Proceedings of the 2019 Winter Simulation Conference
N. Mustafee, K.-H.G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y-J. Son, eds.

VERIFYING AND VALIDATING A PARALLEL DEVS SIMULATION MODEL: EXAMPLE OF A POWER GRID MODEL

Ange-Lionel Toba

Idaho National Laboratory
Engineering Research Office Building
2525 Fremont Ave.
Idaho Falls, ID 83415, USA

Christopher J. Lynch

Virginia Modeling, Analysis, and Simulation
Center, Old Dominion University
1030 University Blvd.
Suffolk, VA 23435, USA

Mamadou Seck

Amtrak
60 Massachusetts Ave NE
Washington, District of Columbia 20002, USA

ABSTRACT

Verification and validation (V&V) are necessary for determining the correctness and reliability of models and simulations as well as for checking the salient properties and functional requirements of the system. V&V ensures that each step followed in the process of building the model yields acceptable outputs, and that the documented operations and activities that a system must be able to perform are actually performed. This paper describes a V&V process for a Discrete Event System Specification (DEVS) simulation model and presents approaches used to examine model validity. The DEVS model used as an example is the existing *Spark*! model for simulating power grid operations and energy planning.

1 INTRODUCTION

The spread of disease has and continues to threaten humanity. In the early twentieth century, the SIR (susceptible-infected-removed) model was developed by Ross (1910) to project the spread of a disease as a result of contact between infected and susceptible population. The model became very popular in epidemiology (Hethcote, 1989), with the emergence of infectious diseases like Ebola (Astacio et al., 1996), and AIDS (Mollison, 1995). Another model, World 3, introduced by Meadows, Meadows, Randers, and Behrens (1972) was developed to project the dynamic behavior of population, capital, food, non-renewable resources, and pollution and became popular due to increasing interest in environmental degradation (Janssen & De Vries, 1999). Despites their notoriety and their use as building blocks for later global models, these models were criticized over concerns over the extent to which they can be considered valid or trustworthy for decision making, learning, and seeking truth (Nordhaus, 1973; Weiss, 2013).

These concerns extend to all models, far beyond the areas of public health or environmental degradation. The role, for instance, of electricity in our lives is of tremendous importance. Electricity powers our appliances, homes, facilities, cars, medical equipment, and provides access to information and services. Losing access to this resource would have a debilitating effect on society wellbeing. Energy models assist in analyzing power grid infrastructure systems to help inform decisions in managing, monitoring, and controlling all operations along the supply chain.

These models, along with all safety critical systems, deserve a strong Verification & Validation (V&V) focus (Bowen & Stavridou, 1993; Glasow & Pace, 1999). The goal is to increase confidence in the credible and reliable utilization of simulation models and their results and to transfer this confidence to the simulation models' users (Sargent, 2004). V&V techniques range in level of difficulty and mathematical rigor but are collectively oriented towards collecting evidence of a model's correctness for a given purpose or specified conditions, and helps to provide evidence of sufficient accuracy in the simulation results (Balci, 1998; Padilla, Diallo, Lynch, & Gore, 2018). Models are used to understand real systems' behaviors and make predictions; as a result, V&V processes assess model fidelity to requirements and predictive accuracy.

Energy system models have become the main supporting tool for energy policy (Jebaraj & Iniyan, 2006), enabling decision makers to deal with capacity limitations, unexpected changes, maintenance and investment decisions, as well as other challenges, not always foreseeable. The issue of grid balancing, considering increasing renewable penetration is one pressing in the power sector. How to make sure power is available and accessible whenever and wherever needed? Using models is an appropriate approach to answer such question, given their ability to provide insights into how energy systems function and may evolve in the years to come.

In this paper, we present V&V methods used to validate a model developed using the DEVS formalism. Main strength of DEVS is its rigorous formal definition, allowing modular description of models which can be integrated using a hierarchical approach (Bernard P Zeigler, Praehofer, & Kim, 2000). In addition, the precise specification allows for models in other formalisms to be described in DEVS (Van Tendeloo & Vangheluwe, 2017). We used Parallel DEVS (PDEVS), a parallel, hierarchical, modular, modeling formalism (Chow & Zeigler, 1994) and an extension of Classic DEVS (Bernard P. Zeigler, 1984). It enables parallel execution, faster execution thanks to additional optimizers, and is now becoming the default DEVS formalism implemented (Van Tendeloo & Vangheluwe, 2016), which justifies our choice for this research.

V&V for DEVS models is an open research area with opportunities for application, although some efforts have been made in previous studies. In their work, Labiche and Wainer (2005) discuss research paths in the field of DEVS modeling V&V by means of testing. For them, it is critical to identify the model and associated criteria to be used, the specifics of the testing infrastructure, and mechanisms to (1) compute the expected value of a test case execution, and (2) compare actual results with expected results. Saadawi and Wainer (2009)'s methodology looks to offer improved correctness checking reliability of the actual code executing in the real-time system, by using Timed Automata (TA) to verify DEVS models. The objective is for DEVS models to be transformed to semantically equivalent TA models, ensuring that original structure and behavior are maintained. A similar approach is used by Inostrosa-Psijas, Gil-Costa, Wainer, and Marín (2016), who also apply equivalency between DEVS and TA model. Soremekun and Traore (2013) develop a fidelity framework, using the Process Analysis Toolkit (PAT), the Communicating Sequential Processes (CSP) specification of the model as well as the Real Time System (RTS) specification, to enable an effective application of V&V. Trojet and Berradia (2015)'s work consists in improving the verification of simulation models through the integration of formal methods based on Z formalism. This integration takes place by (1) the transformation of DEVS models into an equivalent Z specification and (2) the verification of the consistency of DEVS models on the resulting Z specification. The purpose is to detect errors before running the simulation. Gholami and Sarjoughian (2017) introduce a model-checker integrated into the DEVS-Suite simulator for verifying constrained DEVS models using the DEVS simulator protocol. This method consists of state configurations, input port configurations, discretized time for external inputs, and verification protocol, constrains state space explosion, and increases reliability.

