Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

APPLICATION OF DEVS FRAMEWORK IN CONSTRUCTION SIMULATION

Sivakumar Palaniappan
Anil Sawhney

Del E. Webb School of Construction
Ira A. Fulton School of Engineering
Arizona State University
Tempe, AZ 85287-0204, U.S.A.

Hessam S. Sarjoughian

Arizona Center for Integrative Modeling & Simulation
Dept. of Computer Science and Engineering
Ira A. Fulton School of Engineering
Arizona State University
Tempe, AZ 85281-8809, U.S.A.

ABSTRACT

The Discrete Event Systems Specification (DEVS) framework is based on systems engineering principles and is used for modeling and simulation in many application domains. This framework supports a number of important features such as component based hierarchical simulation model development, scalability, reusability and distributed simulation. This paper demonstrates the application of the DEVS framework for construction simulation through a simple, but representative real world application. The application focuses on analyzing the work flow between various trade contractors in production home building. The software tool 'DEVSJAVA' is used for modeling and simulation of the construction application. DEVSJAVA is object oriented; it implements the DEVS framework using Java programming language. This paper consists of three main components: (i) current overview of the state-of-the-art in construction simulation methods and tools; (ii) introduction to the DEVS framework; and (iii) detailed construction example to highlight the application of DEVS framework.

1 CONSTRUCTION SIMULATION

Construction operations involve a number of repetitive tasks similar to manufacturing operations, for example, earth hauling, tunneling, road construction, and glass installation on tall buildings [Abourizk et al 1992]. This demonstrates the potential to apply discrete event simulation techniques (especially process and resource based simulation) for modeling and analysis of construction operations. The history of construction simulation dates back to 1960's with the development of simple networking concepts [Abourizk et al 1992]. These networking concepts were used as a modeling framework to study construction operations. As a first attempt, "link-node" model was developed and used to assist selection of construction equipment [Teicholz 1963]. After this, the CYCLONE (Cyclic Operations Network) modeling framework was developed by Halpin [1977]. CYCLONE consists of five basic modeling elements such as 'normal', 'combi', 'queue', 'function' and 'counter'. The functionality of each modeling element is documented in the literature [Halpin 1977, Abourizk et al 1992]. This was followed by the development of a software tool using micro computer called MicroCYCLONE [Lluch and Halpin 1982].

The application of CYCLONE to model and analyze a number of construction scenarios is described in detail by Halpin and Riggs (1992). CYCLONE modeling framework provided the foundation for many construction researchers to develop a number of construction simulation tools in the past 20 years. This includes the development of (i) INSIGHT system based on CYCLONE with interactive user interface [Paulson 1987]; (ii) CYCLONE with advanced resource handling capabilities called RESQUE [Chang and Carr 1986]; and (iii) UM-CYCLONE with menu driven user interface [Ioannou 1990].

A discrete event simulation system with object oriented design concepts called 'COOPS' (Construction Object-oriented Process Simulation System) was developed [Liu and Ioannou 1992]. Further, CIPROS, an object-oriented system for simulating construction plans was developed [Odeh et al 1992]. A simulation programming language, called STROBOSCOPE, (State and Resource based Simulation of Construction Operations), was designed and developed to specifically model and simulate construction operations [Martinez and Ioannou 1994]. STROBOSCOPE is based on three phase activity scanning approach. A detailed discussion of the three simulation strategies namely process interaction, activity scanning and event scheduling is provided by Martinez and Ioannou [1999].

The application of hierarchical simulation modeling (HSM) method in construction simulation was demonstrated through a bridge project [Sawhney and Abourizk 1995]. Though a number of construction simulation tools were developed following the CYCLONE methodology, their use was mostly limited to academic and research community [Abourizk and Hajjar 1998]. This is primarily

attributed to the significant differences between the simulation model representation and the real world construction system. This made the process of developing and understanding the simulation model more tedious for construction experts who have limited amount of time to apply simulation in construction. To overcome this difficulty, the concept of special purpose simulation (SPS) was introduced as an application framework for developing construction simulation tools [Hajjar and Abourizk 2000]. SPS is based on object-oriented application framework and the constructs of SPS tools are similar to the objects of real world system. Three independent customized simulation tools were developed based on the SPS concept [Hajjar and Abourizk 2002]. They are (i) 'AP2-Earth' for the analysis of large earth moving projects (ii) 'CRUISER' for modeling aggregate production plants and (iii) 'CSD' for construction site dewatering. The use of the above tools by large construction companies demonstrated the success of SPS based simulation tools.

Based on the experience gained through the development of SPS based simulation tools, a unified modeling methodology for construction simulation was proposed [Hajjar and Abourizk 2002]. This approach resulted in the development of a simulation tool development and utilization environment called 'Simphony'. Simphony is an integrated environment for construction simulation. The development time for new SPS tools was reduced significantly due to the construction simulation object library provided within the Simphony framework. To understand the important features of Simphony, the reader can refer to Abourizk and Mohamed (2000).

In an effort to extend the field of construction simulation, the authors undertook this experimentation with DEVSJAVA. The objective of this work is to demonstrate the application of DEVSJAVA in construction simulation through an example problem in production home building. The remaining content in the paper is organized as follows: The next section presents an introduction to the DEVS framework. The third section presents background information about production home building and the details of the example problem. The simulation model development using DEVSJAVA is described in the fourth section. A discussion on the simulation results is presented in the fifth section. Finally the sixth section presents conclusions based on the simulation results and the contribution of this work.

