Verification And Validation Of The Devs Models

Ahmet ZENGIN
Department of Computer Science Education,
Sakarya University, Serdivan, Sakarya, Turkey,
azengin@sakarya.edu.tr

Etem KÖKLÜKAYA
Department of Electrical-Electronics Engineering
Sakarya University, Serdivan, Sakarya, Turkey,
ekaya@sakarya.edu.tr

Hüseyin EKİZ
Department of Computer Science Education,
Sakarya University, Serdivan, Sakarya, Turkey,
ekiz@sakarya.edu.tr

Abstract : Simulation remains attractive for teaching, training and performance analysis of computer networks. This paper presents a robust simulation environment targeted for teaching and learning the complex dynamics of computer networks. The general-purpose DEVS-Suite simulator supports animation with I/O and state trajectories of computer network models developed using parallel DEVS modeling approach. The simulator offers high-level model abstraction as compared with simulators such as ns-2. The combined capabilities afforded by the robust DEVS-Suite simulator assists in understanding the fundamentals of computer network topologies and the logics of communication protocols. This newly developed DEVS-Suite offers an expressive, yet relatively simple to use, simulation environment for students and educators to develop and experiment with computer network models. The paper concludes with observations on the proposed modeling approach and outline some evaluations.

In this paper we discuss verification and validation of DEVS simulation models. Four different approaches to deciding model validity are described; two different paradigms that relate verification and validation to the model development process are presented; various validation techniques are defined; conceptual model validity, model verification, operational validity, and data validity are discussed; a way to document results is given; a recommended procedure for model validation is presented; and model accreditation is briefly discussed.

Keywords: DEVS, DEVS-Suite, Computer Networks, Education, OSPF, ns-2.

I. Introduction

DEVS-Suite is a general-purpose, discrete event simulation environment which supports visualization and tracking capabilities [1], [2]. This is the new generation of the DEVSJAVA simulator [3] based on DEVS formalism [4]. This simulator also supports variable structure modeling [5]. The DEVS-Suite user-interface provides a consistent, efficient, integrated hierarchical component-based representation of models with run-time I/O and state trajectories and tabular data visualization. The OSPF models developed on top of DEVS-Suite is the result of using networking theory as well as software engineering principles. Particular attention is paid to reliability and maintainability in view of the ns-2 simulator. With the developed OSPF simulator, students can create arbitrary network topologies, experiment with the models, and in particular track the dynamics of the network related to routing. Students can be empowered to learn the network concepts interactively. DEVS-Suite simulator can be run on a personal computer as well as online via DEVS-Suite Web Start [6] which enables e-learning using Java Web Start technology [7].

The beginning stage was to create a model and simulator software that can help with the design of the network. All common components of a packet switched network with atomic node and data link models of various capacity assignments were defined using the parallel DEVS formalism at desirable level of abstraction. DEVS modeling approach supports hierarchical modular model construction and distributed execution, and therefore offers a basis to characterize complex, large-scale systems with atomic and coupled models. Many application areas such as swarm routing [8] and processor architectures [9] have been developed using the DEVS Formalism. In particular, the link state routing protocol OSPF with its behavior is modeled. In order to show accuracy of model execution, a set of experiments for validation were performed and results were compared with the ns-2 simulator. Later, evaluations together with ns-2 comparison were carried out in terms of modeling and educational aspects.

The remainder of this paper is organized as follows: Section II briefly reviews the DEVS, DEVS-Suite and ns-2 simulators and describes the OSPF protocol. Section III defines the developed simulation environment detailing the basic model components. Section IV presents an example that shows learning of the OSPF protocol using the DEVS-Suite simulator. Section V describes the validation of the DEVS-Suite network model using simulation of the OSPF in ns-2. Section VI covers evaluation of the developed framework in terms of modeling and education aspects. Section VII presents conclusions and future work.

