Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, eds

METHODOLOGY FOR SYNCHRONIZING DISCRETE EVENT SIMULATION AND SYSTEM DYNAMICS MODELS

Hani Alzraiee
Tarek Zayed
Osama Moselhi

Concordia University
1455 De Maisonneuve West,
Montreal, Qu, H3G 1M8, CANADA

ABSTRACT

Integrating Discrete Event Simulation (DES) and System Dynamics (SD) simulation methods require synchronization of their simulation clocks to ensure that actions are executed in an orderly manner. This paper presents a synchronization methodology for integrating DES and SD models. A hybrid simulation-based method consisting of SD components at the higher decision level and DES components at the lower decision level is expected to benefit from the developed method. The proposed methodology integrates DES and SD models on a single platform, which enhances the simulation of construction operations. It consists of three elements: 1) advancing mechanism, 2) DES advancing algorithm, and 3) messages sequence mechanism. The paper provides a description of the three elements of the synchronization method. An illustrative preliminary experiment that utilizes DES and SD engines is presented to demonstrate the use of the developed synchronization method and to illustrate its capabilities.

1 INTRODUCTION

The decision-making process is extremely essential part of any construction operation. Simulation is widely regarded as an effective tool for analysis of construction operations because of its ability in handling the complexity and uncertainty inherent in construction processes (Halpin et al. 2003). It has powerful ability that assists project managers to test various scenarios of project execution plans; such analysis assists construction decision makers in making informed decisions (Zayed and Halpin 2000). Construction operations comprise of discrete and continuous variables. The discrete variables arise from the operational level while the continuous variables arise from the global level. The interactions between those two types of variables are inevitable at the execution stage. Better understanding of these interactions mechanism can inevitably enhance planning and executing the construction operations. DES and SD are the most widely used simulation methods to simulate variables of construction operations in which DES is used to model variables of discrete nature while SD is used to model continuous variables (Brailsford and Hilton 2001).

For effective planning and management of construction operations, the global/context effects on local decisions at operational level need to be estimated and involved in decision-making process. Global/context effects refer to the impacts of the policy decision and context on operation outcomes. For instance, how changes in staffing, overtime, and scope affect productivity of different project operations. Interactions between the global and operation level are not captured through using DES or SD separately. Hybrid simulation that integrates DES and SD can enhance the simulation process through capturing the neglected interactions between continuous and discrete variables of the operation being modeled. However, adopting hybrid simulation for construction needs effort to solve the system state updates. This is be-

978-1-4673-4781-5/12/$31.00 ©2012 IEEE

cause DES updates system states and advances simulation clock based on the occurrence of events while SD updates system states and advances simulation clock based on time interval elapse. This paper is dedicated to develop the background and the tools needed for synchronizing DES and SD on a single platform.

The next section presents an introduction to DES and SD modeling methods and characteristics, followed by an overview of the available synchronization methods and limitations. Then, the elements of the proposed synchronization method are presented and discussed. Finally, the paper concludes with preliminary experiment, result, and conclusion of the proposed method.

2 SYSTEM STATE UPDATE IN DES AND SD MODELS

System states update is the procedures followed to update the variable's states and then advancing the simulation clock. Simulation time in DES models advances at the occurrence of events, subsequently system states are updated. Events in the DES model mostly occur at unequal time points. In SD models, simulation time advances at equal time intervals, and then, system states are updated at the end of the intervals. The behavior of DES and SD models in state updates are demonstrated in Figure 1. Figure 1 (a) demonstrates a typical behavior of DES model, where simulation time advances from event say E1 to E2 using discrete jumps. Continuous system such as SD model updates their states at equal time intervals as shown in Figure 2 (b). The whole simulation time is divided into equal time intervals, and then, states are updated of the intervals. Figure 1 (c) represent a hybrid system behavior that incorporates discrete and continuous variables interactions (Pritsker et al. 1997)

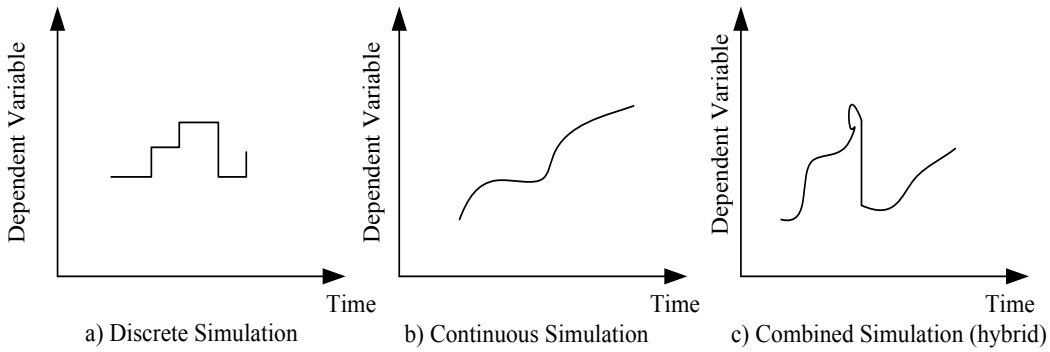

Figure 1: Simulation Modeling Techniques.