2 DEVS FRAMEWORK

The Discrete Event Systems Specification (DEVS) framework was introduced in 1972 by Zeigler [Zeigler et al 2000]. This framework was later extended to support parallel model specification and execution [Chow 1996], hierarchical model development and modularity, and object orientation [Zeigler and Sarjoughian 2003]. DEVS framework was implemented as a simulation environment called 'DEVSJAVA' using the Java programming language [ACIMS 2005].

The DEVS framework and DEVSJAVA simulation environment have a number of important features for modeling and simulation: They are summarized as follows:

- DEVS framework is founded on system-theoretic principles including component based hierarchical modeling using input/output ports and couplings. The framework defines two types of models – atomic model and coupled model; atomic models are the basic modeling constructs whereas coupled model represents a group of atomic and/or coupled models. It supports feedback loops; the process logic of a component can be customized as function of the property of an incoming entity. This helps to develop simple and effective simulation model without any repetitions as in case of 'feed forward' model.
- The framework supports scalability and reusability through the use of one coupled model as a basic component in another coupled model. It supports hierarchical simulation model development. This enables to build and test large complex simulation models in an incremental fashion.
- DEVS framework supports distributed simulation of models thereby allowing development and deployment of very large-scale complex models.
- Since the DEVS framework is generic, its application is not limited to process and resource based simulation.
- DEVSJAVA offers comprehensive flexibility to customize the simulation model development based on the specific requirements of each problem.
- The modeler can visually monitor the simulation of entities across various processes through the 'STEP' option. This helps to iterate through the simulation event by event and to check the correctness of the model through traces.

The above features of DEVS framework can be effectively applied to construction simulation.

2.1 Modeling and Simulation Concepts and Artifacts

A general framework of modeling and simulation is defined to consist of three basic elements: real system, model and simulator [Zeigler et al 2000]. The real system is considered as the fundamental source of data or input/output pairs. It can be either in existence or proposed. The model is a set of instructions which define how inputs and/or states may be processed and outputs generated. Every model has a corresponding simulator which is responsible for executing the model instructions according to a well-defined abstract protocol. Figure 1 shows these elements and their key relationships.

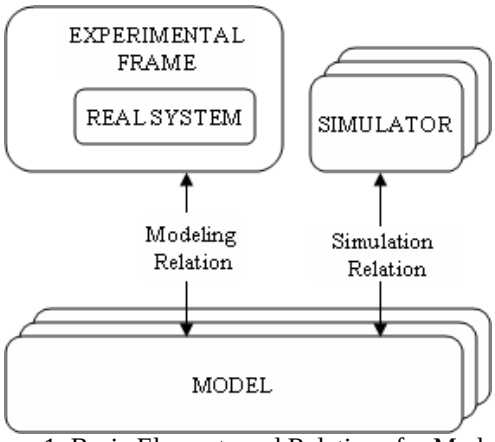

Figure 1: Basic Elements and Relations for Modeling and Simulation

The modeling relation links the real system and the model. It defines how well the model represents the real system that is being modeled (scope of the model). The simulation relation links the model and the simulator. It represents how faithfully the simulator can execute the instructions of the model (validity or correctness of the model). Experimental frame specifies the conditions under which the system is experimented. It systematically handles the existence of many models for a given system and thus formalizes the relationship between the system and its models. It supports specifying the scope of the model by defining the objectives of a simulation project in terms of observable inputs and outputs. Experimental frame consists of the generator and the transducer. The function of a generator is to create entities at specific time intervals and release them into the system. The transducer collects simulation output data specified by the modeler.

2.2 Types of Models in DEVS

There are two types of modeling elements in DEVS framework namely atomic models and coupled models [Zeigler et al 2000]. Atomic models are the basic building blocks of the DEVS framework from which larger models are built. For example, each of generator and transducer can be an atomic model.

An atomic model contains the following information [Zeigler and Sarjoughian 2003]: (i) set of input ports to receive external inputs (ii) set of output ports to send output (iii) set of states and parameters; examples of state include 'active' or 'passive'; parameter refers to 'sigma'; values for 'sigma' are positive values including zero and infinity. Sigma specifies the time that a model will stay in a given state. Sigma can also be stated as the time to next state transition which is determined by the time advance function (iv) internal transition function that specifies the state to which the system will change after the time specified by the time advance function has elapsed (v) external transition function that specifies the system state changes when an external input is received (vi) confluent transition function that is applied when the arrival of an external input and an internal transition happens at the same time and (vii) output function that generates external output.

A coupled model is a model that consists of several atomic and/or coupled models connected together internally through coupling. Experimental frame is one example for coupled model since it consists of two components namely generator and transducer. Figure 2 shows an example of a coupled model.

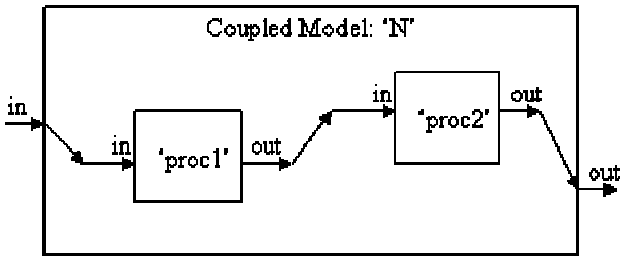

Figure 2: Example for Coupled Model

A coupled model consists of all information that an atomic model consists (closure under coupling). In addition, they contain the coupling relationship namely external input coupling, internal coupling and external output coupling. External input coupling connects the input ports of the coupled model to one or more input ports of the coupled model's components. Internal coupling connects the output port of one component to the input port of another component within the coupled model. External output coupling connects output ports of one or more components to output port of the coupled model.