II. Validation Of Simulation Models

A. What is validation?

Validation is a degree of which how correct a model represents the behavior of its system counterpart from the perspective of intended use of the model [4]. The degree of representation of the model or abstraction is determined by modeler according to user demands. Whenever modeler and simulation user meets are satisfied, it is considered as model is valid. In other words, model validation must be evaluated for its condition of being useful instead of perfectness of it. Validation is a highly required and an integral part of the entire simulation lifecycle by which model is credible and accredit. Model validation process is intended for building the correct model and it helps to find right one [10]. Simulation validity is related to simulation design and simulation usage purpose. Validation process has to be performed both modeler and simulation user [11].

In order to validate a model, quantitative and qualitative tests can be performed [11] [12]. In DEVS framework, an experimental frame is used to perform validation tests. If behaviors of both model and its system counterpart are within acceptable tolerance, model is said to be valid [4]. Validation tests are various tests rather than single detailed tests by which confidence of the model increase as it passes them. Passing a test doesn't express model is valid; on the other hand failure of any test allows modeler to make decision on the model redesign.

B. Validation process

Figure 1 shows verification and validation processes in discrete event modeling and simulation study with experimental frame. In DEVS modeling process, experimental frame is used to make decisions on the model. Though it is main function is to transfer outcome measures to variables, it is also used in evaluating how well model objectives are achieved. In the case study, it is tried to show using experimental frame for verification and validation process. Figure 1 helps to map V&V onto whole modeling process rather giving an idea that V&V process is a singular phase or step in modeling. V&V is a continuous process confluent with modeling and simulation [13]. As shown in the Figure 1, modeling activity starts with problem statement and statement of objectives. After development of the conceptual model, conceptual model validation is conducted to determine detail level of proposed model. Validation of the conceptual model means detail of proposed model is sufficient and performed assumptions are accurate.