3 SYNCHRONIZATION (TIME MANAGEMENT)

Time management or synchronization means execution of events in distributed simulation in a correctly manner and ensures that repeated executions of a simulation with the same inputs produced exactly similar results (Fujimoto 2003). Time management algorithms broadly fall into two categories termed conservative and optimistic synchronizations. These time approaches are mainly developed to serve the execution of multi-simulation programs on multiprocessor computing platforms (Parallel Simulation) or to serve executing simulations on geographically distributed computers interconnected via a network (Distributed simulation). In both cases, the execution of a single simulation model (composed of several simulation programs) is distributed over multi computers (Fujimoto 2001).

3.1 Conservative Time Management (CTM)

CTM means, that synchronization algorithm takes precautions to avoid local causality constrains. These mechanisms usually assume that simulation consists of a collection of *logical processes* (LPs) that com-

municate by exchanging time-stamped messages or events. The goal of the synchronization mechanism is to ensure that each LP processes events in timestamp order; this requirement is referred to as the *local causality constraint* (Fujimoto 2001; Fujimoto 1999). For instance, if LP is at simulation time 20, conservative protocol guarantees that no event has LP simulation time less than 20. The first algorithm of generating CTM was developed by (Bryant 1977; Chandy and Misra 1978). Each LP sends a message with non-decreasing time stamp to support interactions between models in which communication network ensures that messages are received in the same order as they were sent from the LP. Messages organized in order (first-in-first-out) are same as schedule events execution. Simulation process starts with the event of lowest time step. Local events scheduled within the LP can be handled by having a queue within each LP. When the messages queue of any model become empty, the process becomes deadlock and cannot proceed any more. Null messages are used to avoid this deadlock. They have timestamps that cannot create any event or update state. Null messages introduce key property called *lookahead* concept. If the LP is at simulation time T, and it guarantee that any message sent in the future will have timestamp at least T+L, then the LP is said to have a *lookahead* of L period. Null messages algorithm results in an excessive number of null messages, which is not efficient (Fujimoto 2001). However, this method generates a high computational overhead to ensure sequence time advancement. In addition, it needs enormous memory requirements (Mattern 1993). This type of time management is used where the causality constraint violation is likely not permitted.

3.1.1 Time Bucket (TB) Synchronization Method

The TB concept is one of the conservative time synchronization algorithms that follow the conservative approach. It was developed to synchronize distributed DES simulators (Fujii et al. 1993). The TB is based on dividing the overall DES model simulation time into small time intervals called TB, thereafter, allowing the models or simulators of the distributed simulation to interact or interface at the end of the time interval (Steinman 1990). The time bucket size should be large enough to overcome any overhead computations and small enough to capture any radical changes in the system state. One of the main drawbacks of the TB is the inability to capture event states that have event time less than the TB size.

3.2 Optimistic Time Management (OTM)

Unlike CTM, OTM methods allow violation of local causality constraint but they allow detection of violations and then recover from them. The OTM methods have two important features, first, they have tendency to exploit a greater degree of parallelism in executing processes, and secondly, synchronization mechanism is more transparent to the application program than CTM methods. OTM methods require more computations than CTM methods as they need to recover from violation of causality constraint. The Time Warp (TW) algorithm (Jefferson 1985) is the most well known optimistic method of time synchronization. It allows free simulation time advancement and when causality violation occurs (processing event of higher timestamp before receiving event with lower timestamp) then TW rolls back and reprocesses these events in timestamp order, with restoring the state that existed prior to violation. Anti-messages sent to the same queue cancel the previously sent messages (Fujimoto 2001). Two problems arise in this situation; first, certain computations I/O of operations cannot be rolled back secondly, computations consume more memory between sending, un-sending, and roll back of messages. Both problems are solved by Global Virtual Time (GVT), which is a lower bound on timestamp on any future rollback, any data stored for LP before GVT will be destroyed. TW sometimes is overly optimistic (Steinman 1993) and involves many process cancelations.

The conservative and optimistic time synchronization approaches were developed specifically to synchronize distributed discrete event simulation models that share similar state updating and time advancing mechanisms, hence, they are discrete event oriented methods. Running simulation models on multi processers and networks is a complex task and time consuming for modeling construction operations. Dis-

tributed simulation is best used for simulating large models, which is not the case in construction. In the proposed synchronization method, the DES and SD models are integrated on a single computation platform (not distributed one) which certainly preserves the distinct features of DES and SD models. This allows both models to be executed without concerns of one method prevails the other.

4 PROPOSED SYNCHRONIZATION METHOD FOR INTEGRATING DES AND SD

The synchronization method described in this research aims at simplifying the process of integrating DES and SD models on a single platform. This can assist in adopting and enhancing the current hybrid simulation practice in construction. The method consists of three elements: (1) advancing mechanism and Time Bucket, (2) DES advancing algorithm, and (3) messages sequence mechanism. The following section provides detailed explanation to these three elements.