The coupling details for the example shown in Figure 2 are given below:

External Input Coupling (EIC):
EIC = { (N, "in"), (proc1, "in") }
Internal Coupling (IC):
IC = { (proc1, "out"), (proc2, "in") }
External Output Coupling (EOC):
EOC = { (proc2, "out"), (N, "out") }

Further details about atomic and coupled models, their formal specification including simulation protocols and several examples can be found in [Zeigler et al 2000] and [Zeigler and Sarjoughian 2003].

3 CONSTRUCTION EXAMPLE

This section presents the details of a production home building example problem that is used to demonstrate the features of DEVSJAVA.

3.1 Problem Background

Production home building consists of mass construction of 100 or so similar homes on a tract called subdivision. These homes (sometimes called lots) are similar in nature (in terms of their basic configuration) but have minor variations (based on home buyer option selections) to satisfy individual customer preferences [Bashford et al 2003]. The construction of each lot consists of 10 to 12 major phases and a total of 90 to 100 activities or work packages. These activities are completed by the coordination of 25 to 35 specialized trade contractors.

Successful transfer of a lot from one trade contractor to another trade contractor is critical for the timely completion of a residential construction project. Many factors cause delay during this transfer and they include: (i) preceding trade does not complete the work on time (ii) preceding trade completed the work but defects in construction are found by a building inspector or a quality control personnel (iii) the work completed by preceding trade was damaged and (iv) communication failure occurred between the two trade contractors. Among these, failure in code compliance inspections or quality control inspections are the prime factors that causes delay during a control transfer most of the times. This was obvious from the data collected from City of Peoria for the period October 2003 to September 2004. The inspection pass rate of critical inspections was computed and it was found that the first time inspection pass rate of critical inspections – UG/Soils, pre slab, framing, drywall and final were 59%, 72%, 15%, 75%, and 33% respectively [City of Peoria 2004]. From this, it can be noted that the first time inspection pass rate for critical inspections varies from 15% to 75%. The poor inspection pass rate of one trade can significantly influence the work flow of a succeeding trade, which is true in case of framing.

The negative influence of workflow variability on the succeeding trade is demonstrated through a parade game in construction [Tommelein et al 1999]. This game consists of a sequence of five commercial construction trades. The variability in the number of jobs released from one trade to another trade was controlled by number written on the rolling dice. The impact of workflow variability and methods to improve the workflow is presented in construction literature [Ballard 1999; Ballard et al 1998].

Few studies have been reported on the application of discrete event simulation to analyze production home building scenarios [Sawhney et al 2001; Bashford et al 2003; Sawhney et al 2005]. However, the scope of these studies does not include modeling the workflow in production home building. The current study focuses on the following: (i) analyze the work flow variability in residential construction and (ii) demonstrate the application of DEVSJAVA for construction simulation.

3.2 Problem Details

The objective of the test problem is to model the workflow in production home building. The preceding trade is considered as rough framing trade and the succeeding trade is drywall trade. The inspection pass rate of the framing phase is varied from 15% to 100% and its impact on the work flow to the drywall trade is measured in terms of number of jobs released per week from the framing trade.

The process details of rough framing trade is described through the following steps: (1) home builder notifies the framing trade (2) check for framing crew availability, start the construction work if crew available else wait for crew (3) trade notifies the home builder after the completion of work and then builder notifies city inspector (4) city inspector completes the inspection (5) check inspection result, if pass then the lot is released to the succeeding trade, if fail then the following group of steps happen till the lot passes the inspection (6) home builder notifies framing trade about fail in inspection (7) check for framing crew, if crew available, crew starts working on rework else wait for crew (8) framing trade informs builder about completion of rework (9) builder notifies city inspector for inspection, the process flow repeats from step-5 again. The above process flow details for framing trade are shown in Figure 3.

The following input data regarding the framing trade is needed to conduct the simulation: (1) start pattern (2) inspection pass rate (3) crew size (4) duration for construction and rework. Start pattern refers to the number of jobs per week that arrives to the framing trade from his predecessor. Figure 4 shows the start pattern of the framing trade that was used in this study. This start pattern was obtained based on a complete lot simulation.

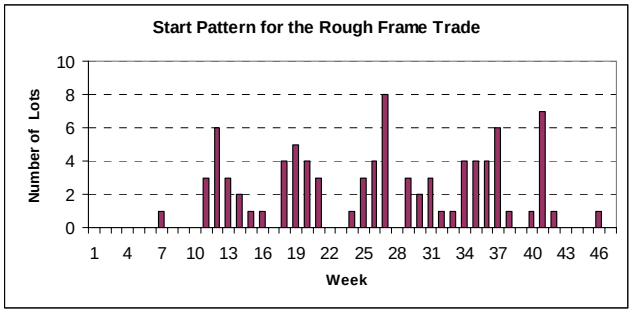

Figure 4: Start Pattern for the Rough Framing Trade

The framing inspection pass rate was computed using the inspection data collected for one complete year - October 2003 to September 2004 [City of Peoria 2004]. From a total of 35631 framing inspections that belong to 932 lots, the framing inspection pass rate was computed as 15%, 43%, 55%, 62%, 71%, and 100% for the first, second, third, fourth, fifth, and sixth inspections respectively.