The V&V approach used in our study is implemented on PythonPDEVS, a parallel DEVS simulation modelling language (Van Tendeloo & Vangheluwe, 2015). We use Python as it is a high-level programming language that is machine independent and convenient for managing complex tasks (Python, 2018). Unlike the previous studies mentioned above, using this platform removes the need for transformation or formalism mapping. Since no unified semantic foundation exists, the framework mapping may vary from one configuration to another and eventually affects the consistency of verification testing. Beyond the use of

this formalism, we highlight the importance of subject matter experts in the V&V process to vouch for accuracy in the real system representation and model behavioral trends.

In Section 2 we describe the simulation formalism and platform used in this study. Section 3 provides the V&V methods applied to the simulation. Section 4 provides our conclusions.

2 DEVS MODELING AND SIMULATION FRAMEWORK

DEVS is a mathematical modeling formalism for describing (discrete and continuous) dynamical systems. It is appropriate for event-driven models, that is, systems whose states change anytime an event takes place. Systems are described as a tuple M = ($T, X, Y, Q, q_0, \delta_{ext}, \delta_{int}, \lambda$) (Bernard P. Zeigler, 1976). DEVS generic system-theoretic concepts and mathematical formulation provide a basis for describing component models via structure and behavior specifications. This framework lends itself to object-based abstraction (Zengin, Köklükaya, & Ekiz, 2010), with objects being modeled as an *atomic* model and relationships between them being modeled as *coupled* models. The *atomic* models can be used as building elements, part of a larger *coupled* model (Solcány, 2008). This model design describes its components (either atomic models or other coupled models) and specifies their interconnections. Unlike *atomic* models, *coupled* models do not require functions to specify their behaviors; rather, they specify couplings between their components.

Both atomic and coupled models have an interface consisting of *input* (X) and *output* (Y) ports to communicate with other. In atomic models, every *state* (Q) in the model is associated with a *time advance* (T) function, which determines the duration during which the state remains unchanged. Once the time assigned to the state has passed, an *internal transition function* (δ_{int}: $Q \rightarrow Q$) is fired and an internal transition is triggered, producing a local state change (δ_{int} (s) = s'). At that moment, the model execution results are spread through the model's output ports by activating an *output function* (λ). Input external events (events received from other models) are collected in the input ports. An external transition function (δ_{ext}: $Q \times X \rightarrow Q$) specifies how to react to those inputs, using the current state (S), the elapsed time since the last event (e) and the input value (X) (δ_{ext} ((s, *e*), *x*) = s').

PDEVS is well suited for formally describing concurrent processing and distributed large-scale systems, such as a power grid system. We consider the power grid system to be a network of components for the supply, delivery, and consumption of electricity (Adegbululgbe et al., 2007). It is composed of independent and collectively synergistic generation stations (including storage units), which supply transmission lines to transport electricity from supply sources to demand centers, and population (load), which consumes electricity (Kaplan, 2009). These components are represented as *atomic* DEVS models, *load, generator*, and *transmissionLines*. Electricity is an indispensable commodity, used in virtually all daily activities. It is critical to plan to make sure that electricity needs are always met. Planning generally consists of two activities both conducted in the least expensive manner, unit commitment and generation dispatch (DOE, 2005). Unit commitment is about scheduling generating units for use ahead of time, while generation dispatch handles dispatch generation in real time, performed by a Balancing Authority (FERC, 2005). In the model, we consider these two activities to be performed by two components modeled as atomic DEVS models, *unitCommitment* and *dispatcher*. The dispatch operator decides, based on the quantity of power available and current demand, whether to import or export. The idea of electricity import/export implies the notion of market dynamics, where electricity is exchanged between areas, called balancing area (BA). BA is a geographical location with a specific generation fleet, load profile, and a dispatch operator. BA are modeled as *coupled* DEVS model. Finally, the power grid is a network of BA connected to each other, exchanging electricity if needed. BA in excess (more generation than demand) export while BA in deficit (more demand than generation) import. Evidently, BAs can only trade if there is an existing transmission line between them. Therefore, the whole grid is modeled as a *coupled* DEVS model.