4.1 Advancing Mechanism and Time Bucket

The TB concept explained in section (3.1.1) is utilized to develop the proposed synchronization mechanism. Firstly, the simulation time length (L) of the hybrid simulation model is divided into equal time intervals called TB, secondly, at the end of these time intervals, interfacing of variables and data exchange between variables take place, finally simulation advances to TB_2 where it resumes from the end of TB_1 as demonstrated in Figure 2. The aforementioned three steps are explained in details in the following paragraphs.

Initially, at the start of simulation time of length L, hybrid simulation system initializes the simulation clocks for DES and SD engines as well as for the modules variables. Now, the simulation clock of the models is advanced from the start of TB_1 to the end of TB_1, where $TB = TB_1 = TB_2 = \ldots. TB_n$. When the simulation time reaches the end of TB_1, simulation progress is halted to allow interfacing of variables and exchanging of data between DES and SD modulus as shown in Figure 2. This causes a new system state called *hybrid DES_SD state* to arise. The data flow directions between the DES and SD modules are predetermined based on the selected hybrid model structure and through selected interface variables within the hybrid model boundary. Variables of the hybrid modules that are not selected to participate in the interface process, will not participate in the integration process mechanism. However, the effects of the updated data of the interface variables will be propagated to influence those exempt variables from interfacing, as a result, all variables in the model are expected to be impacted. When the process of interfacing and data exchanging is accomplished at the end of TB_1, the DES_SD resumes advancing simulation time to TB_2 and again at the end of TB_2 interfacing of variables take place. This process continues on this mechanism until the simulation time reaches to the end of simulation length L.

TB should be small enough to capture any significant changes in the system states and large enough to discard overhead computations. The rationale behind using the TB concept is that SD updates states at equal and pre-known time intervals while DES updates states at the occurrence of the events. In DES, events generally occur at unequal time points with difficulty of predicting these time points. Therefore, it is easier to trace model states at stipulated time points, which is guaranteed by TB, than tracing model states based on the occurrence of events where the time of occurrence is not known in advance. This approach ensures that every simulation method (DES or SD) will preserve its unique characteristics in computations. The TB size is proposed to be equal to the SD model STEP TIME as shown in equation (1) and Figure 2.

$$TB \text{ (hybrid model)} = \text{SD model STEP TIME} \qquad (1)$$

Despite the method of TB Synchronization is simple to implement, yet there are some drawbacks inherited in this method, such as, TB is not applicable for DES models that have events of zero time. Such events exist in computer and manufacturing fields where control theory is dominant, but this hardly no-

ticed in construction since no redundant operations of zero duration exist. The other limitation, concerns with TB size, which should be large enough to facilitate low synchronization overhead, and small enough to capture any significant change in the states of the hybrid model variables. Furthermore, it is an essential precaution not to allow two consecutive events, say (E_1) and (E_2), to have total events time less than the TB size. If this occurs, the (E_1) will not be captured in the proposed hybrid computation process. This is because initializing, occurring, and vanishing of (E1) take place before the next earliest stipulated interfacing process (i.e. end of TB_1). Therefore, any selected TB size should ensure that no two consecutive events occur within single TB interval.

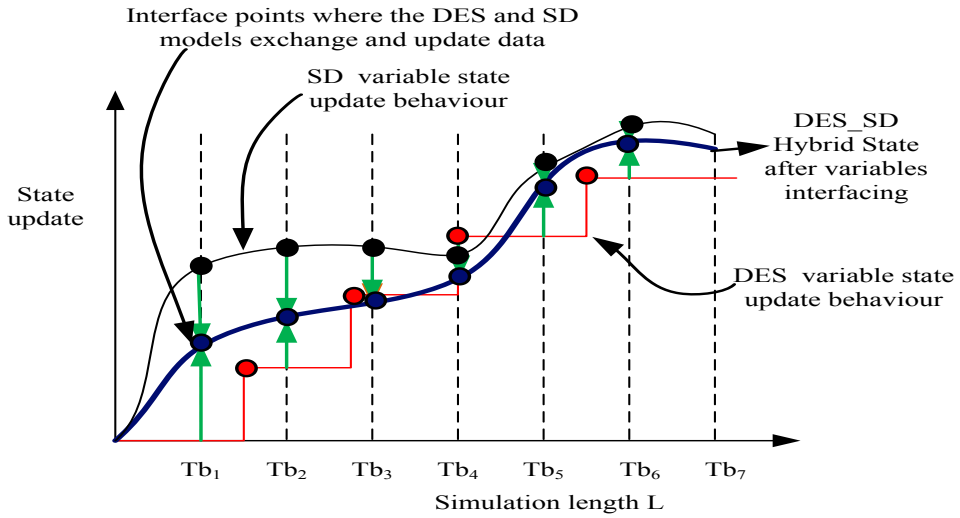

Figure 2: Synchronization of DES and SD Modules Using Time Bucket.

