1981 Winter Simulation Conference Proceedings
T.I. Ören, C.M. Delfosse, C.M. Shub (Eds.)

# TESTS FOR THE VERIFICATION AND VALIDATION OF COMPUTER SIMULATION MODELS

Robert E. Shannon
Professor
Industrial and Systems Engineering Department
The University of Alabama in Huntsville
Huntsville, AL 35899 USA

ABSTRACT

This paper discusses the quantitative as well as qualitative tests which can be run in trying to convince the user of a simulation model that the results are valid

## 1. INTRODUCTION

There is little question that simulation is one of the most powerful analysis tools available to those responsible for the design and operation of complex processes or systems. The concept of simulation is both simple and intuitively appealing. It allows the user to experiment with systems (existing and proposed) where it would be impossible or impractical otherwise. Unfortunately its use also presents the potential for disaster. John McLeod compared simulation to the scalpel used by a surgeon. In the right hands it can accomplish tremendous good, but it must be used with great care and by someone who knows what they are doing.

This paper considers the general problem of how to bring to an acceptable level, the users' confidence that any inference about a system derived from the simulation is correct. Basically, three questions are of concern:

1. Does the model adequately represent the real world system?
2. Is the model generated behavioral data characteristic of the real system behavioral data?
3. Does the simulation model user have confidence in the model's results?

Consequently, we are going to be concerned with tests that fall into three groups: tests of model structure, tests of model behavior and tests of the policy implications of a model.

In the following discussions, we will use the terms verification and validation in the sense used by Fishman and Kiviat [2]. Verification entails the comparison of a model to empirical reality. In the process model structure may be compared directly to descriptive knowledge of the real system structure or model behavior may be compared to observed real system behavior. Validation on the other hand is the process of establishing confidence in the soundness and usefulness of the model's output. A model is created for a specific purpose, and its adequacy or validity can only be evaluated in terms of that purpose. The goal is to generate a model that creates the same problem and behavior characteristics as the process or system being studied. Validation is a continuous process, beginning with the start of the study, that continues as the model builder accumulates confidence that the model behaves plausibly and generates problem symptoms or modes of behavior seen in the real system. Validation then expands to include persons not directly involved in constructing the model. At this point we can further clarify the distinction between verification and validation. While verification is an activity entailing comparison of a model to empirical reality, validation is a communication process that requires the model-builder to communicate the bases for confidence in a model to a target audience. Unless the modeler's confidence in a model can be transferred, the models usefulness will never be realized. Thus through verification testing, the model builder develops personal confidence in the model and through validation measures, transfers that confidence to others.

It is important to realize that validation should be considered one of degree and not an either - or notion; it is not a binary decision variable where the model is valid or invalid. There are no one or two tests which will serve to validate a simulation model, rather, confidence in the usefulness of a model must gradually accumulate as the model passes more tests and as new points of correspondence between model and empirical reality are examined. Testing goes on continuously in the process of constructing and using the model.

Validation must be considered from three different perspectives: (1) the model builder, (2) the technical evaluator, and (3) the ultimate model user. Only the model builder has the capacity to conduct all of the confidence building tests. The technical evaluator (generally a supervisor, referee, or client) is usually limited to reviewing the information and technical data provided by the modeller. The ultimate user rarely has the technical background or mathematical sophistication to be able to understand the verification tests conducted. Yet ultimately, all three levels must be convinced of the models validity if its results are to be used.

## 2. MODEL BUILDING PROCESS

In discussing the model building process, we will follow the basic framework suggested by Zeigler [11]. This framework consists of six elements: (1) the real system, (2) the experimental environment, (3) the conceptual model, (4) the formal model, (5) the computer implementation, and (6) the experimentation.

By the real system we mean that there is a method of isolating out a part of reality, labeling it a real system and collecting information and data about it. The real system is nothing more than a source of potentially acquirable data. At any point in time we will have acquired only a finite subset of this data from what is an infinite set or universe. In general, the real system is (or will become) a source of behavioral data consisting of time based trajectories of input, state and output variables.

The experimental environment, characterizes a limited set of circumstances under which the real system is to be studied. Here we are concerned with the specification of the goals and purpose of the study. The explicit purpose of the model has significant implications for the whole model building, validation and experimentation process. For example, if the model's goal is to evaluate a proposed or existing system in an absolute sense, this imposes a heavy burden upon the accuracy of the model and demands a high degree of isomorphism. On the other hand, if the goal is the relative comparison of two or more systems or operational procedures, the model may be valid in a relative sense even though the absolute magnitude of responses varies from that which would be encountered in the real world. It is not surprising then to learn that a model may be valid in one experimental environment but invalid in another. Thus there may be many valid models (at least one for each experimental environment).

The conceptual model is the model builders perception and understanding of the real world system. Here, the modeller is defining the components, descriptive variables and interactions which constitute the real system. The conceptual model consists of a hypothetical, complete explanation of how the real system functions. The conceptual model often takes the form of a block diagram and systems specifications.