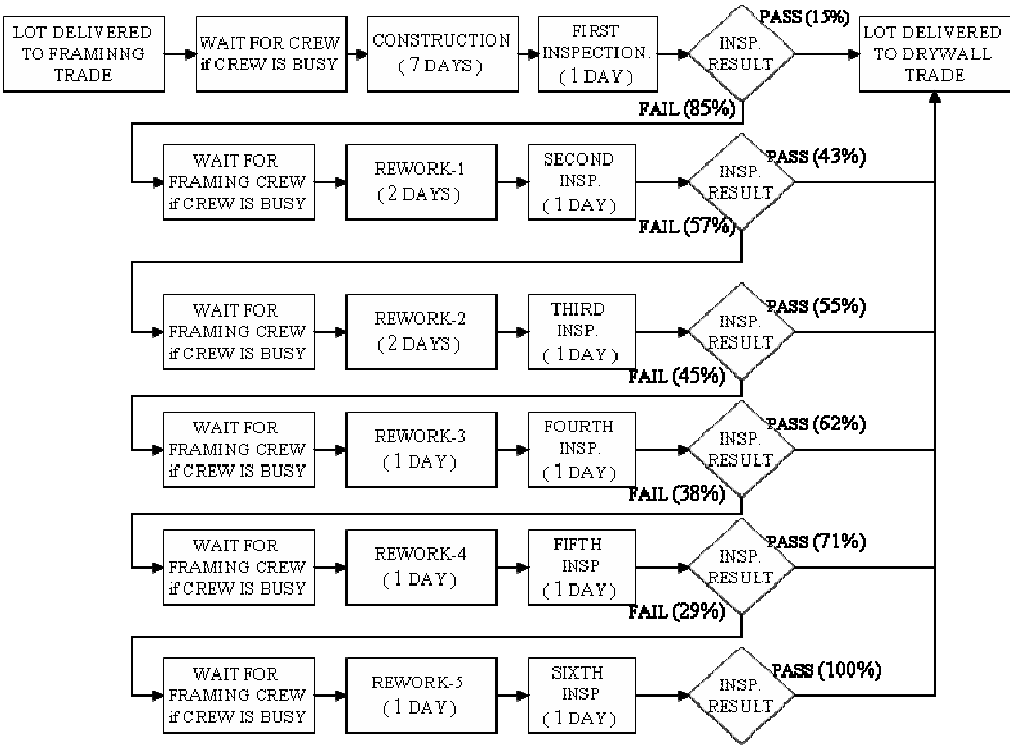

Figure 3: Process Map for the Rough Frame Trade

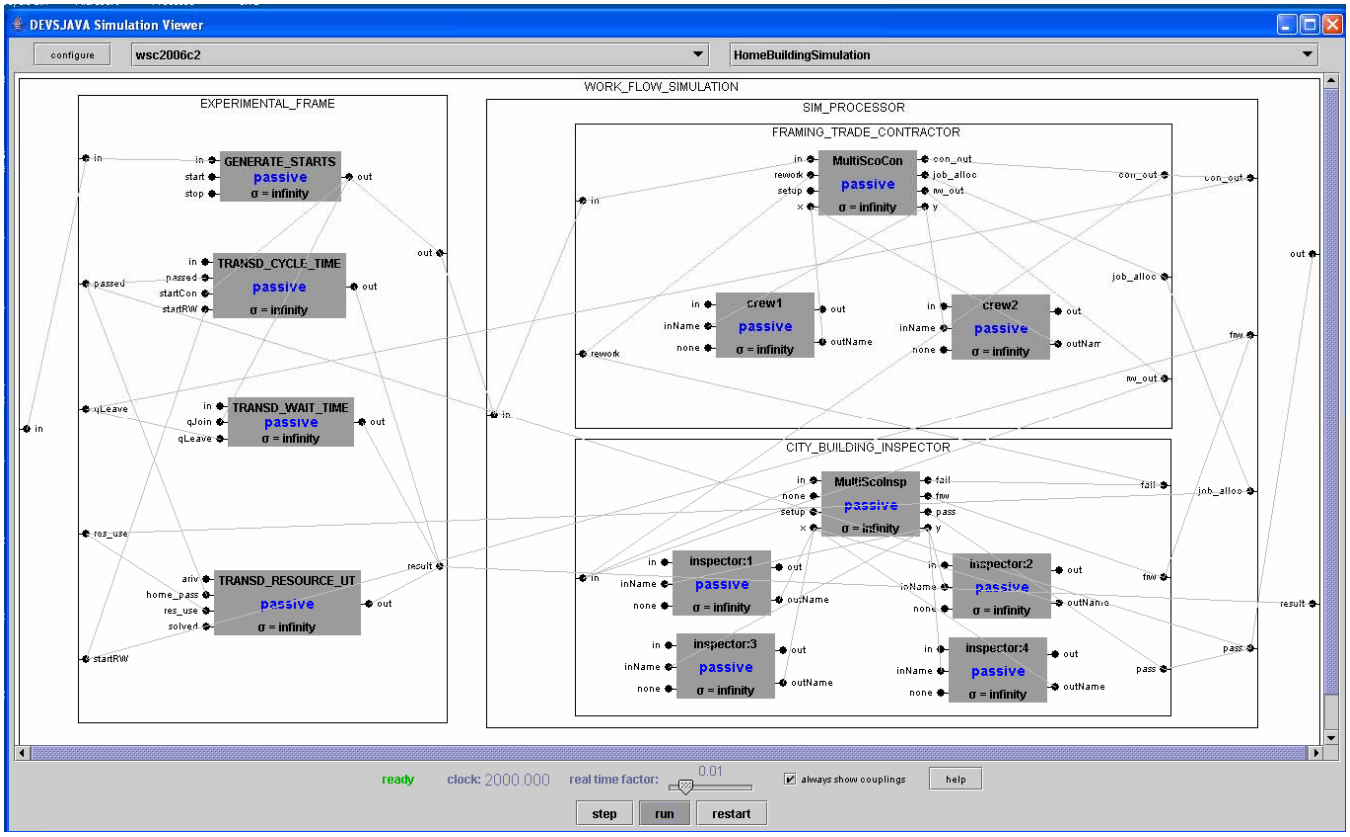

Figure 5: Snapshot of the Complete Simulation Model

Two scenarios were analyzed based on inspection pass rate. In scenario-1, the pass rate is based on the actual inspection data analysis which is 15%, 43%, 55%, 62%, 71%, and 100%. In scenario-2, the pass rate is 100% in the first inspection.