Concisely, the proposed synchronization method is time oriented rather than being event oriented one. It is expected not to face any implications with SD method, since both are emerged from the same concept, but for DES method, further arrangements are needed. As explained earlier, DES simulation is driven by the occurrence of events that have unpredictable occurrence time points. Therefore, an algorithm that breaks the DES simulation length L into time buckets and facilitates the full integration process of DES and SD models is inevitable. This algorithm should make the DES time advancing mechanism compatible with the proposed time synchronization method. Unless DES is capable of preparing variable`s states needed by SD at the point where interface of modules occur, SD modules will not be able to update its variables from the project operational level that was modeled using DES method. Hence, results of the hybrid model are likely to be doubtful.

4.2 DES Advancing Algorithm

The second component of the proposed synchronization method is to make the DES simulation advancing mechanism compatible with the proposed method. An algorithm that divides the DES simulation length (L) into intervals, facilitates integration, and resumes the simulation is needed. The developed algorithm is depicted in Figure 3. Initially for the DES engine to start advancing the simulation time, a condition such as, the required resources and entities should be available at the start of TB_1. Now the simulation is in position to start advancing at the beginning of TB_1, if entities seize the required resources, then all data of active resources and entities in the simulation model are read and saved, otherwise, idle resources data are read and saved. If the process involving the active resources has not finished processing the entity at the end of TB_1, then pause DES simulation clock advancement, save all data and perform DES and SD modules interfacing. Otherwise, eliminate saved resources and entity data and return to re-allocate next process and its entities, attributes and resources. In DES model, events having occurrence time less than

the TB_1 finish their processes before the interfacing of variables can take place, hence their data are not captured in the next earliest scheduled interfacing, but their effects are propagated to second event. Therefore, in order to avoid events that start and finish before the end of the TB, it is advised to set the SD TIME STEP less than or equal to the lowest expected event time in the model.

After the interfacing is accomplished, all saved data at end of TB_1 (interface time point) of active or idle resources are used by the DES engine for next round of computations that begins by the commencement of TB_2 and continue in the same sequence explained in TB_1. The time point between end of TB_1 and start of TB_2 is the point where the simulation clock resumes progressing of model simulation. The algorithm continues until the model reaches the initially set simulation time of length L and then terminates the simulation run.

