# A Method for Quantified Confidence of DEVS Validation

**Megan Olsen**
Department of Computer Science
Loyola University Maryland
4501 N Charles St., Baltimore, MD 21210 USA
mmolsen@loyola.edu

**Mohammad Raunak**
Department of Computer Science
Loyola University Maryland
4501 N Charles St., Baltimore, MD 21210 USA
raunak@loyola.edu

## ABSTRACT
The Discrete Event System Specification (DEVS) framework provides a formal approach for defining a conceptual model that represents a source system. To use the model in analyzing that system, it must be validated. Generally DEVS models are validated in terms of behavior and structure, utilizing the experimental frame. Although there has been discussion on approaches to build valid DEVS models, and techniques for validating the model, we currently cannot quantify our confidence in the model's validity. We propose a technique to quantify our confidence in the validity of DEVS models. This metric can be utilized to discuss the level of validity of a model and compare competing models. Our metric continues the DEVS trend of making modeling and simulation more formal, and increasing standardization of the modeling and simulation process.

### Author Keywords
DEVS; Validation; Verification

### ACM Classification Keywords
I.6.4 Model Validation and Analysis; I.6.1 Simulation Theory

## INTRODUCTION
The field of Modeling and Simulation (M&S) is growing rapidly in usage and influence. Computational simulations are created to study many fields by computer scientists, engineers, and experts in the studied domain such as cancer biology, ecology, or psychology. Regardless of the process that is followed and the tools that are used to develop these models and simulations, the models and simulations must be properly verified and validated before their results can be trusted. Validation in particular is a crucially important part of building confidence in the results derived from a simulation model.

Verification and validation (V&V) are defined in a number of ways in both the software engineering and modeling and simulation communities. The definition we find to be most accurate and most widely referenced is that verification determines whether the model solves the problem correctly, whereas validation determines whether the model solves the correct problem. In essence, verification is testing *how* the model was built, and validation is assessing *what* has been modeled and whether the model can sufficiently reproduce the behavior of the real system for its simulation purpose. There are many steps within the modeling and simulation process where each must be applied to achieve high confidence in the model and simulation [1].

*SpringSim-TMS/DEVS, 2015, April 12 - 15, Alexandria, VA, USA*


This definition of V&V has also been commonly accepted and applied to the special subclass of simulations defined using a formalism in the field known as DEVS (Discrete Event System Specification) [28]. DEVS provides a mechanism by which a model is explicitly defined separately from a simulation, which is often lacking in simulation practice. As Tolk recently showed, one of the issues plaguing the field of Modeling and Simulation is a lack of explicit terminology, including a definition of a conceptual model [20]. We argue that the DEVS model is in essence the conceptual model of the simulation, and should be treated as such for V&V.

Formalisms such as DEVS have increased the rigor in the field and addressed issues with explicit model definitions. However, DEVS has not been fully extended to incorporate verification and validation. In general, validation is performed to increase one's confidence in the model. In DEVS as with other M&S approaches, there is no mechanism to quantify confidence in the validity of a simulation model based on the performed validation. Thus the users of the simulation model, who may not be the ones who developed it, may find it difficult to decide which available model is the best to use for their specific purpose. The DEVS community has increased emphasis on standardization [25], and we propose that standardization should extend to include a process to quantify confidence in simulation models built using the DEVS formalism. We argue that following such a standard process will clarify the level of validation performed and corresponding confidence in the model for both the model's creator and any other party considering the model, which in turn will help increase the reuse of models.

In this paper we present an approach to calculate an explicit confidence level for a DEVS simulation model, based on validation performed on that model. We focus on behavioral, structural, and data validation of the simulation model. In the next section we discuss related work in validation within and outside of DEVS, then provide an overview of V&V in DEVS, followed by a description of computing behavioral confidence level, structural confidence level, and confidence on the model data. We show how our proposed framework combines these three confidence levels to produce an overall quantified level of confidence of the whole model. We end with an example of applying our approach on a simple DEVS model and conclude with a discussion on the potential impact and future direction of this research.

## RELATED WORK
For DEVS models, it can be difficult to make a careful distinction between whether the DEVS model is coded cor-

rectly (verification) and whether it represents the source system closely enough for its purpose (validation). DEVS discusses validation through the use of an experimental frame, which defines the purpose, behaviors, structure, and data of the model [28]. Structural validation focuses on the static parts of the model and ensures, through the application of specific validation techniques, the structural consistency and integrity. Behavioral validation focuses on the dynamic part of a DEVS model. Both structural and behavioral validation are focused around the model's purpose.