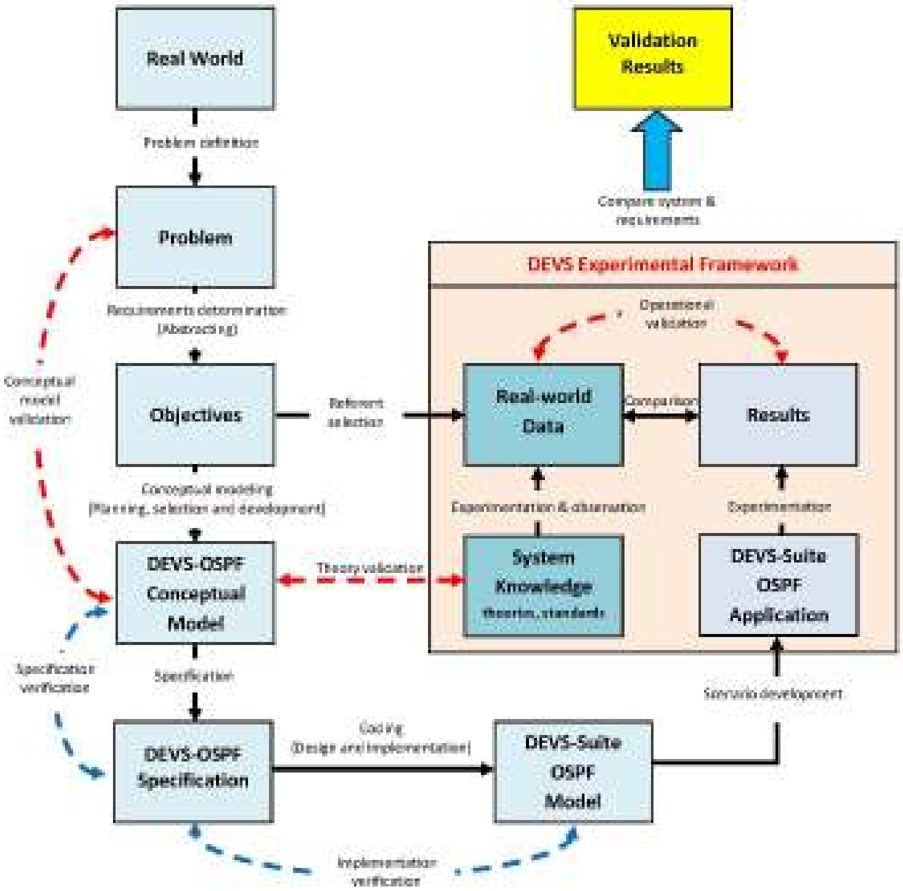

Real World
Validation Results
Problem definition
Compare system & requirements
Problem
DEVS Experimental Framework
Requirements determination (Abstracting)
Operational validation
Conceptual model validation
Objectives
Relevant selection
Real-world Data
Comparison
Results
Conceptual modeling (Planning, selection and development)
Experimentation & observation
Experimentation
DEVS-OSPF Conceptual Model
Theory validation
System Knowledge
theories, standards
DEVS-Suite OSPF Application
Specification verification
Specification
Scenario development
Coding (Design and implementation)
DEVS-OSPF Specification
DEVS-Suite OSPF Model
Implementation verification

Fig. 1. Verification and validation in DEVS modeling process and Experimental Frame

Theory validation relates to the technical details of developed model and how it relates to the relevant formalisms, approaches, standards, knowledgeable expertise and underlying theories, for example, for OSPF protocol model, its resemblance to RFC 2328 standard [14]. Validation of theory requires asking questions what theory is used in the models? how the theory is used in the models ? and how the theories are combined in the models ? [15].

Operational validation refers to determining model outputs is sufficiently correct data obtained from the real world. Operational validation activity includes validation of required model representations and their associated knowledge bases and validation of simulation algorithms, formalisms and models and their associated data [16]. Operational validations are directly related to model credibility [11].

Besides above kinds of validation, two aspects of verification can be applied: specification and implementation verification. Specification verification ensures to truly specify the model behavior defined in conceptual model, while implementation verification deals with all specifications are coded in the model run on DEVS-Suite as built.

C. Validation techniques

Many verification and validation techniques are available for modeling and simulation study [13] [25]. In this section, various V&V techniques are tried to summarize and an approach proposed by Forrester and Senge [12] is given in particular.

Analytical method validation is a process of performing several tests designed to verify that an analytical test system is suitable for its intended purpose and is capable of providing useful and valid analytical data. A validation study involves testing multiple attributes of a method to determine that it can provide useful and valid data when used routinely.

III. Devs Modeling And Simulation Framework

Discrete Event Systems Specification (DEVS) [4] is well suited for formally describing concurrent processing and the event-driven nature of arbitrary configuration of nodes and links forming network systems. DEVS is a mathematical modeling formalism for describing (discrete and continuous) dynamical systems and supports hierarchical modular model construction, distributed execution, and therefore characterizing complex, large-scale systems. Its generic system-theoretic concepts and mathematical formulation provide a basis for describing component models which have modular structure and behavior specifications. This framework lends itself to object-based abstraction, encapsulation, modularity and hierarchy concepts and implementation. Its simulation protocol enforces causality, concurrency, and timing among DEVS atomic and coupled models.

DEVS-Suite is an open source, discrete event, general-purpose simulation environment [1]. It is a new generation extended from the DEVSJAVA simulator and DEVS Tracking Environment. The main modules of the DEVS-Suite are DEVSJAVA [3], DEVS tracking Environment [20], and timeview [2]. DEVS-Suite can simulate models specified using the DEVS formalism [4]. The architecture of the DEVS-Suite simulator environment is Model Facade View Control (MFVC) [20] by which simulation data can be displayed with its animation and viewing of time trajectories generated by the parallel DEVS abstract simulator. Soft synchronization among timeviews and animation is supported based on the simulator's logical (or real-time) execution speed [21].

IV. Devs-Suite Validation Framework: Ospf Case

Validation is the most important phase in developing a model. Validation tests are needed in a modeling study since a model cannot be accepted unless it passes them [25], [10]. Validation schemes are generally framework based methodical procedures as well as dynamic processes which have to be applied by a modeler during model development. There are many model validation schemes summarized in [25] and [11]. From these schemes, a widely approved validation scheme is presented in[12] and followed in this study. In this scheme, confidence in the model increases as model passes more tests. This validation scheme is adopted and applied in this study due for it is appropriateness for dynamic models such as distributed systems and networks. Applied validation scheme is mainly divided into four phases. These phases are (1) specifying model objectives, (2) validating model structure, (3) validating model behavior and (4) policy implications. In following sections, these steps are followed to validate the developed model and summarized due for place limitations.