3 VERIFICATION AND VALIDATION

In performing V&V, we ensure that the model and its components are close to the real system. In our case, the power grid is the real system and we present the steps taken to validate our model. *Spark*! describes the

behavior of the power grid. The model was developed from scratch, on PythonPDEVS platform, using DEVS formalism and object-oriented technology (Toba & Seck, 2019). Each model component is described as a Python class, and instances of these classes are created after parameterization. For example, *Loads* are differentiated by name and type of power demand (i.e., residential, industrial or commercial) and BA zone to which the demand belongs. Figure 1 provides a glimpse of the Python® script of the *Load* class.

```
#The class is characterized by the name and quantity of demand
def __init__(self, zone, name=None):

    # Always call parent class' constructor FIRST:
    AtomicDEVS.__init__(self, name)

    #Name of input/output ports used
    #Output port of the class, sending the dmd to the dispatcher
    self.DemandToDispatcher = self.addOutPort(name="DemandToDispatcher")

    #Input port of the class, receiving the proportion of demand met from the dispatcher
    self.SupplyFromDispatcher = self.addInPort(name="SupplyFromDispatcher")

def extTransition(self, inputs):
#External Transition Function
    #Message from the Dispatcher, specifying the amount of demand met
    supplyFromDispatcher = inputs.get(self.SupplyFromDispatcher)
    if self.state == "wait" and supplyFromDispatcher != None:
        self.state = "advance"
    else:
        print "ERROR in LOAD Atomic model EXTERNAL TRANSITION FUNCTION, ZONE : __ %s" %self.zone
    return self.state

def intTransition(self):
#Internal Transition Function
    self.total_time += self.timeAdvance()
    if self.state == "idle":
        self.state = "request"
    elif self.state == "request":
        self.state = "wait"
    elif self.state == "advance":
        self.state = "request"
    else:
        print "ERROR in LOAD Atomic model INTERNAL TRANSITION FUNCTION"
    return self.state

def outputFnc(self):
# Output Funtion, specifying the output to send to the dispatcher
    if  self.state == "request" :
        return {self.DemandToDispatcher: [copy.copy(self.load)]}
    else:
        return {}

def timeAdvance(self):
# Time advanse function
    if self.state == "advance":
        return 1
    elif self.state == "idle" or self.state == "request":
        return 0
    elif self.state == "wait":
        return INFINITY
    else:
        raise DEVSException(\
            "unknown state <%s> in LOAD Atomic model ADVANCE FUNCTION"\
            % self.state)
```

Figure 1: Atomic DEVS model implementation via PythonPDEVS of the Load.

The objective of the model is perform energy planning of large-scale grids by analyzing expansion plans on a long term while also performing short term day-to-day activities of the system. V&V objective is to examine how representative the model is. Model credibility is a function of how well the system is represented, or how much fidelity the model shows, with regards to the study objectives. In our case, we test the model with inputs including weather information for renewable sources, power plants technical constraints, load profiles, generation technology costs, and with grids of various scales and locations while accounting for all seasons throughout the year. The simulation's domain of applicability includes conditions of high scalability, high renewable penetration, and high variability, in terms of load changes and renewable intermittence. Though it is infeasible to conduct exhaustive testing, we consider a combination of feasible conditions within the context of our study.

3.1 Verification

Verification consists in determining whether the implemented model is consistent with its specification (Sanders, 1996). The specification describes the relevant variables, model components, behaviors, and relationships between components. Verification checks for transformational accuracy in ensuring that the transformation of the conceptual model into an executable one are accurate enough with respect to the model' s design specification (Balci, 1998). Conceptual models generally provide informal depictions of the real systems, containing information related to the overall functionality of that system (McKenzie, 2010). This model represents concepts or entities and relationships between them. It brings clarification and consistency to the meaning of various terms and concepts and facilitates implementation. Doing verification thus entails making sure all concepts and whole functionality modeled can be translated in an executable model. An executable model is a working prototype, one that can be simulated and results of which can be observed and analyzed.

For verification, we use the Control Analysis method (Sargent, 2004) to check that state transitions take place as expected. This technique is conducted to check state transitions and conditions prompting them. This technique requires the identification of all possible states the model execution goes through, the specification of state transitions, and how they match with the system requirements. Testing each DEVS model, we pay close attention to the (1) sequence and nature of tested inputs; (2) the time between these inputs; (3) the firing of the output functions; (4) the state transitions; and (5) the recipient of the messages sent by the model under test. For this purpose, we check for *reachability* and *deadlock.*

Checking for *reachability* ensures that all states are achievable (Inostrosa-Psijas et al., 2016). We must ensure that, given a set of transition functions, states of a system are reachable from a given initial state of that system. To verify this property, we specify a set of *input* (X), set of *output* (Y), set of *state* (Q), *time advance* (T) function, *internal transition function* (δ_{int}: $Q \rightarrow Q$), external *transition function* (δ_{ext}: $Q \times X \rightarrow Q$), and *output function* (λ), for all atomic DEVS model components. We also define the input and output ports of each atomic DEVS model, ensuring that the right inputs are received, and from the right atomic model. Figure 1 shows the model implementation of the *Load* component. The command "PRINT" is added to track errors and check if states transition as they should, given input, output and conditions attached. Figure 2 displays implementation output of state transition of the Load atomic DEVS model, representing the residential demand in a BA, we call "Mali".

```
_ Current Time:       0.00 _____________________________________

INITIAL CONDITIONS in model <interZone.mali.residential>
        Initial State: idle
        Next scheduled internal transition at time 0.00

INTERNAL TRANSITION in model <interZone.mali.residential>
        New State: request
        Output Port Configuration:
                port <DemandToDispatcher>:
        Next scheduled internal transition at time 0.00

 INTERNAL TRANSITION in model <interZone.mali.residential>
         New State: wait
         Output Port Configuration:
                 port <DemandToDispatcher>:
                         {'name': u'residential', 'quantity': 86.37179605274245}
         Next scheduled internal transition at time inf

EXTERNAL TRANSITION in model <interZone.mali.residential>
        Input Port Configuration:
                port <SupplyFromDispatcher>:
                        {u'residential':  16.5834645903095, u'industrial': 119.2017461724335, u'commercial': 75.21296240761653}
        New State: advance
        Next scheduled internal transition at time 1.00

_ Current Time:       1.00 _____________________________________

 INTERNAL TRANSITION in model <interZone.mali.residential>
         New State: request
         Output Port Configuration:
                 port <DemandToDispatcher>:
         Next scheduled internal transition at time 1.00
```

Figure 2: Trace output of State transition of residential type *Load* atomic DEVS model, in Mali.