The M&S community in general considers the validation process to include a) conceptual model validation, b) operational validation of the simulation model, and c) data validation. They primarily focus on operational validation, as the conceptual model is not generally as well defined. They identify validation techniques that can be applied to increase one's confidence in a model, such as extreme condition tests, results validation, and face validation [19, 8]. It is well accepted that these techniques should be applied throughout the modeling process, in addition to performing verification checks [2]. However, less focus has been placed on quantifying the validation effort on simulation models.

Labiche investigates the application of software engineering testing techniques for performing V&V on DEVS models [12]. The paper discusses selecting test cases based on some criterion, but does not specify any mechanism for quantifying the coverage achieved through that selection. Zengin examines the question of how much validation is enough by performing a case study on a DEVS-Suite model by examining available validation techniques and discussing the difficulties of validating simulation models [29]. They find DEVS-Suite to be a useful tool for validation, and suggest that a tool is needed to aid in successful validation approaches.

Wang proposes a method of assessing V&V activities based on detection, classification, and documentation of model defects during the simulation model construction period [26]. The focus of Wang's work is properly classifying the defects, akin to IBM's orthogonal defect classification (ODC) [4]. Wang also proposes a framework for cataloging V&V techniques and selecting specific V&V techniques for an M&S project based on applicability [27], which can be used in tandem with the work described in this paper.

Hollmann attempts to formalize the verification and validation process of DEVS modeling [10]. They propose a method to choose the most significant set of configurations for validation of a DEVS model to increase confidence in the model's results. This paper builds on Hollmann's work as we focus not on choosing parameters for running the simulation, but instead on criteria for measuring confidence in the model afterward.

We note that DEVS is not the only formalism used for M&S. SysML extends UML for systems engineering and is used successfully in modeling and analyzing processes involving software and other resources [11]. SDL is primarily used for modeling communication networks and protocols [7]. Petri nets and colored Petri nets (CPN) are often used to model distributed and parallel systems [9]. All of these modeling mechanisms are grounded in formal specification, have tool support to develop simulation models, and generally lend themselves well for verification. The DEVS formalism is the most widely used formalism for developing discrete event simulations due to its specific focus, flexibility, and composability [30]. We therefore focus on the DEVS formalism for this work.

Although validation is performed on most models, it is not generally discussed sufficiently [6, 17]. There is an imminent need to improve the documentation of validation activities performed on simulation models, which could be accomplished by standardizing and quantifying validation. Within DEVS there have been numerous steps to standardize, such as standardizing the simulation of DEVS models in tools [13], or formalizing models and their context through experimental frames to improve validation [21].

In our earlier work we developed approaches to quantify validation coverage in agent-based models [14] and in discrete event models [16]. In this paper we build on our earlier work by overhauling the approach to focus on calculating a level of confidence in the model through the validation of behavior, structure, and data for DEVS, leading to an overall quantified level of confidence in the entire model. We propose that in future work the validation confidence level derived from DEVS could be extended for other modeling approaches and languages (e.g.SysML, CPN) as well.

## VALIDATION PROCESS

A formalism can improve the process of modeling and simulation by making certain steps and processes more concrete. DEVS was shown to provide a more suitable process for some analyses than other modeling approaches (Figure 1) [15]. The DEVS formalism provides a concrete conceptual model, and thus conceptual model validation can be more rigorously performed. The formalism is ideally built using data about the real system that is being studied, known as the source system or System Under Study. Generally an executable simulation model is built to experiment with the conceptual model, using one of the many frameworks available, such as CD++ [22], DEVS-C++ [5], or PowerDEVS [3]. These experiments are run using an experimental frame, which is essentially a mechanism for clarifying the data, parameters, and structure of the simulation model. The experimental frame performs two roles: providing data for the creation of the model, and including all data and parameters necessary for running the simulation. For the results of the simulation model to be trusted, it must be verified against the conceptual model, as well as operationally validated using the experimental frame.

We define the experimental frame as in [28], that it can be utilized toward two related goals: defining the types of data to be used, and interacting with the model to either measure or observe results. We use the experimental frame for both purposes, to define our data and as an oracle when an oracle is possible.