The number of framing crews in a subdivision normally varies from two to four. In this study, it is assumed that the framing crew size is two. The duration for construction was computed using the schedule data obtained from one of the leading home builders; it was found to be 7 days. Since the current study assumes that all lots passes in the 6th inspection, there can be a maximum of 5 reworks. The duration of rework-1, rework-2, rework-3, rework-4 and rework-5 were assumed as 2 days, 2 days, 1 day, 1 day and 1 day respectively. Actually, it was found that the reworks follow exponential distribution based on inspection data analysis. Constant durations for rework are used in this study; the objective behind this is to minimize the variability due to statistical sampling and to observe the effect on the system performance due to the variation in the input data to the simulation model.

4 SIMULATION MODEL DEVELOPMENT

The simulation model was implemented using DEVSJAVA. This model consists of 5 coupled models and they are: (i) 'WORK_FLOW_SIMULATION' representing the complete simulation model; it consists of the remaining four coupled models as basic components (ii) 'EXPERIMENTAL_FRAME' (iii) 'SIM_PROCESSOR' representing the construction, inspection and rework (iv) 'FRAMING_TRADE_CONTRACTOR', a coupled model representing the framing construction and reworks process and (v) 'CITY_BUILDING_INSPECTOR', a coupled model representing the inspection process by city building inspector. The last two coupled models are two basic components within the coupled model 'SIM_PROCESSOR'. The complete simulation model is shown in Figure 5.

4.1 Coupled Model 'EXPERIMENTAL_FRAME'

Experimental frame is a coupled model that specifies the conditions under which the simulation model is experimented or simulated. It consists of two types of atomic models namely 'generator' and 'transducer'. The component 'generator' is represented by the atomic model 'GENERATE_STARTS'. It generates a specific number of lots each week and releases them to the framing trade. The 'out' port of the generator is connected to the 'out' port of the 'EXPERIMENTAL_FRAME' which in turn connects to 'in' port of the 'SIM_PROCESSOR'.

The transducer collects data on simulation output variables. The experimental frame consists of three transducers. They are 'TRANSD_CYCLE_TIME', 'TRANSD_WAIT_TIME' and 'TRANSD_RESOURCE_UT'. These three transducers compute the cycle time, average waiting time of lots for resources and resource utilization. The output ports of 'SIM_PROCESSOR' are connected to the input ports of these transducers through the input ports of experimental frame. The transducer 'TRANSD_CYCLE_TIME' computes the cycle time of each lot. It uses the port 'startCon' to note the time at which each entity is created; further it uses the port 'passed' to note the time at which each entity passes the inspection; cycle time is computed as the difference between these two times. The finish time (inspection pass time) of all entities is noted and using this, the number of lots per week released by the framing trade is computed. The number of lots per week received by drywall trade is same as the number of lots per week released by the framing trade. The transducer 'TRANSD_WAIT_TIME' computes the average waiting time of lots for framing crew. It marks the time at which entities join/leave the queue by using the ports 'qJoin' and 'qLeave' respectively. The waiting time is computed as the difference between the queue leaving time and the queue joining time. The transducer 'TRANSD_RESOURCE_UT' computes the resource utilization; it uses the port 'res_use' to mark the time at which a job is allocated to a resource.

4.2 Coupled Model 'SIM_PROCESSOR'

This coupled model represents the simulation model specification. It consists of two coupled models namely 'FRAMING_TRADE_CONTRACTOR' and 'CITY_BUILDING_INSPECTOR', which represent the framing construction process and the framing inspection process respectively. These two coupled models are based on multi-server architecture. Multi-server architecture is a reusable modeling template defined within the DEVS framework; it is used to model a scenario where there is a process that is served by several servers or resources [Zeigler and Sarjoughian 2003]. The number of servers can be specified as an input variable while defining the architecture.

The coupled model 'FRAMING_TRADE_ CONTRACTOR' represents the construction and rework processes performed by the framing trade contractor. It is based on the multi-server architecture and consists of two types of atomic models namely multi-server coordinator and a simple processor. There are two processors (resources) defined within this coupled model and they represent the two framing crews namely 'crew1' and 'crew2'. The multi-server coordinator is represented by the atomic model 'MultiScoCon'. The responsibilities of the coordinator are (i) to queue the incoming entities in a FIFO based queue if all resources are busy; (ii) to send or assign jobs to one or more processors that are available (iii) to receive the completed jobs from one or more processors and send

them through the output port. The multi-server allows sending jobs simultaneously to multiple crews (e.g., crew1 and crew2) and thus all crews can process their work concurrently. It also supports receiving processed jobs that are completed at the same time.