A. Model objectives

It is important to set objectives correctly when starting to model a system. In the previous sections, we gave some background about motivation and problem entities. Once again, it is good to list main objectives to model such a system. We aimed to develop a network simulator having;

- system theoretic design with modular and hierarchy,
- easy to deploy,
- good visualization,
- performance and scalable,
- high highly tractable,
- advanced testing framework,
- parallel and distributed capability.

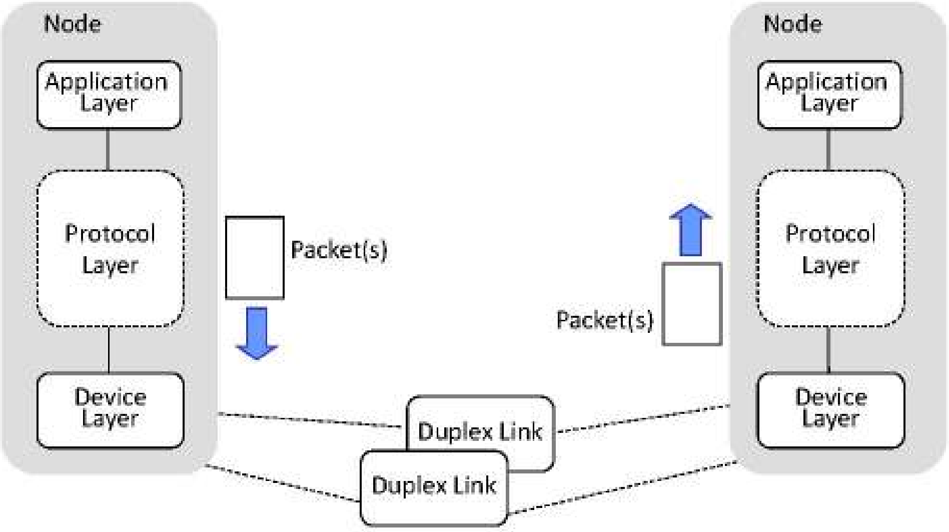


Fig. 2. Network layers structure of the OSPF model

Except for last one, model has all properties. Parallel and distributed capability of the model is still ongoing research and will be done by DEVS/HLA framework [26].

B. Model structure validity

Some empirical and theoretical structure tests are experimented with the model to show its structural confirmation. Experiments may be separated into two categories: network structure confirmation done by theoretical tests and OSPF protocol confirmation done by empirical tests.

Theoretical structure tests of the OSPF model are done with comparison model structure with generalized knowledge about the system. DEVS network model is developed according to network OSI standard with several abstractions. Since protocol implementation and education is in focus, first abstraction is to flatten seven segment OSI layers to three layers (see Figure 2).

Empirical structure tests of the OSPF model includes comparison of the OSPF model structure with information obtained directly from the real system. To do this, we develop network models up to ten thousands nodes and measure efficiency of the networks. Efficiency is estimated as performed network tasks such as message delivery, scheduled events and routing databases' correctness. In Figure 3, efficiency trajectory of the developed OSPF model is shown. For small scale models, efficiency is ideal (i.e. simulator is running with highly correctness), however in large models, simulator is deviating from reality (for example, 99.5% for 10.000 nodes). According to the tests and evaluations above, developed model is structural valid.

C. Model behavior validity

Besides structural validity tests, certain tests are needed to measure how accurately the model can reproduce the major behavior patterns shown by the real system. To do this, we select widely accepted network simulator 2 (ns2) models instead of real system. In order to validate behavior of the DEVS-Suite OSPF model, first we considered a small scale topology as shown in Figure 1. In this work, the ns-2 network simulator is chosen for comparison to show the behavior validity of developed OSPF model in DEVS-Suite. Furthermore, to show the key structural differences between modeling DEVS-Suite and ns-2, we use the same model configurations. Finally, comparison with analogous ns-2 test traces is included.