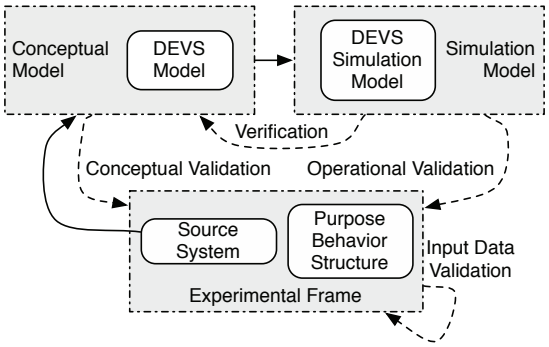


**Figure 1. Overview of the DEVS modeling process, with a focus on verification and validation.**

Although DEVS provides a useful formalism for defining a conceptual model, it does not necessarily aid in operational validation (how well does the model mimic the source system's behavior). To determine the extent of validation, and to properly validate a simulation model, we must be able to determine the aspects of the real-world system that are validatable within the simulation model. The fact that DEVS is the underlying modeling approach is irrelevant at this stage; validation does not care about how the simulation model is created, only what is being modeled and how it is being assessed for operational accuracy. For a single DEVS model, whether atomic or coupled, any aspect specific to the formalism (input/output of an atomic model, links between coupled models, etc) is a part of the simulation that needs to be verified for transformational accuracy between the conceptual model and the simulation model. In the case of verification, quantifying the success is similar to any software system; we could use a coverage metric that counts whether each of the formalism's statements works as planned, for instance. However, that metric does not help us compare the simulation to the broader real-world picture of the system being modeled.

We consider DEVS validation to encompass the experimental frame's three main properties: purpose, behavior, and structure [28]. The purpose guides us, defining why the model was built and what type of questions may be asked of it. The behavior and structure of the model are based on that purpose. Behaviors are essentially pairs of input and output; given a certain state/input of the model, how should it act? When validating behavior we are validating structure indirectly, but to have full confidence in a model we also need to perform explicit structural validation. To calculate the level of confidence in the model, we must track the validation of both behaviors and structure with regard to purpose, and validate the data used in simulation model execution.

## OVERVIEW OF CALCULATING MODEL CONFIDENCE

We have argued that to measure the confidence level on a DEVS simulation model completely, the behavior, structure, and data of the model need to be validated and the level of that validation needs to be assessed. In this section, we provide a high-level description of the proposed process, and in the following sections present techniques for calculating confidence in the model's validity.

The first step in behavioral validation is to identify all behaviors captured by the model that require validation. The second step involves determining what validation techniques are applicable to each of the identified validatable behaviors. The next step is to perform validation on the simulation model by applying those validation techniques. When a particular validation technique is applied on a simulation model it may validate more than one behavior, as many behaviors may be interconnected in such a way that they may not be validated separately. As validation is performed, the validation techniques applied and their level of success needs to be recorded. The final step is to calculate confidence in the model. This confidence level quantifies the amount of validation performed successfully versus the amount that could be performed on the model behaviors.

From the perspective of the experimental frame, the structural aspect of a DEVS model incorporates elements of both the conceptual model and the simulation model. Quantifying structural validity requires fewer steps, however. There is a set of techniques already identified to be applicable for performing structural validation [8]. We propose applying these techniques and using the percentage of these techniques that have been successfully applied to determine our level of confidence in the structural validity of the model. Similarly, we propose calculating data validation based on the successful application of a set of validation techniques on each set of input data to ensure it properly matches the source system and is free of outliers and other errors.

We present a mechanism for combining all three quantified aspects into an overall confidence level for the validity of the model. All levels of confidence are in the range of $0$ to $100$, with $100$ representing the maximum possible confidence in the model but not that the simulation is $100\%$ correct. However, it is often impossible to fully validate a simulation model due to constraints such as budget, time, and the absence of required data. A desired level of confidence should therefore be determined at the requirements stage, to guide later validation efforts. Additionally, the confidence calculation can also guide the choice of validation techniques to apply as the goal should be to perform the validation that will increase model confidence most efficiently.

## BEHAVIORAL VALIDATION

To calculate our confidence in the validity of a model, we first determine the behaviors of the model that need to be validated. We consider behaviors to encompass a single high-level input/output pair in the DEVS model, related to a definable expectation in the source system. Multiple behaviors will generally be necessary to define a specific module. For instance, in a hospital emergency department (ED) simulation, a behavior could be the process by which a patient is placed inside the ED upon arrival in a simple case, or a sub-process of that process flow (e.g., registration) in a more complex model. Similarly, while simulating the operations of an ATM machine, a behavior could be the process of withdrawing money or a sub-process such as checking whether the entered PIN is correct.