In most cases, the great complexity of the conceptual model precludes its consideration as a possible simulation model due to resource limitations. Fortunately, because of the selection of a particular experimental environment, the modeller can likely construct a relatively simple model (called the formal model) that is valid in that environment. In principle the modeller simplifies by aggregating or lumping together components and elements that are strongly connected through structure, function or both. By lumping together components and simplifying interactions, hopefully a model results that neither oversimplifies the system to the point where the model becomes trivial nor carries so much detail and complexity that it becomes intractable and prohibitively expensive to run. The formal model usually takes the form of a logic flow diagram.

The next step is to write the step-by-step instructions for implementation of the formal model. The computer program implementing a formal model should not be identified with the model itself. Different programming languages encode the same model in different ways. Great care must be taken to insure that the computer program behaves as the modeller intended.

Finally, the computer program is used to generate model behavioral data through planned experimentation and the results analyzed. In this stage there also are numerous pitfalls awaiting the unwary. Depending upon how the experiments are conducted and analyzed, erroneous conclusions can be drawn.

Summing up, in using simulation to study a system or process, there are several different types or classes of error possible, any one of which can lead to drawing erroneous conclusions. Table 1 shows the six elements of the modeling process and the possible errors which can be made. These errors can lead to incorrect functioning of the software and/or incorrect performance of the model relative to user needs. Therefore, we must evaluate the internal consistency of the design (verification) and the external performance characteristics (validation).

Table I

OPPORTUNITIES FOR ERRORS

| Element | Possible Errors |
|---|---|
| 1. Real System | a) Defining the system<br>b) Defining the boundaries |
| 2. Conceptual Model | a) Misunderstanding how system works<br>b) Excluding relevant variables |
| 3. Experimental Domain | a) Specify goals of study |
| 4. Formal Model | a) Design of model<br>b) Data used |
| 5. Computer Model | a) Logic and coding |
| 6. Experimentation | a) Procedure for use of model<br>b) Interpretation |

## 3. SOFTWARE ENGINEERING

Before discussing the quantitative and qualitative tests which can be run for verification and validation, we should at least mention methods designed to try to avoid errors. Computer scientists have been devoting a great deal of effort to trying to develop design and management techniques which will minimize errors. These efforts fall under the general area of Software Engineering. One of the major ideas that has come out of this effort is the validity of the 40 - 20 - 40 rule of software engineering. This rule says that 40% of the overall effort should be devoted to the analysis of the problem and design of the model, 20% to the coding and 40% to the testing/integration of the model. Most errors are the result of starting to write computer code to soon and not leaving enough time for the testing of the model.

During the design and coding phases, it appears that coding errors can be minimized by using:

1. Structured Programming
   a) restrict flow of control to a few basic patterns (usually three),
   b) each module has a single entry and exit point,
   c) properly indented to show nesting of modules within other modules,
   d) no module exceeds 50 lines liberally commented.
2. Top Down Design
   a) design, code, test and integrate higher level modules before doing lower level modules,
   b) testing is done by substituting stubs (dummy programs) for each lower level module,
   c) modules at same level are coded in order in which they will be executed.
3. Chief Programmer Concept
   a) separation of clerical and intellectual tasks,
   b) one programmer responsible for both design and coding,
   c) Programming production library consists of two parts (1) internal library-machine legible on disc or tape, (2) external library - human legible in binders,
   d) programmer works only with external library,
   e) librarian prepares all machine input as directed by programmers, makes all runs and maintains external library.

These concepts are discussed in several articles and books, but all three are presented by McGowan and Kelly [5].

## 4. VERIFICATION AND VALIDATION TESTS

The remainder of this paper is presented in more or less outline form. Some general types of tests we can conduct are [4]:

1. Using fixed values - substitute fixed values for all parameters and variables during construction of each module. This allows checking the results against hand calculated values. Thus one can detect and correct logical errors by producing a step-by-step trace of the model.

2. Testing the model against analytical models - Obviously if the problem could be solved analytically, we would not be simulating it. However, often times we can test some of the modules (such as queue build-up) against an analytical analysis.

3. Trace driven method - If the system being modeled already exists and the data is available (or obtainable), one can use actual data to drive the model. Thus instead of sampling statistical distributions to generate job characteristics (characteristics or attribute values of entities flowing through the system) a job profile of actual values is read and its actual characteristics used. In this way exactly the same load as that of the real system would go through the simulator. This method allows direct comparisons of the simulator's and the system's performance on individual jobs.

4. Statistical analysis - Tests of means, variances etc. between real data and model data. Tests of such things as interarrival times, process times, down times etc. can be conducted for input, status and output variables.

## 4.1 Reasonableness Tests

If a model is to appear credible, it must exhibit behavior that is similar to the real world. Some aspects of this reasonable behavior are continuity, consistency and response to degeneracy and absurd conditions [8].

1. Continuity - requires that small changes in input parameters cause consequent small changes in output and state variables unless they can be understood and justified by the underlying process.

2. Consistency - essentially similar runs of the model should yield essentially similar results, i.e., there should not be widely differing responses due to changing the initial seeds for the generators.