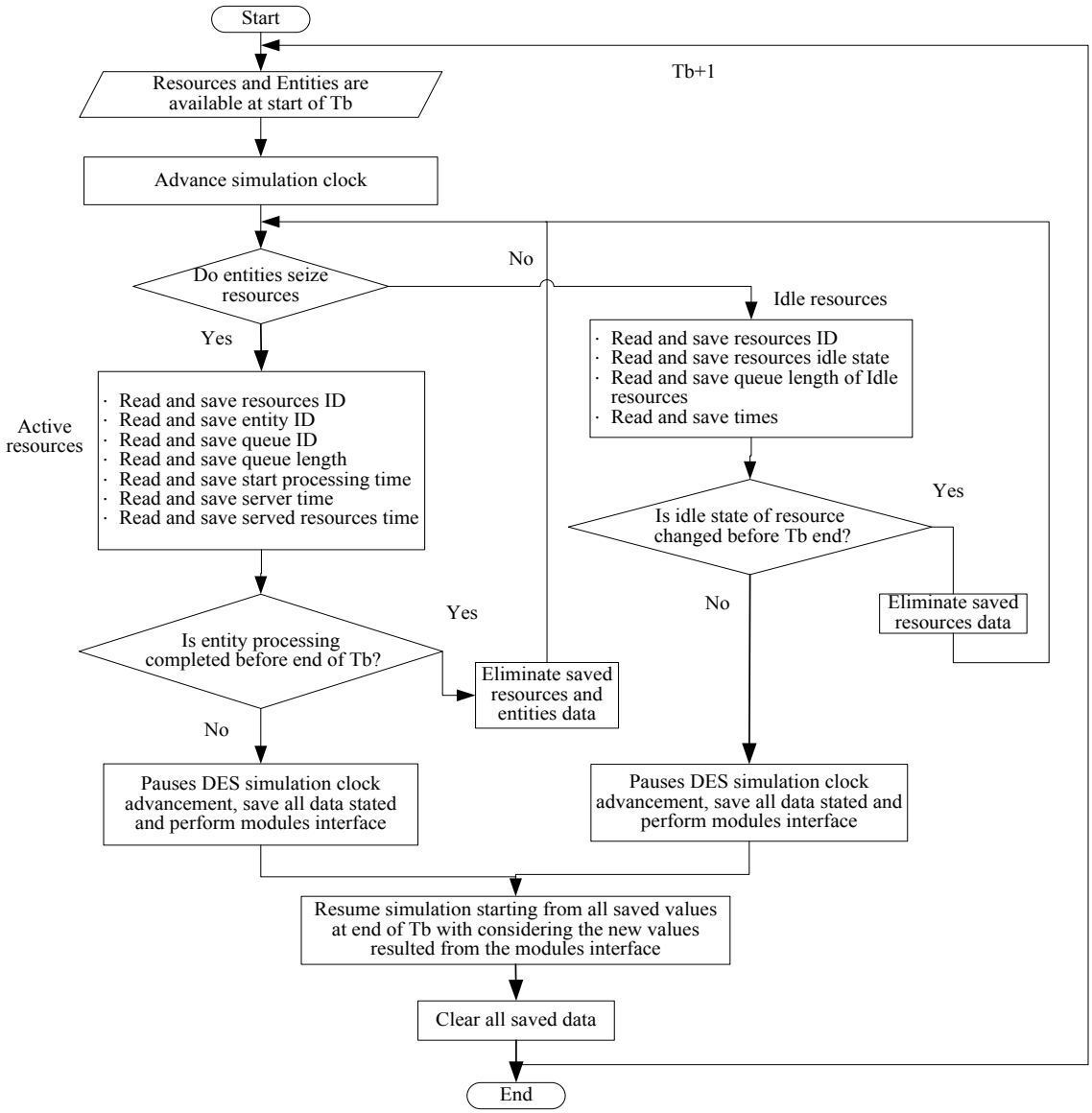

Figure 3: Simulation Time Advancing Algorithm for DES Model.

4.3 Messages Sequence Mechanism

The third and last component of the proposed synchronization methodology is describing the sequence of messages between the modules (DES and SD) and the executer (program that executes the synchronization algorithm). These messages will carry the commands responsible on implementing the proposed synchronization mechanism as shown in Figure 4. The bottom right Figure is the architecture of the developed hybrid simulation system, its details are beyond the scope of this paper. Messages (1), (2), (3), and (4), executer confirms the initialization of the DES and SD models and makes those two components ready to start advancing the simulation clock. Message (5), the SD provides initial values such as TB_1 size that triggers the executer to start advancing simulation clock (6). Message (7) advances simulation clock of DES model to the end of TB_1, at end of TB_1, states of DES model's variables are read and saved. Message (8), the needed values for interfacing process are read and managed by the executer. Message (9), the executer starts interfacing variables by exporting the new values of variable to SD, this message flags the end of the computations processed in TB_1. Starting from message (10), the messages (5), (6), (7), (8), and (9) contents are repeated until the simulation time length is elapsed.

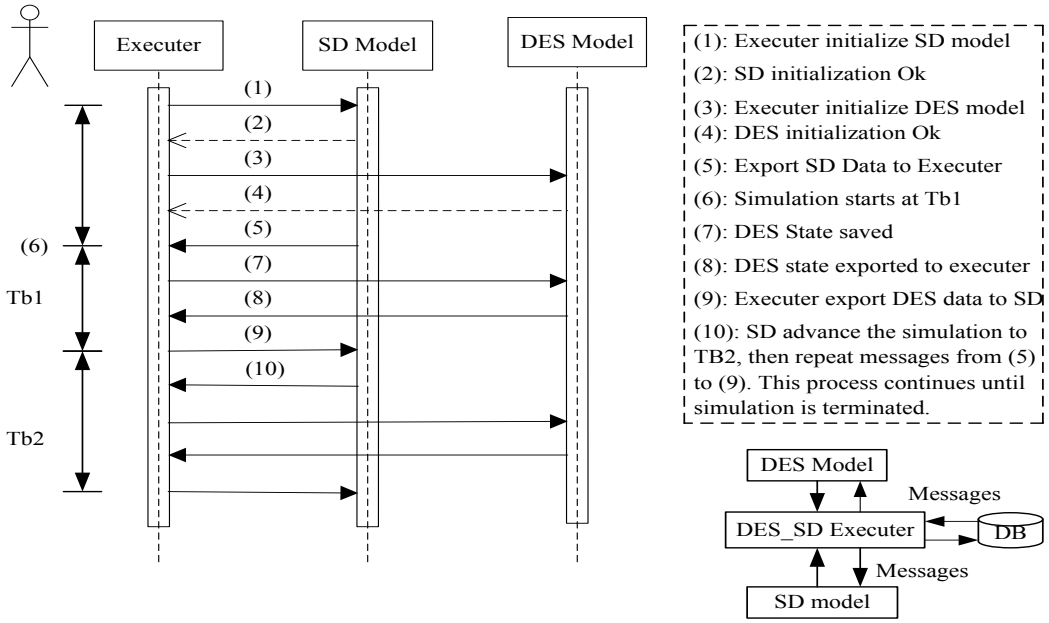

Figure 4: Messages Sequence between the Hybrid System Components.