The coordinator 'MultiScoCon' receives incoming entities through three input ports namely 'in', 'rework' and 'x'. Entities that come to the framing crew for construction work are received through the input port 'in'; entities that come for rework after a 'fail' in the first or succeeding inspections are received through the port 'rework'; entities that complete service with a framing crew are received through the port 'x'. The coordinator uses four output ports namely 'con_out', 'rw_out', 'y' and 'job_alloc'. The entities that have completed construction work are sent through 'con_out'; entities that have completed rework are sent through 'rw_out; 'y' output port is used to send entities to processors for service; the output port 'job_alloc' is used to send data to the transducer (calculates resource utilization) whenever a job is allocated to a processor.

The coupled model 'CITY_BUILDING_ INSPECTOR' represents the inspection process performed by the city building inspector. Inspection happens normally after the completion of each construction or rework process by a framing crew. This coupled model consists of 5 atomic models out of which 4 are simple processors and 1 is a multi-server coordinator. Each processor denotes an inspector resource and they are inspector:1, inspector:2, inspector:3 and inspector:4. The functionality of the coordinator (called 'MultiScoInsp') is similar to the coordinator 'MultiScoCon' defined in the coupled model 'FRAMING_TRADE_CONTRACTOR'.

The coordinator uses two input ports to receive incoming entities: (i) port 'in' to receive entities that come for inspection after construction or rework and (ii) port 'x' to receive entities that have completed inspection process by processors inspector:1 through inspector:4. Four output ports namely 'pass', 'fail', 'y' and 'frw' are used by the coordinator. Entities are sent to processor for inspection through port 'y'; entities that passes the inspection are sent through the port 'pass'; entities that fail in the inspection are sent through the port 'fail'; the port 'frw' is used to send entities to transducer in order to note the time at which an entity goes for first rework.

5 SIMULATION RESULTS AND DISCUSSION

The simulation model was run for two scenarios: (i) Scenario-1: the inspection pass rate is based on the actual data and it is 15%, 43%, 55%, 62%, 71%, and 100% in the first, second, third, fourth, fifth, and sixth framing inspections respectively and (ii) Scenario-2: the inspection pass rate is 100% in the first inspection.

Each scenario was run for 10 runs; the mean and variance of number of starts plot for each run was computed; and then the average mean and average variance of all 10 runs was computed. A representative number of starts plot was selected for each scenario in which the mean and variance of the selected run is close to the average mean and average variance. The number of jobs released to the drywall trade per week for Scenario-1 and Scenario-2 are shown in Figure 6 and Figure 7 respectively.

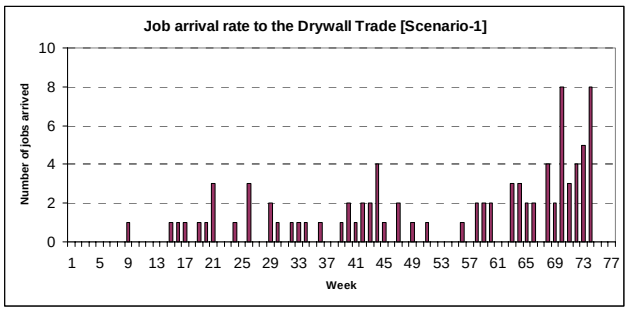

Figure 6: Job arrival rate to drywall trade: Scenario-1

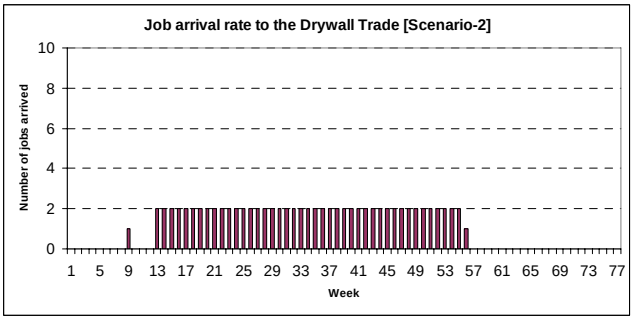

Figure 7: Job arrival rate to drywall trade: Scenario-2

The authors have run the same scenarios using the construction simulation software 'Simphony' and verified that the results of DEVSJAVA are same as the results of Simphony. This confirmed the correctness of the DEVS model implementation for the test problem.

The following are the inferences based on the simulation results:

- When the first inspection pass rate is 15% (Scenario-1) and there is limited crew (2 framing crews) on the upstream side, there is high variability in the workflow (number of lots per week) released to the downstream (drywall) trade.
- On the other hand, when there is 100% (Scenario-2) first time inspection pass rate and limited crew (2 framing crews) on the upstream side, the job arrival pattern to the drywall follows an even flow pattern. The two plots shown in Figure 6 and Figure 7 are compared quantitatively in Table 1.

From the results shown in Table 1, it can be noted that the inspection pass rate and the crew size of the upstream trade influences the workflow variability of the down-

stream trade. Poor inspection pass rate on the upstream increases the workflow variability and the spread of the number of starts on the downstream. Hence, improving the inspection pass rate on the upstream is critical to ensure better work flow to the downstream trade.

Table 1: Comparison of Scenario-1 and Scenario-2

Factor	Scenario-1	Scenario-2
Minimum number of starts per week	1	1
Maximum number of starts per week	8	2
Spread of number of starts	Week 9 to week 74 (66 weeks)	Week 9 to week 56 (47 weeks)
Mean	2.20	1.96
Sample Variance	2.93	0.04

6 CONCLUSIONS

This work demonstrates the application of DEVSJAVA in construction simulation through a detailed residential construction example. DEVS is a formal framework for discrete event modeling and simulation. It has a number of important features such as component-based hierarchical model development, support for modularity and reusability, parallel and distributed execution, and support to the modeler to develop custom components during simulation model development. User can perform the simulation in DEVSJAVA either using the STEP option (iterate through simulation clock event by event) or using the RUN option (fast mode).