| Technique | Maximum Confidence |
|---|---|
| Animation | 3 |
| Degenerate Tests | 4 |
| Internal Validity | 5 |
| Comparison to other model | 6 |
| Turing Tests | 7 |
| Face validation (expert) | 8 |
| Results validation (historic data) | 10 |
| Trace data | 10 |

**Table 1. Ranking of the most common validation techniques with the maximum level of confidence they can provide for a given behavior.**

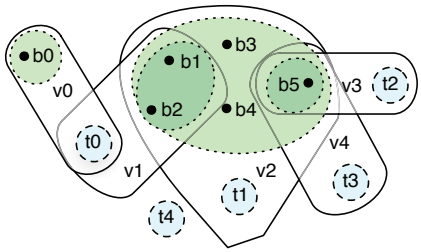


**Figure 2. A visualization of each element of a $v_i$. Each $t_i$ is a validation technique, each $b_j$ is a behavior, each dotted circle around a set of $b_j$ is a $d_i$, and each full shape is a $v_i$. Note that each $b_j$ can be in multiple $v_i$. The size of the shape represents $c_i$.**

To aid in determining the behaviors of the model, we suggest utilizing the previously published categories presented as *aspects* and *oeis* (observable emerging information) of a discrete event simulation [16]. Although the coverage metric proposed in that work is different from our confidence level in this paper, the simulation behaviors discussed here are identical to that concept of *oei*s.

Once we have identified all behaviors that must be validated, we must determine the applicable validation techniques for each of them. A single technique may apply to many different behaviors, and each behavior will often be validated with more than one applicable technique.

In our proposed framework we take into account that not all validation techniques will contribute equally toward building confidence in the simulation model. By applying any validation technique successfully, we increase our confidence in the validity of the model. However, which validation technique(s) should be applied for each behavior? We propose utilizing rankings such as those in Table 1. Each commonly accepted technique is listed with a suggested relative level of possible confidence increase.

The proposed ranking is based on a survey of the most commonly used validation techniques [17], a discussion of their uses [19], and the authors' experience. We rate animation the lowest as it can be misleading and hide underlying issues, but rank results validation and trace data the highest as they can be powerful techniques when the required data exists. We rate expert validation slightly below the validation involving data due to the potential for human error, and model comparison slightly less due to difficulty in knowing the validity of the other model without a standard confidence calculation. We rank Turing tests between these two as they are rarely discussed in published papers, and are a subset of face validation. Additional techniques exist and can be ranked as necessary given the purpose of the model, as well as to encompass additions to DEVS such as Cell-DEVS [23]. This ranking is subjective and used only for the purpose of illustration. More research and case studies from the community will be needed to establish a more generally accepted ranking of validation techniques.

Given a simulation model to validate, techniques are chosen based on their usefulness in creating the highest possible confidence in the model. There may be more validation techniques that are applicable than are necessary to achieve a reasonable level of high confidence in the model. For the highest model confidence, however, we must apply every applicable validation technique successfully. If it is not possible to apply all applicable techniques for any reason, we can determine our level of confidence as described in the next section.

### Confidence Calculation

Given a simulation model's list of behaviors, the list of applicable validation techniques by behavior, and the list of successfully applied validation techniques, we can quantify our confidence in the model's behaviors. For some DEVS simulation model $M$, let $B = \{b_0, b_1, \ldots, b_n\}$ be the set of behaviors to be validated, and $T = \{t_0, t_1, \ldots, t_m\}$ be the list of available validation techniques for operational validation such as those in Table 1. Each of these elements has an associated confidence weighting $w(t_i)$, defined as in Table 1. For $b_i$, the weight $w(b_i)$ is determined based on the impact of this behavior for model validation, and $0 < w(b_i)$. For instance, a behavior may have a very strong affect on overall validation, or may be almost trivial. In the simplest case all behavior weights are equal.

When performing validation it is possible that a single validation technique is applied successfully and validates multiple behaviors simultaneously. Therefore, a validation technique must be paired with some set $D \subseteq P(B)$, where $|D| \geq 1$ and $P(B)$ denotes the power set of $B$. A validation technique $t_i$ may be used multiple times when validating a single simulation, each time used to validate a different set of behaviors by focusing on a different aspect of the simulation model and/or using different sets of available data or experts. We must consider a single validation approach $v_i \in V$ as a triplet $v_i =< t_i, d_i, c_i >$, where $t_i \in T$, $d_i \in D$ is a set of $b_j$s, and $c_i$ is the confidence gain from applying $t_i$ to $d_i$. There can only be one $t_i \in v_i$, as each validation technique is considered independently, although it may validate more than one $b_i$ with one application. Each $b_j$ and $t_i$ may be duplicated as many times as necessary in $V$. Figure 2 demonstrates the relationship between $v$, $t$, and $b$.