3. Degeneracy - when certain features of the model are removed, the output should reflect their removal. Example if I remove a new piece of machinery then the model should respond as if it is not there.

4. Absurd conditions - (a) if I introduce absurd inputs I should not get equally absurd outputs, i.e., if I increase advertising budget to $\infty$ then sales should not also go to $\infty$, (b) absurd conditions should not arise during the simulation, i.e., people with negative height, cars going two ways at once.

## 4.2 Tests of Model Structure

1. Structural - Verification Test

   Verifying that the structure of the model does not obviously contradict reality. There must be a mapping or homomorphism between the conceptual model and the computer model.

2. Parameter and Relationships - Verification Tests

   Tests of the underlying assumptions about parameter values and variable relationships. These are usually statistical tests of means, variances, regression analysis, goodness-of-fit tests, etc.

3. Extreme - Condition Tests

   Structure and output should be plausible for any extreme and unlikely combination of levels of factors in the system, e.g. if in process inventories are zero - output should be zero. Also the model should bound and restrict the behavior outside of normal operating ranges.

4. Boundary - Adequacy Test

   Considers whether the model incorporates the relevant relationships necessary to satisfy the models purpose. Closely tied to structural verification testing.

5. Dimensional - Consistency Test

   Entails dimensional analysis of the models equations.

## 4.3 Tests of Model Behavior

1. Behavior - Reproduction Tests

   Comparison of model behavior and output to historical behavior and output [10].

   a) Analysis of variance

   b) $X^2$ test

   c) Kolmogorov-Smirnov test

   d) Regression analysis - X vs. Y does it give slope = 1 intersect = 0?

   e) Spectral analysis

   f) Theil's Inequality test [3]

   g) Turing test [9].

2. Symptom Generation Tests

   a) Does the model recreate the difficulties which show up in the real world system?

   b) Does the model produce known results under certain inputs e.g. if unemployment increases do sales decrease?

3. Behavior-Prediction Tests

   Testing whether the model can predict system behavior through field tests [10].

4. Behavior-Anomaly Test

   If behavior of model is contrary to our knowledge of the real system but we can find instances where in fact the real system did behave this way it is strong proof of validity. If we cannot find instances of such behavior, we trace the model structure that created such behavior.

### 4.4 Tests of Policy Implications

1. Changed-Behavior Test

   If previous changes have been tried in the real world system, we can make the same changes in the model and see if it reacts the same way the real world system did.

2. Sensitivity Analysis Tests

   By conducting sensitivity analysis by varying the values of the parameters and seeing how it changes the behavior, we can get a feel for the impact of uncertainty in parameter values. If slight changes in parameter values lead to different policy implications, we should tread carefully.

## 5. CONCLUSIONS

The problems of verification and validation are perhaps the most critical issues faced by the simulationist. It would be nice if one could just run two or three nice clean statistical tests and everyone concerned accept the validity of the model. Unfortunately, however, that is not the case. Each study, and more importantly, each user presents a unique challenge. It is fairly easy for the modeller to be convinced of the validity of the model. It is far more difficult and more critical to convince the user.

The literature on the validation of simulation models is extensive. The last two proceedings of this conference have contained excellent papers [6,7] and an extensive bibliography containing 125 references has been published by Balcı and Sargent [1].

## 6. REFERENCES

Balcı, O. and R. G. Sargent (Spring 1980), Bibliography on Validation of Simulation Models, Newsletter of The Institute of Management Sciences College on Simulation and Gaming, Vol. 4, No. 2.

Fishman, G. S. and P. J. Kiviat (1968), The Statistics of Discrete Event Simulation, Simulation, Vol. 10, pp.185.

Kheir, N. A. and W. M. Holmes (April 1978), On Validating Simulation Models of Missile Systems, Simulation, Vol. 30, No. 4, pp. 117-128.

Lazos, C. (December 1979), Evaluating Computer Systems Simulation Models, Proceedings of 1979 Winter Simulation Conference, pp. 309-316.

McGowan, C. L. and J. R. Kelly (1975), Top-Down Structured Programming Techniques, Petrocelli/Charter, New York.

Oren, T. I. (1980), Assessing Acceptability of Simulation Studies, Proceedings of 1980 Winter Simulation Conference, Vol. 2, pp. 19-22.

Sargent, R. G. (1979), Validation of Simulation Models, Proceedings of 1979 Winter Simulation Conference, Vol. 2, pp. 497-504.

Schlesinger, J. R. et al (1974), Developing Standard Procedure for Simulation, Validation, and Verification, Proceedings of 1974 Summer Computer Simulation Conference, pp. 927-933.

Schruben, L. W. (1980), Establishing the Credibility of Simulations, Simulation, Vol. 34, No. 4, pp. 101-105.

Shannon, R. E. (1975), Systems Simulation: The Art and Science, Prentice-Hall Inc., Englewood Cliffs, N. J.

Zeigler, B. P. (1976), Theory of Modelling and Simulation, John Wiley and Sons, N. Y.