1) Test Simulation Environment and Configurations: When appropriate, we execute the DEVS-Suite OSPF model tests under the conditions as close as possible to the ns-2 tests, but some small differences exist because of different internal parameter settings and/or different level of modeling detail. Simulation experiments were performed both in ns-2 version 2.32 and DEVS-Suite version 2.0. The Table I below shows some differences between ns-2 and DEVS-Suite implementations.

2) Simulation Results and Evaluation: Together structural tests, these validation tests give us at least some assurance that the behavior of the DEVS OSPF model is reasonable. DEVS-Suite more suitable for system level simulation as opposed to ns-2 being suitable for detailed network protocol designs (algorithms).

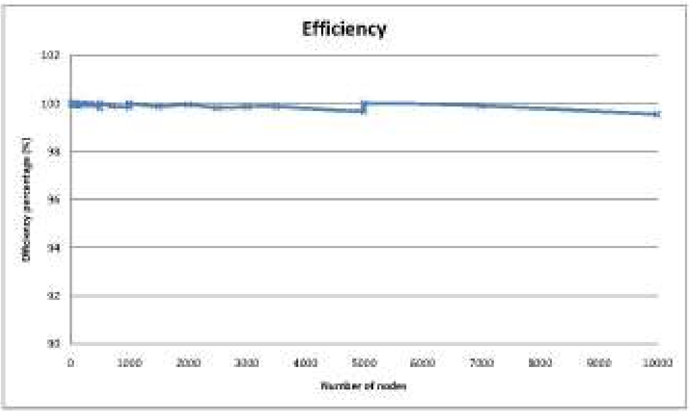


Fig. 3. Efficiency tests of the OSPF model

According to the configuration parameters listed in Table I, simulation experiments with both ns-2 and DEVS-Suite simulators are performed for ten seconds. The throughput results as a function of time are shown in Figure 4.

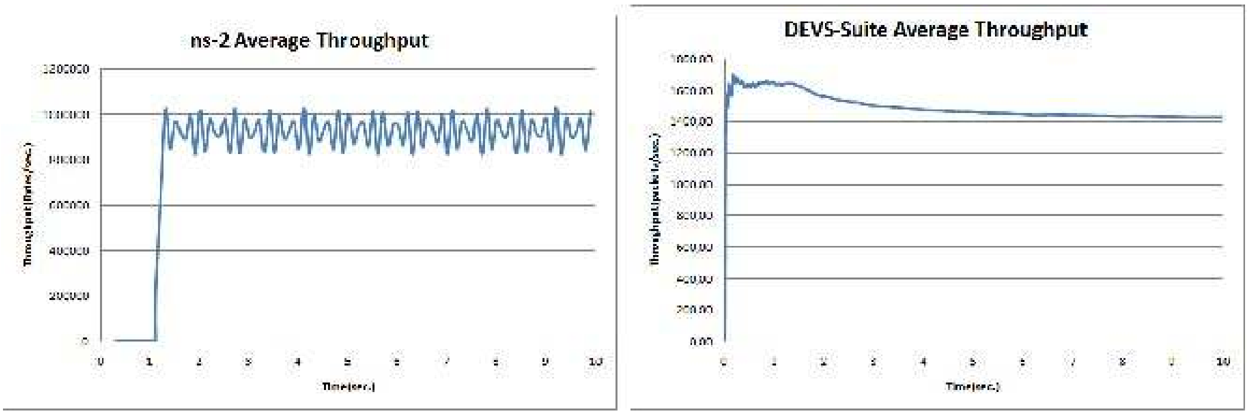
As depicted in Figure 4 (a) and (b), after a stabilization phase time (2 seconds), throughput curves converge to nearly the same average values, 822.4 KB/sec. for ns-2 and 1489 packets × 552 bytes = 821.9 KB/sec. for DEVS-Suite. We also observed the routing tables for the four router nodes to be consistent – the creation of tables was validated step by step.

V. Comparison/Discussion

In the following sections, we identify some of the reasons behind the differences between ns-2 and DEVS-Suite simulators with respect to modeling and education aspects.