The application of DEVSJAVA is not restricted to process and resource based simulation but it can be widely applied in many domains to conduct various types of simulation, for example agent-based simulation. Further DEVSJAVA can be integrated with other software tools/components such as MATLAB/Simulink to bring in additional functionality into the simulation. This affords extending construction simulation with optimization capability to help manage large-scale, complex enterprises [Sarjoughian et al 2005].

DEVSJAVA enables the modeler to develop an effective simulation model representation without any repetitions. The example problem discussed in this paper consists of one construction cycle and five rework cycles as shown in Figure 3. When this is implemented using a conventional discrete event simulation tool, the simulation model representation will be similar to Figure 3. This means that the same set of modeling elements will be repeated for each rework cycle. From the equivalent DEVSJAVA simulation model shown in Figure 5, it can be noted that the model does not have any repetitions of process flow for the five rework cycles. The logic is internally captured within the multi-server architecture used for the construction and inspection process.

A simulation environment/tool can be used for teaching, research or industry practice. Simulation tools for teaching or research usually do not provide sophisticated user interfaces. Use of simulation in industry practice needs the support for rapid model development and simulation in a short time. This requires (i) provision of basic modeling constructs through user interface to enable the modeler to drag and drop the modeling elements and to build simulation model and (ii) effective user interface for each modeling element to specify input to the simulation and read output after the simulation.

In general, when sophisticated user-interface features are provided to the modeler, the modeling freedom is often limited and it depends on the basic framework adopted in the simulation tool. On the other hand, where there is comprehensive modeling freedom available, user-interface features available to the modeler are often limited or none.

Development of simulation models in DEVSJAVA requires not only knowledge of sound modeling and simulation principles, but also varying degrees of Java programming language depending on the complexity of models being developed. That is, while the atomic and coupled models in DEVSJAVA have common (reusable) syntax, modelers can use the full power of the Java programming language to devise any level of complex logic as needed. To support modeling by domain experts (e.g., managers), the Scalable Entity Structure Modeler (SESM) framework has been developed to support visual and persistent model development (Sarjoughian, 2005). The SESM environment automatically generates complete simulation code for coupled models and partial simulation code for atomic models. Therefore, the combination of SESM and DEVSJAVA allows domain experts including construction experts to develop models while allowing researchers to carry out advanced research in the field of modeling and simulation, e.g., develop integrated simulation models where a combination of detailed processes, control, and financial planning is desired (Sarjoughian, et al, 2005).

ACKNOWLEDGMENTS

The study was supported in part by the National Science Foundation (NSF) through Grant Number 0333724. The authors wish to express their appreciation to City of Peoria for providing the data presented in this paper. The opinions, conclusions, and interpretations expressed in this paper are those of the authors, and not necessarily of NSF.

REFERENCES

Abourizk, S.M., D.W. Halpin, and J.D. Lutz. 1992. State of the Art in Construction Simulation. Proceedings of the 1992 Winter Simulation Conference, ed. J. J. Swain, D. Goldsman, R. C. Crain, and J. R. Wilson, 1271-1277.

AbouRizk, S.M., and D. Hajjar. 1998. A Framework for Applying Simulation in the Construction Industry. Canadian Journal of Civil Engineering, 25(3), 604-617.

AbouRizk, S.M. and Y. Mohammad. 2000. Simphony-An Integrated Environment for Construction Simulation. Proceedings of the 2000 Winter Simulation Conference, ed. J.A. Joines, R.R. Barton, K. Kang, and P.A. Fishwick, 1907-1914.

ACIMS. 2005. DEVSJAVA [online]. Available via <www.acims.arizona.edu/SOFTWARE/software.shtml> [accessed March 31, 2006].

Ballard, G. and G. Howell. 1998. Shielding Production: Essential Step in Production Control. Journal of Construction Engineering and Management, ASCE, 124(1), 11-17.

Ballard, G. 1999. Improving Work Flow Reliability. 7th Annual Conference of the International Group for Lean Construction (IGLC-7), Berkeley, California, USA, 26-28 July 1999.

Bashford, H.H., A. Sawhney, K.D. Walsh, and K. Kot. 2003. Implications of Even Flow Production Methodology for the U.S. Housing Industry. Journal of Construction Engineering and Management, ASCE, 129(3), 330–337.

Chang, D.Y. and R.I. Carr. 1987. RESQUE: A Resource Oriented Simulation System for Multiple Resource Constrained Processes. Proceedings of the PMI Seminar/Symposium, Milwaukee, Wisconsin, 4-19.

Chow, A. 1996. Parallel DEVS: A Parallel, Hierarchical, Modular Modeling Formalism and Its Distributed Simulator. SCS Transactions on Simulation, 13(2), 55-102.

City of Peoria. 2004. Record of Inspections Completed from October 2003 to September 2004. Peoria, Arizona.

Hajjar, D. and S.M. AbouRizk. 2000. Application Framework for Development of Simulation Tools. Journal of Computing in Civil Engineering, ASCE, 14(3), 160-167.

Hajjar, D. and S.M. AbouRizk. 2002. Unified Modeling Methodology for Construction Simulation, Journal of Construction Engineering and Management, ASCE, 128(2), 174-185.

Halpin, D.W. 1977. CYCLONE: Method for Modeling of Job Site Processes. Journal of the Construction Division, ASCE, 103(3), 489-499.

Halpin, D.W. and L.S. Riggs. 1992. Planning and Analysis of Construction Operations. Wiley Inter Science, New York, N.Y.