The $c_i \in v_i$ represents the confidence gain from $t_i$'s successful application. It is based on the weight of each behavior in $d_i$ represented as $w(b_j)$, and the weight of the validation technique $w(t_i)$ as seen in Equation 1. Therefore, higher importance behaviors contribute more to our confidence in the

system. Additionally, more powerful validation techniques contribute more to our confidence than less powerful techniques.

$$c_i = w(t_i) * \sum_{j=0}^{|d_i|} w(b_j), \text{where } b_j \in d_i \quad (1)$$

Recall that confidence is based on what validation techniques are applied successfully. Each validation technique successfully applied raises our confidence in the validity of the model. To quantify this level we must define the highest possible confidence. We base the maximum confidence on the potential validation approaches $P = \{v_0, v_1, \ldots, v_p\}$, where $P \subseteq V$; and which validation approaches succeeded, $A = \{v_0, v_1, \ldots, v_a\}$, where $A \subseteq P$. We assume that any validation approach $v_i$ that succeeded, did so for all $b_j \in d_i$. Finally, we may calculate our confidence in the behaviors across the entire simulation model $bc$ as in Equation 2.

$$bc = 100 * \frac{\sum c_i | c_i \in v_i \wedge v_i \in A}{\sum c_i | c_i \in v_i \wedge v_i \in P} \quad (2)$$

Our level of confidence in the behavioral aspect of the simulation model is therefore a real number, $0 \leq bc \leq 100$, where 100 represents the highest possible confidence. Note that confidence does not equal correctness, and this value does not imply that the simulation model is $100\%$ correct. A confidence level of 100 implies that the simulation model has been validated to the fullest extent possible by currently known techniques.

## STRUCTURAL VALIDATION

The structure of a model is indirectly validated through behavioral validation, but for the highest possible confidence in a model it should be directly validated as well. Forrester and Senge [8] recommend the tests listed in Table 2 for validating structure.

Each technique plays an important role in structural validation. Although many have the word "verification" in the name, they are validation tests as they compare the model to the real system. The structure verification test verifies that any structure found in the model is also found in the source system; it is the easiest test to pass. The parameter verification test evaluates whether the parameters adequately match the real system, which is only true if the correct structure underlies the model. Extreme condition tests are necessary to determine that the model will function in abnormal conditions; if it does not, then more structure is needed, but if it does then the correct structure was chosen. The boundary adequacy test essentially determines if the correct level of abstraction has been chosen, which correlates to structure. The dimensional consistency test can reveal incorrect structure assumptions when run with the parameter verification test, and is another simple way to increase model confidence.

To have full confidence in a simulation model, all structural validation tests should be successfully administered. However, there may be constraints that prevent a test from being applied. In that case, the confidence in the model is calculated as discussed in the next section.

| Technique (st) | Potential Confidence (c(st)) |
|---|---|
| Structure verification test | 10 |
| Boundary adequacy test | 15 |
| Parameter verification test | 20 |
| Dimensional consistency test | 20 |
| Extreme condition test | 35 |

**Table 2. Ranking of the most common structure validation techniques with their maximum possible level of confidence.**

### Confidence Calculation

As with behavioral validation, each structural validation technique may be applied at different levels of success. In this case we propose that each structural validation technique $st_i$ has the confidence level $c(st_i)$ attainable via successful application, as shown in Table 2. This confidence level is based on [8, 29].

Each structural validation test may validate a specific percentage of the model. For instance, our model may pass 90% of the tested extreme conditions. We have thus gained confidence in our model, but not full confidence as we are aware of extreme condition test failures that must be fixed before trusting the model's results in all cases. We therefore propose that overall confidence in the structure of the model may be quantified as in Equation 3, where $p(st_i)$ is the level of successful application of $st_i$ (0.9 for extreme condition testing in our example).

$$sc = \frac{\sum_i c(st_i) * p(st_i)}{\sum_i c(st_i)} \quad (3)$$

If all tests are applied successfully we gain the highest possible confidence in the model's structure (100%). If all are successful except extreme condition tests had a 90% successful rate, our confidence in the model's structure is 94%. Recall that this confidence value is in the correctness of the structure matching the source system, not the percentage of the system we know to be correct.

## DATA VALIDATION

A primary aspect of validation is to ensure the validity of the data used for model creation and model execution. Sargent describes three necessary processes for data validity: "(1) collecting and maintaining data, (2) testing the collected data using techniques such as internal consistency checks, and (3) screening the data for outliers and determining if the outliers are correct" [19].