5 PRELIMINARY EXPERIMENT

To demonstrate how the proposed synchronization method can be applied, a hybrid model of a quarry project was developed based on a case study adopted from Marzouk and Moselhi (2003). It involves hauling excavated rocks from a quarry site to construction site of a dam in Northern Quebec. The quarry project involves four cyclic operations, namely loading, hauling, dumping, and returning. The project scope was hauling 225,000 ton of rocks from quarry site to the construction site. Different variables that arise from context level such as scope change, schedule pressure, space limitation, dumping space, and equipment reliability were modeled using SD model as shown in Figure 5

A Visual Basic (VB) application was developed in Microsoft environment to communicate with the STROBOSCOPE (Martinez et al. 1994) and Vensim software systems (Ventana Systems Inc). The operational aspects of construction were modeled using DES method in STROBOSCOPE environment while the strategic/context aspects of the construction operations were modeled using SD method in Vensim en-

vironment. In order to access both simulation engines, VB uses the ActiveX Data Object to connect to the simulation engine in STROBOSCOPE. This allows gaining access to and control of the dynamically changing values of the variables considered in the simulation process, while Vensim's Dynamic Link Library (DLL) is utilized for access and control of the SD model. The model developer or end-user interacts with the VB application to control the input/output operations.

Six variables (interface variables) in the hybrid model are selected to participate in the synchronization process as shown in Figure 5. Characteristics of those interface variables are shown in Table 1. The selected interface variables will either receive or send data updates needed by the model variables. For instance, in Figure 5, Arrow (1) indicates the added scope change while executing the project. The added scope of work is represented in the SD model and the quantity of the scope change is exported to the DES initial scope of work. This type of modeling is not feasible when using the current DES modeling in construction. Arrow (2) considers the effects of space limitation modeled using SD method on the DES productivity model. Arrow (3) reflects the actual fleet productivity after considering the effect of equipment maintenance and failure on the actual material productivity. This type of data exchange between the DES and SD alleviates the limitation associated with DES in capturing the context of the operation being modeled.

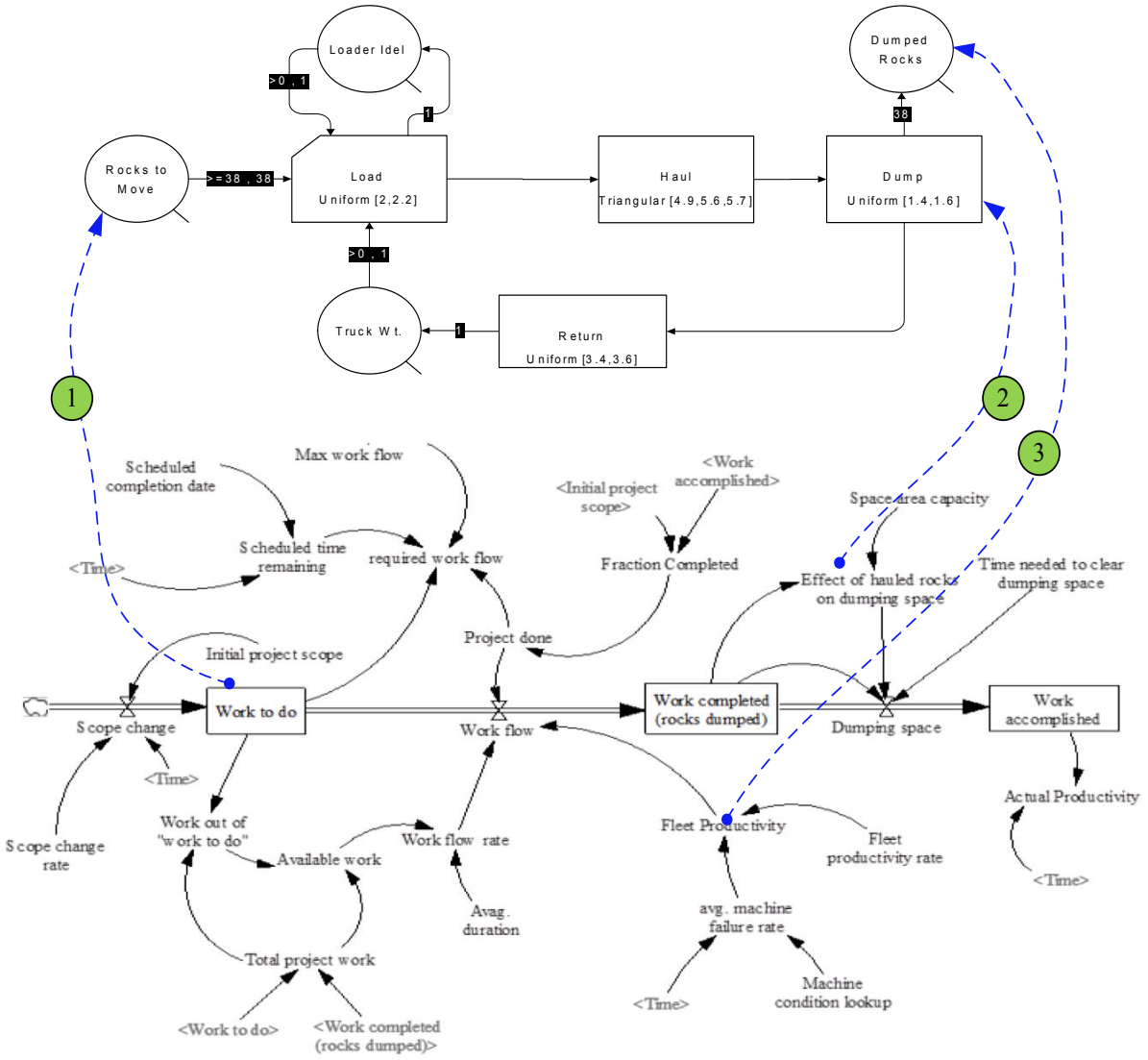

Figure 5: Hybrid Simulation Model Consisted of DES and SD Models.