	DEVS-Suite	**ns-2**
Topology	4 routers, 4 bi-directional links	4 routers, 4 duplex links
Protocol	OSPF	Link State(LS)
Processing speed	1 msec./event	N/A
Event frequency	1000 events/sec.	28388 events/sec.
Packet sizes	552 bytes	552 bytes
Link bandwidth	2 Mbps	2 Mbps
Link delay	1 msec.	1 msec.
Traffic type	Uniformly random	FTP over TCP
Queue Type	FIFO-Priority	DropTail
Queue Limit	200 KB	20 Packets
Simulation time	10 sec.	10 sec.

Table I. Simulation Model Parameters of Ns-2 and DEVS-Suite



(a) (b)

Fig. 4. Simulated performance measurements of ns-2 and DEVS-Suite for throughput

VI. Conclusion And Future Work

This paper presented a new simulation environment for education and research of computer networks. The DEVS-Suite simulator for modeling OSPF overcomes the limitations of ns-2 concerning the visualization and execution performance. In contrast to ns-2, DEVS-Suite$_{\text{WebStart}}$ enables users to automatically download its latest version and network models and execute them as efficiently given their computer hardware and software resources. Our experiences show that students benefit from the capabilities of DEVS-Suite simulator. The simulator’s visualization and automation provided for tracking of components’ behavior as time trajectories and tabular data help teaching and learning of networking theory with reduced burden on being an expert software developer. Visualization of the protocol logic in the form of creation and alteration of routing tables and databases facilitate active learning. In addition, students can gain the knowledge of modeling and simulation technique for performance evaluation of networking systems. Future work includes developing other network protocol models and evaluating them through classrooms. Additionally, the DEVS-Suite can be extended with data analysis features. Another desirable addition is to use variable structure DEVS since in some cases it can better represent the dynamic nature of the distributed computer networks. Another attractive capability is to visually develop models and automatically generate source code using CoSMoS (Component-based System Modeler and Simulator) [23].

Acknowledgment

This work has been funded by the Sakarya University Scientific Research Projects Agency under contract 2007-05-02-001. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Sakarya University.

References

[1] S. Kim, H. Sarjoughian, and V. Elamvazhuthi, “DEVS-Suite: A simulator supporting visual experimentation design and behavior monitoring,” in *Proceedings of the Spring Simulation Conference*, San Diego, CA, March 2009, pp. 29–36.

[2] “DEVS-Suite,” http://sourceforge.net/projects/devs-suitesim/, 2010.

[3] ACIMS, “DEVSJAVA modeling and simulation tool,” http://www.acims.arizona.edu/SOFTWARE, 2010.

[4] B. P. Zeigler, H. Praehofer, and T. G. Kim, *Theory of Modeling and Simulation*. New York: Academic Press, 2000.

[5] X. Hu, B. P. Zeigler, and S. Mittal, “Variable structure in DEVS component-based modeling and simulation,” *SIMULATION: Transactions of The Society for Modeling and Simulation International*, vol. 81, no. 2, pp. 91–102, 2005.

[6] H. Sarjoughian, “DEVS-Suite WebStart,” http://acims1.eas.asu.edu/WebStarts/, 2010.

[7] “Java Web Start Technology,” 2010.

[8] A. Zengin, H. S. Sarjoughian, and H. Ekiz, “Honeybee inspired discrete event network modeling,” in *16th European Simulation Symposium*, Budapest, Hungary, 2004, pp. 176–182.

[9] H. Sarjoughian, Y. Chen, and K. Burger, “A component-based visual simulator for mips32 processors,” in *Frontiers in Education*, Saratoga Spring, New York, October 2008.

[10] O. Balci and R. G. Sargent, “Validation of multivariate response models using hotelling’s two-sample t2 test,” *Simulation*, vol. 39, no. 6, pp. 185–192, 1982.

[11] M. S. Martis, “Validation of simulation based models: A theoretical outlook,” *The Electronic Journal of Business Research Methods*, vol. 4, no. 1, pp. 39–46, 2006.

[12] J. W. Forrester and P. M. Senge, “Test for building confidence in system dynamics models,” *TIMS studies in the management sciences*, vol. 14, pp. 209–228, 1980.