In initial conditions, at time 0, the state is *idle*. An internal transition is scheduled at the same time unit, during which the model evolves to state *request*. In this state, the model is expected to request the demand needed (residential). The output function is thus triggered, sending a message to the *dispatcher* atomic DEVS model, specifying this need, through output port "DemandToDispatcher". The next internal event happens then, with state transitioning from *request* to *wait*. The model stays in this state for INFINITY, meaning, no state transition can happen without an (external) input from another atomic DEVS model. The input must be from Dispatcher atomic DEVS model, specifying the amount of demand met, via input port "SupplyFromDispatcher". Once such an input is received, the external transition function is triggered, prompting a transition to state *advance* (this state is added just to increment simulation time. One thing to clarify is that all these transitions occur, though sequentially, at the same simulation time. This is the reason why the time advance of the state *advance* is placed at 1, to allow the model to jump to the next simulation time). The next event is an internal transition, with the model transitioning to state *request*. The cycle repeats and this *load* atomic model behavior mimics the actual process of hourly regional/regional electricity demand presented to the balancing authority.

Check for *deadlock*: This property ensures that models executions do not stop at some point without being able to progress (Saadawi & Wainer, 2009). Because we devise the power grid as a multi-agent system, we need to account for state transitions concurrency and synchronization. This test is critical as it helps to ensure *reachability*. If no deadlocks are found in the DEVS model, is it an indication that all states would eventually be reached, transition functions would be triggered correctly and timely, and output functions would fire the appropriate messages, from the sending DEVS model to the designated receiving DEVS model. Figure 3 displays the state trajectories of 3 of the atomic DEVS models considered, namely Load, Generator and Dispatcher. The states transitions are synchronized in such a way to prevent deadlock.

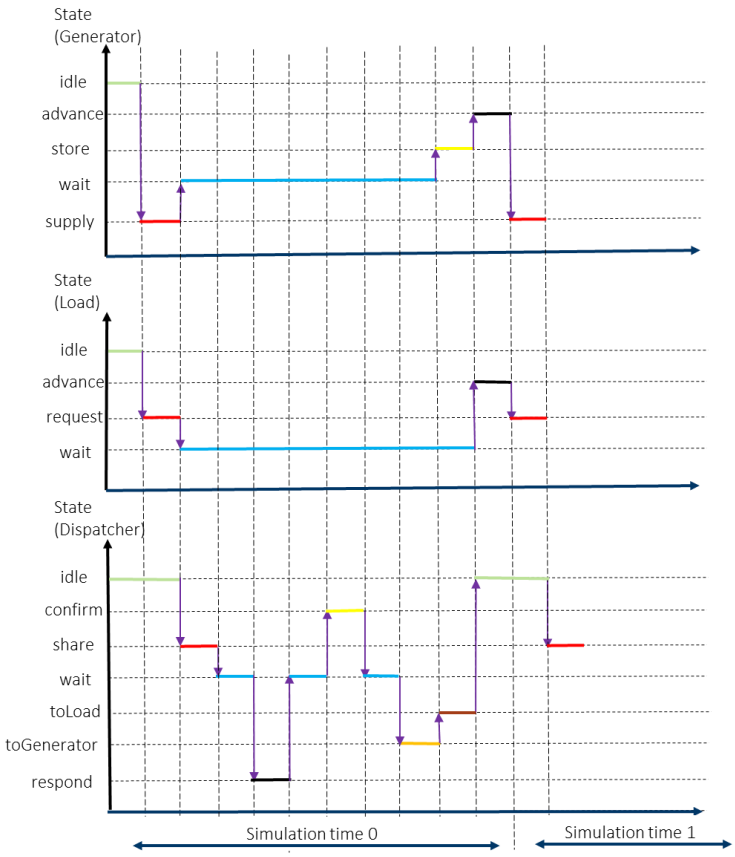

Figure 3: State trajectory of *Generator*, *Load*, and *Dispatcher* DEVS model.

Models *Load* and *Generator* concurrently transition to states *request* and *supply*, respectively. At state *supply*, the output function is triggered, sending power quantity availability information to the Dispatcher model. At this point, the internal transition function is triggered, and state *supply* changes to *wait*. The model waits for the dispatcher model to indicate the electricity quantity actually in need. Once received, the external transition function is triggered, prompting state change from *wait* to *store*. When state is *store*, the model stores the surplus, if it exists. The state is transitioned to *advance* right after the output function is fired. In the meantime, in the Load atomic DEVS model, the output function was also triggered at state *request*, sending demand information to the Dispatcher model. At this point, the internal transition function is triggered, and state *request* transitions to *wait*. The model waits for the dispatcher model to indicate the amount of demand that could actually be met. Once received, the external transition function is triggered, prompting state change from *wait* to *advance*.

The initial state of the Dispatcher model is *idle*, which transitions to *share* only when the model receives input from both Load and Generator models. Once received, the model evolves from state *share* to *wait*. At this point all DEVS Dispatcher models initiate electricity imports/exports exchanges, make bids, identify least expensive transaction partners, and sell/buy electricity. During these transactions, states transition to *respond*, and later *confirm*. Once transactions are over, the Dispatcher's state changes to *toGenerator*. The output function is triggered, sending message to Generator model. After this message sent, the state transitions to *toLoad*, triggering the output function to fire a message to the Load model. From that point, the model moves back to its initial state *idle*, and the cycle repeats at the next simulation time.