To capture any minor changes in the model behavior, TB was set to 0.94 minute, which is equal to the STEP TIME in the SD model. This means that interfacing between DES and SD for all variables considered will occur every 0.94 minute.

Table 1: Selected Interface Variables in DES and SD Models for Synchronization.

Sender		**Receiver**		
Interface variable	**Module Type**	**Interface Variable**	**Module Type**	**Note**
Scope change	SD	Initial scope	DES	Arrow (1) in Figure 4.
Effect of hauling on dumping space	SD	Dumping	DES	Arrow (2) in Figure 4.
Equipment Maintenance and failure rate	SD	Actual productivity	DES	Arrow (3) in Figure 4.

The DES model was ran initially alone without interaction with the SD model. This was done to compute the DES model productivity and duration, which were 365 ton/hr and 617 hr respectively. Thereafter, making use of the step time (TB), the hybrid simulation environment was established through the iterative interfacing of DES and SD selected variables. The simulated productivity was found to fluctuate between 303 ton/hr to 340 ton/hr as shown in Figure 6 (a, c) compared to 365 ton/hr in the DES model. As well, the dumping space constraint resulted in limiting productivity to 320 ton/hr. This constraint caused a reduction in the fleet productivity as shown in Figure 6 (b). The impact of this constraint was injected in the "dumping" process in the DES model. The simulation conducted shows how interfacing process between simulation variables in the DES and SD models can be achieved and how the outcomes of DES can be impacted when considering influential variables of the operation.

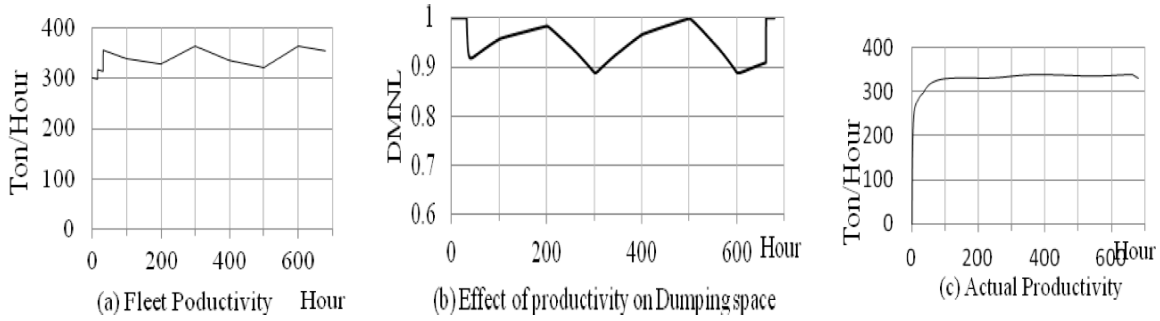

Figure 6: Hybrid Model's Results.

6 CONCLUSION

A novel method of synchronizing DES and SD simulation methods has been presented. The approach and the components of the method were outlined. The developed method provides a step toward for enhancing hybrid simulation in construction. Previous synchronization methods were mainly designed for use in parallel or distributed simulations and more specifically to integrate simulators or simulation models of DES nature. To test the proposed method, a preliminary experiment was conducted successfully through

utilizing STROBOSCOPE and Vensim engines. Six interface variables in DES and SD model were selected to participate in the synchronization through mapping the data between DES and SD.

While this research demonstrates the feasible integration of the DES and SD simulation methods on single platform, for simulation of construction operations, considerable work is still needed to capture the full potential of that integration. This includes conducting more comprehensive synchronization experiments, more interface variables to participate in the synchronization process, and more possible hybrid structure testing. Further enhancement of the synchronization application possibly can be achieved by using VB.NET interface or developing new DES engine.

ACKNOWLEDGMENT

The authors would like to thank Natural Sciences and Engineering Research Council of Canada (NSERC) for funding and supporting the present research.

REFERENCES

Brailsford, S., & Hilton, N. 2001. "A comparison of discrete event simulation and system dynamics for modeling health care systems." *Proceedings of the 26th Meeting of the ORAHS Working Group 2000,* Glasgow, Scotland. 18-39.

Bryant, R. 1977. Simulation of packet communication architecture computer systems. Computer Science Laboratory, MIT, Cambridge, MA

Chandy, k., Misra, J. 1978. "Distributed simulation: A case study in the design and verification of distributed programs". IEEE Transactions on Software Engineering, 5(5), 440-452

Fujii, S., H. Tsunoda and M. Yamane, 1993. A Study on Distributed Simulation Model of Virtual Manufacturing system under CIM Environment. Production Research 1993, ed. V. Orpana and A. Lukka, 281-282, Elsevier Science Publication.