[13] O. Balci and R. G. Sargent, “Validation of simulation models via simultaneous confidence intervals,” *American Journal of Mathematical and Management Sciences*, vol. 4, no. 3 & 4, pp. 375–406, 1984.

[14] I. E. T. Force, “Ospf version 2 (rfc 2328) internet standards track protocol,” http://www.ietf.org/rfc/rfc2328.txt, 2009.

[15] C. M. Macal, “Model verification and validation,” in *Workshop on ”Threat Anticipation: Social Science Methods and Models”*, The University of Chicago and Argonne National Laboratory, Chicago, IL, April 7-9 2005.

[16] Validation, “Us department of defence rpg special topic,” 2004.

[17] A. Chow, “Parallel DEVS: A parallel, hierarchical, modular modeling formalism and its distributed simulator,” *Transactions of the Society for Computer Simulation International*, vol. 13, no. 2, pp. 55–67, 1996.

[18] Y. J. Kim, J. H. Kim, and T. G. Kim, “Heterogeneous simulation framework using DEVS BUS,” *Simulation*, vol. 79, pp. 3–18, 2003.

[19] H. Sarjoughian and F. Cellier, *Discrete Event Modeling & Simulation Technologies: A Tapestry of Systems and AI-based Theories and Methodologies for Modeling and Simulation*. Springer Verlag, 2001.

[20] H. S. Sarjoughian and R. Singh, “Building simulation modeling environments using systems theory and software architecture principles,” in *Proceedings of the Advanced Simulation Technology Conference*, Washington DC, April 2004, pp. 99–104.

[21] E. Helser, “Design and analysis of view syncrhonization in DEVS-Suite,” Master’s thesis, Computer Science and Engineering Department, Arizona State University, 2009.

[22] Ptolemy, “Ptolemy project,” http://ptolemy.eecs.berkeley.edu/ptolemyII/, 2010.

[23] H. S. Sarjoughian and V. Elamvazhuthi, “Cosmos: a visual environment for component-based modeling, experimental design, and simulation,” in *Simutools ’09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009, pp. 1–9.

[24] Mathworks, “SimEvents,” http://www.mathworks.com/products/simevents/, 2010.

[25] Y. Barlas, “Formal aspects of model validity and validation in system dynamics,” *System Dynamics Review*, vol. 12, no. 3, pp. 183–210, 1996.

[26] H. S. Sarjoughian and B. P. Zeigler, “DEVS and HLA: Complementary paradigms for modeling and simulation?” *Transactions of the Society for Modeling and Simulation International*, vol. 17, no. 2, pp. 187–197, 2000.

[27] D. M. Nicol, “Scalability of network simulators revisited,” in *Proceedings of the Communications Networking and Distributed Systems Modeling and Simulation Conference*, 2003.

[28] G. F. Riley, M. H. Ammar, and R. Fujimoto, “Stateless routing in network simulations,” in *in Proceedings of the Eighth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 2000, pp. 524–531.

[29] ns 2, “ns-2 network simulator,” http://www.isis.edu/nsnam/ns/, 2010.

[30] G. V. Wilson, “The history of the development of parallel computing,” http://ei.cs.vt.edu/ history/, Virginia Tech/Norfolk State University, 1994, retrieved on 2008.

[31] Wikipedia, “Round-robin scheduling,” http://en.wikipedia.org/wiki/Round-robinscheduling, 2008.

[32] H. Sarjoughian and K. Shaukat, “A comparative study of DEVS and ns-2 modeling approaches,” *Transactions of the Society for Modeling and Simulation International*, 2009.

[33] H. S. Sarjoughian, S. Kim, M. Ramaswamy, and S. Yau, “An soa-devs modeling framework for service-oriented software system simulation,” in *Winter Simulation Conference*, Miami, FL, 2008, pp. 845–853.

[34] G. R. Mayer and H. S. Sarjoughian, “Composable cellular automata,” 2009, simulation: Transactions of The Society for Modeling and Simulation International.

[35] C. M. Overstreet and A. Martens, “Introduction to special issue: Modeling and simulation in teaching and training,” *SIMULATION*, vol. 82, pp. 681–683, 2006.