the objects or processes being compared.
- Both verification and validation are part of the simulation process, they do not take place as a separate process.
- The process of verification and validation should proceed as soon as the objects to be compared become available.
- There is a large variety of verification and validation methods due to the different simulation projects with different simulands, and different objects or artifacts produced during simulation.

Different Approaches in Verification and Validation

- **Informal approach** in verification and validation is qualitative in nature. These methods generally rely on subjective human evaluation, rather than detailed mathematical analysis. Experts examine the objects or artifacts of the simulation project and assess the model based on the examination results, their reasoning, and their expertise.
- **Static approach** in verification and validation involves the assessment of the model's accuracy based on its foundational characteristics and the design of the executable model. It often involves the analysis of the programming language codes of the implemented model which are usually supported by automated tools that can perform analysis. These methods are often performed by developers and other technical experts.
- **Dynamic approach** in verification and validation is generally objective and quantitative in nature. Consequently, numerical data and results are

involved. These methods assess the model's accuracy by carrying out the executable model and evaluating its results.

- **Formal approach** in verification and validation utilizes mathematical proofs of correctness. The statements about the model are developed using formal languages or notations, and are manipulated using logical rules. Thus, the conclusions derived from the model are definitely mathematically inclined. Formal methods are quite difficult to apply in practice, yet useful results can be achieved using formal methods in some constrained situations.

Verification Methods (Sokolowski & Banks, 2009)

Common questions during verification:

1. Does the program code of the executable model correctly implement the conceptual model?
2. Does the conceptual model satisfy the proposed usages of the model?
3. Does the executable model produce the expected result at the correct time and in the required format?

- **Inspection** – It is an informal method of verification, where organized teams of developers and testers inspect the model's *artifacts*, such as the document design, algorithms, equations, and the programs' codes. The inspectors manually compare artifacts with the appropriate object of comparison. Note that the person doing the inspection may or may not be the developers of the model being inspected, depending on the resources of the project and the developing organization.
- **Data Analysis** – It is a static verification method that compares data definition and operations on the conceptual model to those in the executable model. This method ensures that all data are properly defined, such as the data types and suitable data ranges, and that proper operations are applied to the data structure in the executable model. Data dependency analysis and data flow analysis fall under this verification method.
- **Comparison Testing** – It is a dynamic verification method that can be used when multiple models of the same simuland are available. The models are executed with the same input and the results are compared.
- **Inductive Assertion** – It is a formal verification method that compares the program codes of the executable model to the descriptions of the simuland in the conceptual model. The proofs are formulated through mathematical induction.

Validation Methods (Sokolowski & Banks, 2009)

Common questions during validation:

1. Is the conceptual model a correct representation of the simuland?
2. How close are the results, produced by the executable model, to the behavior of the simuland?
3. Under what range of inputs are the model's results credible and useful?

- **Face Validation** – It is an informal method of validation that compares simuland behavior to the model. In this method, observers who may be potential users or subject matter experts review or observe the results of a simulation. Then, observers subjectively compare the behavior of the simuland, as reflected in the simulation results, with their knowledge of the behavior of the actual simuland under the same condition.
- **Cause-Effect Graphing** – It is a static validation method that compares the cause and effect relationship in the simuland to those in the conceptual model. *Causes* are the events or conditions present in the simuland, while the *effects* are the outcomes or state changes after an event. Note that the effects may also be the cause of further effects.
- **Predictive Validation** – It is a dynamic validation method that compares specific outcomes in the simuland behavior to the corresponding outcomes in the model. This method can also be used if the input and output values, relative to the behavior, are available.
- **Predicate Calculus** – It is a formal validation method that encompasses a formal logic system that allows creation, manipulation, and proof of formal statements that describes the properties of an object. This method can possibly prove the consistency between the simuland and the conceptual model.

Reference:
Sokolowski, J. & Banks, C. (2009). *Principles of modeling and simulation a multidisciplinary approach*. John Wiley & Sons, Inc.