3.2 Validation

Validation consists in determining the level to which the model is an accurate representation of the real system (Sanders, 1996). It checks for representational accuracy, ensuring that the representation of the real system in a conceptual one, and the results produced by its execution are accurate enough, with respect to its intended uses. According to (Balci, 1998), questions to be answered during validation include:

- Does the conceptual model correctly represent the real system?
- How close are the results produced by the executable model to the behavior of the real system?

The main function of the model is long term energy planning, while ensuring reliability of the system, in the short term. Two critical operations in energy planning are the unit commitment and economic dispatch phases. The unit commitment phase consists in scheduling the ON/OFF times of plants, ahead of time, in such a way to minimize the costs associated with hourly generation (Jabr, 2013). The economic dispatch phase consists in monitoring load, generation and ensuring balance of supply and load in real time, by ordering generators to increase/decrease their output based on system security needs, and in the least expensive manner (Alvarado & Oren, 2002). For the purpose of our study, we deem our model valid if it performs those operations with satisfactory accuracy. For validation, we use 3 methods, namely functional Testing, face validation, and graphical comparison (Balci, 1998; Sargent, 1996, 2004).

Functional testing validates the input-output transformation of the model. The emphasis is placed, not on the mechanisms in models, rather, what is produced as output, given a set of input. In this case, our unit commitment algorithm is used on a benchmark case which has been widely researched in the literature, with 24-hour load profile and a 10, 20, 40, 60, 80, and 100 generating units setting (Kazarlis, Bakirtzis, & Petridis, 1996). We compare the results, in terms of total generation costs derived from generating unit commitment scheduling, with Delarue, Cattrysse, and D'haeseleer (2013)'s EPL (extended priority list) model implementation and also Carrión and Arroyo (2006)'s MILP (Mixed-integer linear programming) implementation (Table 1).

The differences in results may be explained by the heuristic used in our algorithm, which performs plants scheduling based on usage history and current need, without look-ahead mechanism. Units are thus (un)committed to cover demands at the present time (here, we consider hour as time unit), without considering needs at next hour and associated startup/shutdown costs. These differences in results seem to grow as the number of units increase but stabilize.

Table 1 shows that our model offers solutions less than 5% off the optimal one, with 100 units considered. Looking at the difference in costs, which is a about $200,000, we can see a quite significant loss of money. However, given a grid of large scale, our model fares satisfactorily.

Table 1: Total costs of scheduling using the EPL algorithm, the MILP model, and *SPARK!* in US dollars.

Number of Units	EPL Total Cost ($)	MILP Total Cost ($)	Our Method Total Cost ($)	Difference: EPL – Our Method	Difference: MILP – Our Method
10	563,977	563,938	564,638	0.12%	0.12%
20	1,124,481	1,125,721	1,155,768	2.71%	2.60%
40	2,246,926	2,246,243	2,310,738	2.76%	2.79%
60	3,366,240	3,367,262	3,514,932	2.84%	4.20%
80	4,489,342	4,488,560	4,665,359	3.77%	3.79%
100	5,609,109	5,609,210	5,872,037	3.92%	4.48%

Looking at Figure 4, we notice that the computation time in MILP tend to grow linearly with the number of plants considered, while our model offers a less steep slope, with a seemingly constant value. What is lost in accuracy is won in time execution. This is a good advantage over MILP, especially now that power grids are extending in scales. Our model handles better scalability and shows more practicality and usefulness, for the its intended purpose, which gives us good confidence regarding the credibility of our results.

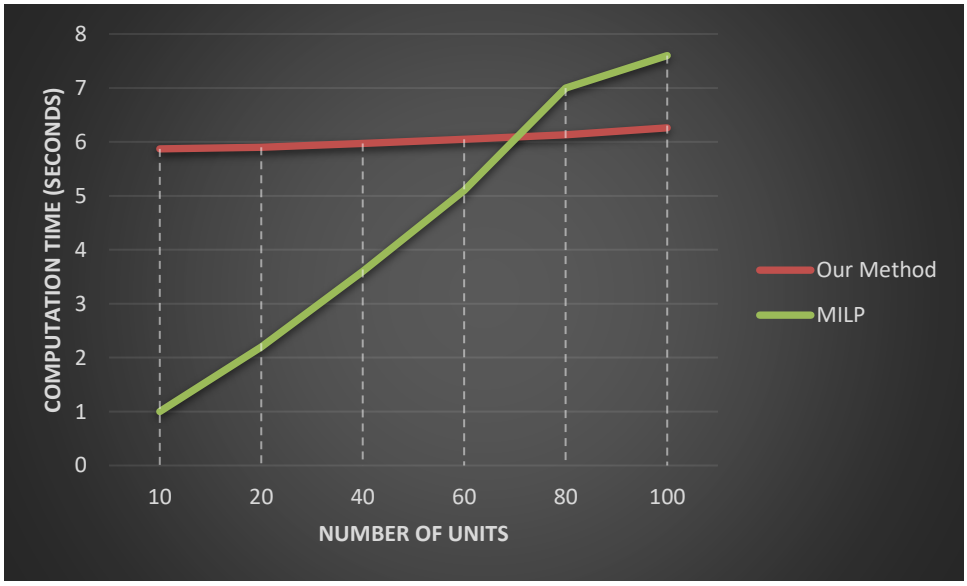

Figure 4: Time difference in computation between our method and MILP in seconds.