We can also validate our input data specifically to ensure that it is consistent with the source system. In DEVS, this input data is part of the experimental frame, and used for running behavioral validation. A key differentiation here is that there are two types of data: the data used to build the model, and the data used to run the model. Here data validity refers to the validation of data used to run the model, as the data used to build the model is validated through structural validity as discussed in the prior section.

We propose that Goodness of Fit and Face Validity are the minimum validation techniques necessary for data validation [18]. Either technique could be applied multiple times, as for instance the goodness of fit test should be used to validate each distribution within the model. Additional techniques can also be added as appropriate.

### Confidence Calculation

All applicable validation tests need to be run on all input data to achieve maximum confidence that we are using the correct data for our simulation. To calculate confidence in the data, we need to enumerate the list of data $I = \{i_0, i_1, \ldots, i_r\}$. This list is part of the model's experimental frame, and we assume that the modelers will collect it as part of their modeling and validation work.

Let $DP = \{dp_0, dp_1, \ldots, dp_p\}$ denote the set of applicable data validation techniques. For each dataset $i_j$ we have the set of successfully applied data validation techniques $ADP_j \subseteq DP$. If we assume that all data validation techniques are equally important, then $c(i_j)$, the confidence in each dataset $i_j$, can be computed by Equation 4. By dataset we refer to any set of data used to run or build the model, such as a distribution or set of input/output pairs.

$$c(i_j) = 100 * \frac{|ADP_j|}{|DP|} \tag{4}$$

A simulation modeler may decide that some input data are more important to validate than others. In that case, we propose incorporating the relative importance of the input data by attaching weights to each $i_j$ denoted as $w(i_j)$. The overall confidence in data validity is then defined as in Equation 5.

$$dc = \frac{\sum c(i_j) * w(i_j)}{\sum w(i_j)} \tag{5}$$

## OVERALL CONFIDENCE

Once we have confidence in the behavior, structure, and input data of the simulation model, we can define confidence in the model as a whole. We propose that behavioral and structural validation are equally important. However, as behavioral validation indirectly validates some aspects of structure, their validation confidence levels should not equally influence our overall confidence in the simulation model. Additionally, data validation is crucial for correct behavioral validation, but very hard to perform completely across all data for the system. In reality, assumptions of data validity based on data source and collection are more common. We therefore propose the following calculation for overall model confidence $mc$:

$$mc = 0.5bc + 0.3sc + 0.2dc, \tag{6}$$

where $bc$, $sc$, and $dc$ are as defined previously. Thus, our confidence in model validity is slightly more affected by behavioral validation than structural or data validation.

## EXAMPLE

To demonstrate the confidence calculation we use a model of an Automated Teller Machine (ATM) as an example, such as the publicly available one in the CD++ Sample Models [24]. Our only modification is to allow PIN and Amount to be input instead of assumed rates. The purpose of this model is to represent a cash machine.

### Behavioral Validation

To validate the behaviors, we first determine what ATM behaviors should be validated to match the source system. Let's assume that the behaviors are as follows:

1. $b_0$ =An incorrect PIN number causes card return and no additional screen options, $w(b_0) = 3$
2. $b_1$ =Requested money is given if account has sufficient fund, otherwise not, $w(b_1) = 4$
3. $b_2$ =The user interface question order follows the correct workflow, $w(b_2) = 2$
4. $b_3$ =The card is always returned after completed use, $w(b_3) = 2$

Note that most behaviors do not have an obvious correlation to a single atomic model within the DEVS conceptual model. One confusion in this model is that many of the behaviors to be validated are also aspects of the model that must be verified as well. In a more complex model that would not be the case, as behaviors would be further removed from the atomic and coupled models. However, verification would involve looking at finer-grained detail about individual aspects of the model and how they work to accomplish the goals.

Next we must determine the sets of potential validation approaches, $P \subseteq V$. Let's assume that the validation techniques from Table 1 are the only available validation techniques, and that they are applicable as follows:

- $p_0 = \{\text{Animation}, \{b_0, b_1, b_2, b_3\}, 33\}$
- $p_1 = \{\text{Face Validation}, \{b_0, b_1, b_2, b_3\}, 88\}$
- $p_2 = \{\text{Results validation}, \{b_1, b_2\}, 60\}$
- $p_3 = \{\text{Results validation}, \{b_0, b_3\}, 50\}$

The next step is to perform the validation. If all tests were run successfully, then our set of applied validation techniques would be $A = P$, and our confidence in the model's behavior would trivially be $100$.