Ioannou, P.G. 1990. UM-CYCLONE Discrete Event Simulation System User's Guide. Technical Report, UMCE-89-12, Dept. of Civil and Environmental Engineering, University of Michigan, Ann Arbor, Michigan.

Liu, L.Y. and P.G. Ioannou. 1992. Graphical Object-Oriented Simulation System for Construction Process Modeling. Proceedings of the 8th Conference on Computing in Civil Engineering, ASCE, Dallas, Texas, 1139-1146.

Lluch, J.F. and D.W. Halpin. 1982. Construction Operation and Microcomputers. Journal of the Construction Division, ASCE, 108(1), 129-145.

Martinez, J.C. and P.G. Ioannou. 1994. General Purpose Simulation using Stroboscope. Proceedings of the 1994 Winter Simulation Conference, ed. J. D. Tew, S. Manivannan, D.A. Sadowski, and A.F. Seila, 1159-1166.

Martinez, J.C. and P.G. Ioannou. 1999. General Purpose Systems for Effective Construction Simulation. Journal of Construction Engineering and Management, ASCE, 125(4), 265-276.

Odeh, A.M., I.D. Tommelein, and R.I. Carr. 1992. Knowledge-Based Simulation of Construction Plans. Proceedings of the Eighth Conference on Computing in Civil Engineering, ASCE, Dallas, Texas, 1042-1049.

Paulson, B.C. 1978. Interactive Graphics for Simulating Construction Operations. Journal of the Construction Division, ASCE, 104(1), 69-76.

Sarjoughian, H.S. 2005. A Scalable Component-based Modeling Environment Supporting Model Validation. Interserive/Industry Training, Simulation, and Education Conference, 1-11, Orlando, FL.

Sarjoughian, H.S., D. Huang, W. Wang, D.E. Rivera, K.G. Kempf, G.W. Godding, H.D. Mittelmann. 2005. Hybrid Discrete Event Simulation with Model Predictive Control for Semiconductor Supply-chain Manufacturing. Proceedings of the 2005 Winter Simulation Conference, 256-266, Orlando, FL.

Sawhney, A. and S.M. AbouRizk. 1995. HSM - Simulation-based Project Planning Method for Construction Projects. Journal of Construction Engineering and Management, ASCE, 121(3), 297-303.

Sawhney, A., H.H. Bashford, K.D. Walsh, A. Mund. 2001. Simulation of Production Home Building using Simphony. Proceedings of the 2001 Winter Simulation Conference, ed. B.A. Peters, J.S. Smith, D.J. Medeiros, and M.W. Rohrer, 1521-1527.

Sawhney, A., H.H. Bashford, S. Palaniappan, K.D. Walsh, and J. Thompson. 2005. A Discrete Event Simulation Model to Analyze the Residential Construction Inspection Process. Proceedings of the 2005 ASCE International Conference on Computing in Civil Engineering, ed. L. Soibelman and F. Peña-Mora, July 12–15, 2005, Cancun, Mexico.

Teicholz, P. 1963. A Simulation Approach to the selection of Construction Equipment. Technical Report No. 26, The Construction Institute, Stanford University.

Tommelein, I.D., D. Riley, and G.A. Howell. 1999. Parade Game: Impact of Work Flow Variability on Trade

Performance, Journal of Construction Engineering and Management, ASCE, 125(5), 304–310.

Zeigler, B.P., T.G. Kim, and H. Praehofer. 2000. Theory of Modeling and Simulation. Second Edition, Academic Press, New York, NY.

Zeigler, B.P. and H.S. Sarjoughian. 2003. Introduction to DEVS Modeling and Simulation with JAVA. DEVSJAVA Manual, [online]. Available via <www.acims.arizona.edu/PUBLICATIONS/publications.shtml> [accessed March 31, 2006].

AUTHOR BIOGRAPHIES

SIVAKUMAR PALANIAPPAN is Graduate Research Associate in the Del E Webb School of Construction at ASU. He received his B.E. degree in Civil Engineering from Madurai Kamraj University, Tamil Nadu, India in 1997, and a M.S. degree in Building Technology and Construction Management from I.I.T. Madras, Chennai, India in 2002. He is a doctoral student in the Department of Civil and Environmental Engineering at ASU since January 2004. His research interests are production home building, construction simulation, and automation in construction. His e-mail address is <plsiva@asu.edu>.

ANIL SAWHNEY is Associate Professor in the Del E Webb School of Construction at Arizona State University (ASU). His research and teaching are in the areas of integration of information technology in construction and construction process modeling and simulation. He teaches courses in Construction Planning and Scheduling, Information Technology in Construction, Construction Productivity, and Design and Analysis of Construction Operations. Dr. Sawhney is the Co-Director of the Housing Research Institute (HRI) at ASU. He has published over 50 technical papers in journals and conferences. His e-mail address is <Anil.Sawhney@asu.edu> and his web address is <http://construction.asu.edu/faculty/sawhney>.

HESSAM S. SARJOUGHIAN is Assistant Professor of Computer Science and Engineering at Arizona State University, Tempe and Co-Director of the Arizona Center for Integrative Modeling and Simulation. His research includes simulation modeling theories and methodologies with emphasis on multi-formalism modeling, agent-based and collaborative modeling, simulation-based design, and software architecture. His educational emphasis has lead in the Online Master of Engineering in Modeling and Simulation at ASU. For more information visit <http://www.acims.arizona.edu> and <http://www.eas.asu.edu/%7Ehsarjou/index.htm>.