Face validation: This technique is conducted to compare model and system behaviors under identical input conditions. This assessment is done by subject matter experts or people knowledgeable about the system under study, judging the reasonableness of the model, its assumptions and its outputs.

The advantage of having users/experts involved in the validation phase is that it increases the model's credibility to the users, and their confidence in the model (Carson & John, 2002). In this case, the economic dispatch phase is tested. The set of experts relied on are professional working for Dominion Energy, an American power and energy company supplying electricity in parts of Virginia and North Carolina and natural gas to parts of West Virginia, Ohio, Pennsylvania, and eastern North Carolina. These experts were

able to provide insights about what a power grid is, how it works at the strategic level. Based on their suggestions, we were able to identify key components of the system, specify the relationships between them, and conceptually design the model. In addition, using data provided by them, we were able to run scenarios and present results, including plants dispatch, usage rates and load balancing, which were thought reasonable under the observed conditions. We also check the sensitivity to model inputs, with changes in generation supply and transmission capacity. We ran the power grid simulation twice, one with a given generation fleet and another with twice this fleet. We observed, as expected, an increase in percentage of demand met in BA in deficit and in shared electricity between BA.

We then check the economic dispatch requirements, testing our dispatch algorithm using trivial examples to make sure that the model behaves accordingly. For example, we considered a grid with two BAs and later three BAs, three generating units each and different load profiles, for 2 cases: one with a transmission network, and the other one with none. We could check (1) that less expensive generating units were dispatched first, before more expensive ones, (2) that power was not exchanged in the case of nonexistent transmission network, (3) that in the case of existing transmission network, the transmission capacity was always respected, and (4) that market dynamics takes place as expected, with bid-based transactions.

Graphical comparison is conducted to compare the graphs of values of model variables over time with the graphs of values of system variables, and check for similarities. We compare the simulation model output behavior to another model output behavior using graphical displays. Figures 5 and 6 display the energy mix by country composing the WAPP obtained in the study, and by our model, respectively.

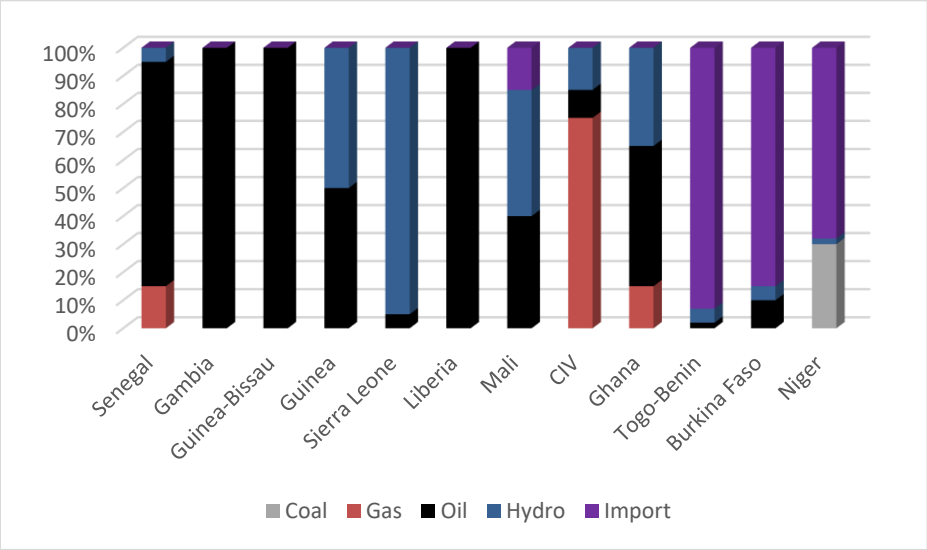

Figure 5: Electricity production shares by country in 2010 from IRENA (2013).

This model, MESSAGE, was used as a planning tool, to help study long term energy planning via various scenarios, and assess the economic, environmental and social implications for the West African Power Pool (WAPP) (IRENA, 2013). We use the data from the study, which reflect the state of affair of the WAPP grid, with existing supply sources, transmission network, costs, and load forecasts, in the year 2010. Based on these two figures, our simulation model can be judged to have sufficient accuracy with respect to the production shares. Results are fairly similar, with very minor differences.

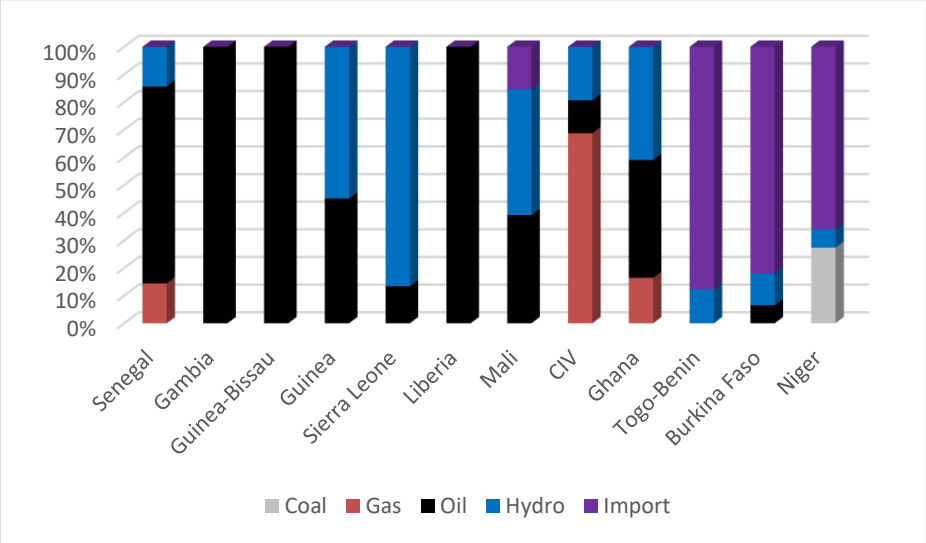

Figure 6: Electricity production shares by country in 2010 from *Spark!*.