Alternatively, if we were unable to run a validation approach or it fails, such as $p_0$, we have $A = \{p_1, p_2, p_3\}$ and our confidence in the model behavior becomes

$$bc = 100 * \frac{88 + 60 + 50}{33 + 88 + 60 + 50} = 85.7\%. \tag{7}$$

### Structural Validation

Let's assume that of the five available structure tests, we were able to successfully perform all but the structure verification test, due to time and budget constraints. Our confidence in the model structure is therefore

$$sc = 20 + 35 + 15 + 20 = 90\%. \tag{8}$$

### Data Validation

The input data necessary for this model are PIN inputs, withdrawal amounts, and corresponding account balances. The data that needs to be validated relates to the rates at which

PINs are correctly entered, and the rates at which accounts may be overdrawn. Since the model and data are simple, we can assume that all data were validated with all techniques, giving us $dc = 100\%$.

### Overall Confidence

Our overall confidence in this simulation model assuming that the behavioral validation is given by Equation 7 and the structural validation is given by Equation 8, is

$$mc = 0.5 * 85.7 + 0.3 * 90 + 0.2 * 100 = 89.85. \quad (9)$$

This confidence level should be interpreted as high confidence in the model, although it also makes it clear that we are not yet completely confident. For a simple model such as the ATM, 100% confidence may reasonably be expected before the model's use. However, most models are significantly more complicated, and thus a confidence level less than 100% will often be sufficient for judging the model as adequate. This confidence calculation supports the primary goal of providing a quantified value for use in comparison and to enable discussion about validation levels of simulation models.

## CONCLUSIONS

In this paper we present an approach for computing confidence in DEVS model validity. Prior to this paper, we are unaware of any attempt to quantify the level of validation in a DEVS simulation model. In our approach, confidence in a model increases with each successful application of a validation technique, and decreases with each unsuccessful application or unused applicable technique. Although Balci has previously argued that confidence need only be discussed qualitatively, we propose that a quantified level of confidence can both guide our choice in model and aid us in convincing those who did not build the model of its accuracy. Additionally, a quantified model confidence can increase the standardization of validation, which will in turn increase the scientific strength of the field.

This approach builds on our previously proposed metric of validation coverage for agent-based and discrete-event simulation models, sharing the idea of quantifying validation by focusing on behaviors. In addition to behaviors, we now calculate model confidence in terms of structure and data as well, and utilize the experimental frame in DEVS. Behavior validation should be utilized to replicate known system behavior, and to test its ability to predict unknown behavior. Our current confidence calculation does not differentiate between these two needs, although we propose including it in our future work along with additional conceptual model validation.

Many extensions to DEVS exist, such as Cell-DEVS. This approach for quantifying confidence levels can be applied to these DEVS extensions by only extending the types of expected behaviors to validate.

In large models it is generally understood that 100% validation is impossible due to time or budget constraints, or a lack of required data. Our framework not only provides a mechanism for calculating confidence in the model based on validation performed, but it also provides a mechanism for making tough decisions about what validation will be most beneficial on the model to maximize confidence given constraints.

In future work we will examine how to calculate confidence such that not all applicable validation techniques must be applied to have full confidence in a particular behavior. We will also perform further experimental validation of the confidence calculation approach on larger simulation models, and extend the framework to other modeling formalisms with a corresponding simulation model.

### Acknowledgments

The authors thank the reviewers for their helpful feedback. They also thank the Clare Boothe Luce Foundation and Loyola University Maryland for funding this project. The statements within belong entirely to the authors.