Fujimoto, R. M. 2003. "Parallel simulation: Distributed simulation systems". *Proceedings of the 2003 Winter Simulation Conference, Edited by D. Ferrin, D. J. Morrice, P. J. Sanchez and S. Chick, 124-13.* New Orleans, LA: Institute of Electrical and Electronics Engineers, Inc

Fujimoto, R. M. 2000. Parallel and Distributed Simulation Systems: Wiley Interscience.

Fujimoto, R. M. 1999. "Parallel simulation: Distributed simulation systems". *Proceedings of the 1999 Winter Simulation Conference, Edited by D. T. Sturrock, G. W. Evans, P. A. Farrinton and H. B. Nemhard, 122-131.* Atlanta, GA: Institute of Electrical and Electronics Engineers, Inc

Fujimoto, R. M. 2001. "Parallel and Distributed Simulation Systems", *Proceedings of the 2001 Winter Simulation Conference, Edited by M. Rohrer, D. Medeiros, B. A. Peters and J. Smith, 147-157, Arlington, VA:* : Institute of Electrical and Electronics Engineers, Inc

Halpin, D., Jen, H., & Kim, J. 2003. "A construction process simulation web service." *Proceedings of the 2003 Winter simulation conference, Edited by D. Ferrin, D. J. Morrice, P. J. Sanchez and S. Chick,*1503-1509. New Orleans, LA: Institute of Electrical and Electronics Engineers, Inc

Jefferson, D. 1985. Virtual Time. The ACM Transactions on Programming Languages and Systems. Journal of Parallel and Distributed Computing, 18(4), 423-434.

Marzouk, M., & Moselhi, O. 2003. Object-oriented simulation model for earthmoving operations. *Journal of Construction Engineering and Management*, 129(3), 173-182.

Mattern, F. 1993. "Efficient Algorithms for Distributed Snapshots and Global Virtual Time Approximation*". Journal of Parallel and Distributed Computing*, 18(4), 423-434.

Pritsker, A. 1997. "Simulation with visual slam and Awesim." New York, NY.: John Wiley & Sons Inc.

Steinman, J. 1990. Multi-node test bed: A distributed emulation of space communications for the strategic defense system. Pittsburgh, 21(3), 1111-1115.

Zayed, T., & Halpin, D. 2001. Simulation of Concrete Batch Plant Production. *Journal of Construction Engineering and Management*, 127(2), 132-141.

AUTHORS BIOGRAPHIES

HANI ALZRAIEE is a Doctoral candidate in Department of Building, Civil, and Environmental Engineering at Concordia University in Montreal, Canada. He holds a M.S. degree in Construction and Project Management and B.E. in Civil Engineering. Mr. Alzraiee has 10 years of professional experience in building and infrastructure design and project management. He is currently a research assistant at the Construction Automation Lab at Concordia University. His current research focuses on hybrid discrete event simulation and system dynamics integration for modeling and simulating construction operations on a single platform. In addition, he is working on research projects for assessing the trenchless rehabilitation methods used to rehabilitate Quebec's Province Sewer System. His email address is h_alzrai@encs.concordia.

TAREK ZAYED Dr. Zayed is an Associate Professor, Dept. of Building, Civil & Environ. Eng., Concordia University, Montreal, Canada. Dr. Zayed has a Ph.D., M.Sc., and B.Sc. in Construction Engineering and Management. He conducted research in infrastructure management, simulation and AI applications in construction, underground system performance, and life cycle cost analysis. He has 22 years of professional experience working in the construction industry training and in academic posts in USA, Canada and abroad. Dr. Zayed is extending his expertise in infrastructure management systems to develop life cycle cost analysis for constructing and rehabilitating highway, water, and sewer systems. In addition, he developed models that optimized budget allocation and risk assessment for infrastructure systems. His web page can be found via <http://users.encs.concordia.ca/~zayed>

OSAMA MOSELHI Dr. Moselhi is Professor of Engineering in the Department of Building, Civil and Environmental Engineering at Concordia University. He served as Department Chair and Executive Advisor to the Dean of the Faculty in graduate studies and research and in planning and appraisal of graduate programs. Since joining Concordia in 1985, after a decade of industry experience, Dr. Moselhi supervised and co-supervised over 50 Masters and Ph.D. graduates, authored and co-authored over 300 scientific publications. His industry experience spans tall buildings, bridges, harbor facilities, and nuclear power plants. He is recipient of numerous honors and awards; including the prestigious Walter Shanly Award in recognition of "outstanding contributions to the development and practice of construction engineering in Canada" Dr. Moselhi served as international consultant on academic affairs and on construction projects in Canada, USA, and the Middle East. His research interest encompasses planning, procurement, resource allocation, tracking and control of construction projects, with a focus on risk management, productivity analysis, management of construction claims and development of decision support systems embracing information technology, remote sensing, web-enabling and spatial technologies. His email address is moselhi@cbs-engr.concordia.ca