4 CONCLUSION

This paper presents the steps followed to verify and validate a DEVS simulation model. *Spark*! a DEVS electricity simulation model is presented as an example. The scope of the paper includes the reasoning, general concepts, and processes for conducting a successful V&V for a DEVS model. Because we use the DEVS modeling formalism, we map all model components activities to states transitions, and capture their behaviors via transition functions. The model is built and each of its components is verified on the PyhonPDEVS platform, with no DEVS model transformation to other semantically equivalent modeling formalism. As we conduct this model verification, several questions had to be answered: Have all states been defined? Can all the states be reached? Does each state respond properly to all possible conditions? For validation, we employ model-to-model techniques and compare our model outputs with outputs of already validated models, considering the same set of input conditions. These outputs show results of unit commitment and economic dispatch operations, the two main real-world system operations constituting energy planning.

This paper also demonstrates the usefulness of users and experts in V&V and presents a case with a power grid simulation model. Essentially using energy experts' intuition to validate the model increase its credibility, as expertise is shared with respect to the real system, rather than with respect to the model. Though subjective to some extent, we show that model validity is to be evaluated for its condition of being useful, that is, scalable, and offering good performance under high variability. In testing our model, we sought to increase our confidence in model credibility as much as constrained by the study objectives rather than trying to test the model completely.

REFERENCES

Adegbululgbe, A., Fenhann, J., Konstantinaviciute, I., Moomaw, W., Nimir, H. B., Schlamadinger, B., . . . Zhang, X. (2007). Energy Supply. In H. Larsen & J. R. Moreira (Eds.), *Climate Change 2007: Mitigation of Climate Change*.

Alvarado, F., & Oren, S. (2002). Transmission system operation and interconnection. *National transmission grid study–Issue papers*, A1-A35.

Astacio, J., Briere, D., Guillen, M., Martinez, J., Rodriguez, F., & Valenzuela-Campos, N. (1996). Mathematical models to study the outbreaks of Ebola.

Balci, O. (1998). Verification, Calibration, and Testing. In J. Banks (Ed.), *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice* (1st ed., pp. 335-393). New York, NY: John Wiley and Sons, Inc.

Bowen, J., & Stavridou, V. (1993). Safety-Critical Systems, Formal Methods and Standards. *Software Engineering Journal, 8*(4), 189-209. doi:10.1049/sej.1993.0025

Carrión, M., & Arroyo, J. M. (2006). A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem. *IEEE Transactions on Power Systems, 21*(3), 1371-1378.

Carson, I., & John, S. (2002). *Verification validation: model verification and validation.* Paper presented at the Proceedings of the 34th conference on Winter simulation: exploring new frontiers.

Chow, A. C. H., & Zeigler, B. P. (1994). *Parallel DEVS: A parallel, hierarchical, modular modeling formalism.* Paper presented at the Simulation Conference Proceedings.

Delarue, E., Cattrysse, D., & D'haeseleer, W. (2013). Enhanced priority list unit commitment method for power systems with a high share of renewables. *Electric power systems research, 105*, 115-123.

DOE. (2005). *The Value of Economic Dispatch: A Report to Congress Pursuant to Section 1234 Of The Energy Policy Act Of 2005*. Retrieved from https://energy.gov/sites/prod/files/oeprod/DocumentsandMedia/value.pdf

FERC. (2005). *Economic Dispatch: Concepts, Practices and Issues*. Retrieved from https://www.ferc.gov/CalendarFiles/20051110172953-FERC%20Staff%20Presentation.pdf

Gholami, S., & Sarjoughian, H. S. (2017). *Modeling and verification of network-on-chip using constrained-DEVS.* Paper presented at the Proceedings of the Symposium on Theory of Modeling & Simulation.

Glasow, P., & Pace, D. K. (1999). *Simulation Validation (SIMVAL) 1999, Making VV&A Effective and Affordable Mini-Symposium and Workshop*. Retrieved from

Hethcote, H. W. (1989). Three basic epidemiological models. In *Applied mathematical ecology* (pp. 119-144): Springer.

Inostrosa-Psijas, A., Gil-Costa, V., Wainer, G., & Marín, M. (2016). *Formal verification of DEVS simulation: web search engine model case study.* Paper presented at the Proceedings of the Summer Computer Simulation Conference.

IRENA. (2013). *West African Power Pool: Planning and Prospects for Renewable Energy*. Retrieved from https://www.irena.org/documentdownloads/publications/wapp.pdf

Jabr, R. (2013). Rank-constrained semidefinite program for unit commitment. *International Journal of Electrical Power & Energy Systems, 47*, 13-20.

Janssen, M. A., & De Vries, H. J. M. (1999). Global modelling: Managing uncertainty, complexity and incomplete information. In *Validation of Simulation Models* (pp. 45-69). Amsterdam, The Netherlands: SISWO.