## REFERENCES

1. Balci, O. Golden rules of verification, validation, testing, and certification of modeling and simulation applications. *SCS Modeling & Simulation Magazine* (2010).
2. Balci, O. A life cycle for modeling and simulation. *Simulation: Transactions of the Society for Modeling and Simulation 88*, 7 (July 2012), 870–883.
3. Bergero, F., and Kofman, E. Powerdevs: a tool for hybrid system modeling and real-time simulation. *Simulation* (2010).
4. Chillarege, R., Bhandari, I., Chaar, J., Halliday, M., Moebus, D., Ray, B., and Wong, M. Orthogonal defect classification-a concept for in-process measurements. *IEEE Transactions on Software Engineering 18*, 11 (Nov 1993), 943–956.
5. Cho, H., and Cho, Y. Devs-c++ reference guide, 1997.
6. Elele, J. N., Hall, D. H., Farid, A. R., Turner, D. J., Davis, M. E., and Keyser, D. R. Lessons learned from independent verification and validation of models and simulation (wip). In *SpringSim TMS DEVS* (2014).
7. Ferenc, B., and Hogrefe, D. The ccitt-specification and description language sdl. *Computer Networks and ISDN Systems 16* (1989), 311–341.
8. Forrester, J., and Senge, P. Tests for building confidence in system dynamics models. In *TIMS Studies in the Management Sciences*, vol. 14. North-Holland, 1980, 209–228.
9. Gehlot, V., and Nigro, C. An introduction to systems modeling and simulations using colored petri nets. In *Proceedings of the 2010 Winter Simulation Conference* (2010).
10. Hollmann, D. A., Cristia, M., and Frydman, C. A family of simulation criteria to guide devs models validation rigorously, systematically and semi-automatically. *Simulation Modelling Practice and Theory* (2014).
11. Huang, E., Randeep, R., and McGinnis, L. System and simulation modeling using sysml. In *Proceeddings of the 2007 Winter Simulation Conference* (199).

12. Labiche, Y., and Wainer, G. Towards the verification and validation of devs models. In *Proceedings of the First Open International Conference on Modeling and Simulation* (2005).

13. Li, X., Vangheluwe, H., Lei, Y., Song, H., and Wang, W. A testing framework for devs formalism implementations. In *Proceedings of the 2011 Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium*, TMS-DEVS '11, Society for Computer Simulation International (San Diego, CA, USA, 2011), 183–188.

14. Olsen, M., and Raunak, M. A framework for simulation validation coverage. In *Proceedings of the 2013 Winter Simulation Conference* (Dec 2013).

15. Popovici, K., and Mosterman, P. J. *Real-Time Simulation Technologies: Principles, Methodlogies and Applications*. CRC Press, 2012.

16. Raunak, M., and Olsen, M. Quantifying validation of discrete event simulation models. In *Proceedings of the 2014 Winter Simulation Conference* (Dec 2014).

17. Raunak, M., and Olsen, M. A survey of validation in health care simulation studies. In *Proceedings of the 2014 Winter Simulation Conference* (Dec 2014).

18. Sargent, R. Verifying and validating simulation models. In *Proceedings of hte 2005 Winter Simulation Conference* (2005).

19. Sargent, R. G. Verification and validation of simulation models. In *Proceedings of the 2010 Winter Simulation Conference* (2010).

20. Tolk, A., Heath, B., Ihrig, M., Padilla, H., Page, E., Suarez, E. D., Szabo, C., Weirich, P., and Yilmaz, L. Epitemology of modeling and simulation. In *Proceedings of the 2013 Winter Simulation Conference* (2013), 1152–1166.

21. Traore, M. K., and Muzy, A. Capturing the dual relationship between simulation models and their context. *Simulation Modelling Practice and Theory 14* (2006), 126–142.

22. Wainer, G. Cd++: A toolkit to develop devs models. *Software - Practice and Experience 32* (2002), 1261–1306.

23. Wainer, G., Frydman, C., and Giambiasi, N. An environment to simulate cellular devs models. In *SCS European Multiconference on Simulation* (1997).

24. Wainer, G. A. Cd++ sample models. **`http://www.sce.carleton.ca/faculty/wainer/wbgraf/doku.php?id=model_samples:start`**. Last Accessed: 11-09-14.

25. Wainer, G. A., Al-Zoubi, K., Hill, D. R., Mittal, S., and Martin, J. An introduction to devs standardization. In *Discrete-Event Modeling and Simulation: Theory and Applications*, G. Wainer and P. Mosterman, Eds. CRC Press, 2011.

26. Wang, Z. Towards a measurement tool for verfication and validation of simulation models. In *Proceedings of the 2011 Winter Simulation Conference* (2011).

27. Wang, Z. Selecting verification and validation techniques for simulation projects: A planning and tailoring strategy. In *Proceedings of the 2013 Winter Simulation Conference* (2013).

28. Zeigler, B., Kim, T., and Praehofer, H. *Theory of Modeling and Simulation*. Academic Press, 2000.

29. Zengin, A., and Ozturk, M. M. Formal verification and validation with devs-suite: {OSPF} case study. *Simulation Modelling Practice and Theory 29*, 0 (2012), 193 – 206.

30. Zhang, M., Zeigler, B., and Hu, X. Formalized approach for the design of real-time distributed computer system. In *Real-Time Simulation Technologies: Principles, Methodologies, and Applications*, K. Popvici, Ed. CRC Press, 2012.