Jebaraj, S., & Iniyan, S. (2006). A review of energy models. *Renewable and Sustainable Energy Reviews, 10*(4), 281-311.

Kaplan, S. M. (2009). *Electrical Power Transmission: Background and Policy Issues*. Retrieved from

Kazarlis, S. A., Bakirtzis, A., & Petridis, V. (1996). A genetic algorithm solution to the unit commitment problem. *IEEE Transactions on Power Systems, 11*(1), 83-92.

Labiche, Y., & Wainer, G. (2005). *Towards the verification and validation of DEVS models.* Paper presented at the in Proceedings of 1st Open International Conference on Modeling & Simulation.

McKenzie, F. D. (2010). Systems modeling: analysis and operations research. *Modeling and Simulation Fundamentals: Theoretical Underpinnings and Practical Domains*, 147-180.

Meadows, D., Meadows, D., Randers, J., & Behrens, W. (1972). The Limits to Growth. New York: Universe book. *Cerca con Google*.

Mollison, D. (1995). *Epidemic models: Their structure and relation to data* (Vol. 5). Cambridge, UK: Cambridge University Press.

Nordhaus, W. D. (1973). World dynamics: measurement without data. *The Economic Journal, 83*(332), 1156-1183.

Padilla, J. J., Diallo, S. Y., Lynch, C. J., & Gore, R. (2018). Observations on the Practice and Profession of Modelling and Simulation: A Survey Approach. *Simulation: Transactions of the Society for Modeling and Simulation International, 94*(6), 493-506. doi:10.1177/0037549717737159

Python. (2018). What is Python? Executive Summary. Retrieved from https://www.python.org/doc/essays/blurb/

Ross, R. (1910). *The prevention of malaria* New York, NY: Dutton.

Saadawi, H., & Wainer, G. (2009). *Verification of real-time DEVS models.* Paper presented at the Proceedings of the 2009 Spring Simulation Multiconference.

Sanders, P. (1996). *DoD modeling and simulation (M&S) verification, validation, and accreditation (VV&A)*. Retrieved from

Sargent, R. G. (1996, Dec 8-11 1996). *Some subjective validation methods using graphical displays of data.* Paper presented at the Proceedings of the 1996 Winter Simulation Conference, Coronado, CA.

Sargent, R. G. (2004). *Validation and verification of simulation models.* Paper presented at the Proceedings of the 36th conference on Winter simulation.

Solcány, V. (2008). *Simulation Algorithms for DEVS Models*. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.302.1894&rep=rep1&type=pdf

Soremekun, O. E., & Traore, K. M. (2013). *Formal verification and validation of DEVS simulation models.* Paper presented at the AFRICON, 2013.

Toba, A. L., & Seck, M. (2019). Spark!: an integrated resource planning and dispatch tool for power grid modelling. *International journal of System of Systems Engineering, 9*(1), 44-61.

Trojet, W., & Berradia, T. (2015). System Reliability using Simulation Models and Formal Methods. *System, 132*(17).

Van Tendeloo, Y., & Vangheluwe, H. (2015). *PythonPDEVS: a distributed Parallel DEVS simulator.* Paper presented at the Proceedings of the Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium.

Van Tendeloo, Y., & Vangheluwe, H. (2016). *An overview of PythonPDEVS.* Paper presented at the Collectif Workshop RED, editor, JDF.

Van Tendeloo, Y., & Vangheluwe, H. (2017). *Introduction to Parallel DEVS Modelling and Simulation* Retrieved from http://msdl.cs.mcgill.ca/people/yentl/papers/2017-DEVSTutorial.pdf

Weiss, H. H. (2013). The SIR model and the foundations of public health. *Materials matematics*, 0001-0017.

Zeigler, B. P. (1976). *Theory of Modelling and Simulation*. New York, NY: Wiley.

Zeigler, B. P. (1984). *Multifacetted modelling and discrete event simulation*: Academic Press Professional, Inc.

Zeigler, B. P., Praehofer, H., & Kim, T. G. (2000). *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*: Academic press.

Zengin, A., Köklükaya, E., & Ekiz, H. (2010). *Verification And Validation Of The Devs Models*. Retrieved from https://core.ac.uk/download/pdf/153447277.pdf

AUTHOR BIOGRAPHIES

ANGE-LIONEL TOBA is a researcher at the Idaho National Laboratory, in the Systems Analysis & Engineering department. He earned his Ph.D. in Engineering Management & Systems Engineering from Old Dominion University. His research interests include Modeling and Simulation theory and Critical infrastructure systems engineering and protection. His email address is danhoangelionel.toba@inl.gov.

CHRISTOPHER J. LYNCH is a Senior Project Scientist at the Virginia Modeling, Analysis, and Simulation Center (VMASC) at Old Dominion University (ODU). He received his M.S. in Modeling and Simulation (M&S) from ODU in 2012 and a B.S. in Electrical Engineering from ODU in 2011. He is a Ph.D. candidate in M&S at ODU with a focus on using visual and audial feedback to verify simulations. His email address is cjlynch@odu.edu. His website is https://www.odu.edu/directory/people/c/clync008.

MAMADOU SECK is the director of Operations Research at Amtrack. He was a Professor at Delft University of Technology, and also at Old Dominion University. His research interests include Modeling and Simulation formalisms, dynamic data-driven simulation, human behavior simulation, and agent directed simulation. His email address is mseck@odu.edu.