OBJECT-ORIENTED TECHNOLOGY FOR ENHANCING ACTIVITY-BASED MODELING FUNCTIONALITY

Jonathan Jingsheng Shi

Department of Civil and Architectural Engineering
Illinois Institute of Technology
3201 South Dearborn Street
Chicago, IL 60616-3793, U.S.A.

ABSTRACT

Computer simulation is a powerful tool for analyzing and optimizing real-world systems with a wide range of successful applications. However, construction engineers have difficulty in mastering such techniques, particularly in modeling a construction process because existing simulation modeling methods require the user to manipulate many modeling elements in order to construct a simulation model for the process. The activity-based construction (ABC) modeling method provides an easy-to-use approach with one single element, e.g., activity, for modeling a construction process. This paper discusses the integration of the object-oriented technology with ABC. The object-oriented technology allows each activity contain six classes of attributes describing the characteristics of the activity including duration, logical sequence, resource requirement, etc. These attributes are relevant to individual activities, and can be easily understood by an engineer who is familiar with the process. After incorporating these attributes, powerful modeling and simulation functions are realized. The integration greatly enhances the capability of ABC so that the enhanced ABC system has achieved both flexibility and ease-of-use essential for a construction simulation system.

1 BACKGROUND

Computer simulation has been growing rapidly with the advancement of computer technology and has become one of the most popular techniques for practitioners in operations research and in the manufacturing industry (Pidd 1984, Paul 1991). Halpin (1993) proposed that process-based modeling and simulation should be developed as the scientific base for the construction engineering and management program because of the process-oriented nature of the industry. Despite extensive research achievements, the application of simulation in the construction industry is still very limited, mostly at universities (Halpin and Martinez 1999).

Modeling is the process of understanding a real-world system and constructing a corresponding mathematical/logical model for the system. It is the first task in conducting simulation. Modeling represents the most important and the most difficult stage for a successful simulation project (Law and Kelton 1991). Three modeling approaches have been developed for discrete simulation (Pritsker et al. 1997): the event-based, activity-based, and process-based.

Special-purpose simulation systems are effective alternatives for non-specialists. Shi and AbouRizk (1997) proposed the resource-based modeling (RBM) method for specific types of construction processes, they also developed an automated modeling system for earthmoving operations (1998). A special-purpose simulation development environment, named Simphony, was developed at the University of Alberta (Hajjar and AbouRizk 1999). Several special-purpose simulation systems were developed based on Simphony (AbouRizk et al. 1999).

The activity-based construction (ABC) modeling and simulation method was developed (Shi 1999) with one single modeling element, namely activity. ABC provides a graphical tool for modeling a real-world system by using activities. Constructing an ABC simulation model for a process is similar to constructing an activity-on-node (AON) network diagram (Shi 2000). With the popularity of network techniques in the industry, ease of use can be fully achieved in ABC. The targeted users of the ABC are construction engineers who may not be simulation specialists.

Another essential need parallel to ease of use is the modeling functionality of the simulation system. In other words, how to achieve construction modeling and simulation functions in an easy-to-use system? This paper incorporates the object-oriented technique into ABC to enhance the ABC functionality.

2 ACTIVITIES

An object-oriented data structure can be formulated to detail the characteristics of an activity. For the purpose of modeling and simulation of construction operations, the activity object contains six classes, each of which contains attributes, as expressed as follows:

$$\text{Activity}\{D, L, R, P, O, Q\} \qquad (1)$$

where,

- D = Activity duration. Duration is expressed by either a constant value or a random distribution.
- L = Logical relationship. Logical relationship contains the order of sequence of the current activity with its preceding activities and its succeeding activities, i.e., L = {*preceding*, *succeeding*}. Both *preceding* and *succeeding* can be specified with a value of either "AND" or "OR". If an "OR" relationship is specified for succeeding activities, a probability corresponding to each succeeding activity must be specified. See section "modeling logical relationship" for detail.
- R = Resource. An activity cannot start its construction until all its required resources are available and are allocated to the activity. After the completion of its execution, the engaged resources must be released accordingly, i.e., R = {*Required_resources*, *releasing_rule*}. *Required_resources* = {*name, unit*}, which lists required resources and corresponding units. *releasing_rule* = {*name, destination*}, which defines the releasing destination of a given resource. See section "modeling resources" for detail.
- P = Processing entities. P = {NI, NP}. NI = iterations needed for the activity before a processing entity is released (equivalent to "combine" function); NP = number of processing entities is released after each operation of the activity (equivalent to "split" function). The default values for both parameters are 1. See section "modeling simulation entities" for detail.
- O = Operational characteristics. O = {*Priority, NC, Max*}. *Priority* = Priority for competing shared resources; *NS* = maximum concurrent operations of the activity; *Max* = maximum cycles of operation of the activity, default infinite. See section "modeling operational characteristics" for detail.
- Q = Production measures. Q = {*Quantity*, *scale*}. *Quantity* = Production quantity measured from each cycle of operation of the activity; *Scale* = measure unit, e.g., m^3. See section "collecting activity statistics" for detail.

The user is required to supply all activity objects with corresponding attribute values based on the characteristics of individual activities. Defaults are used to reduce unnecessary inputs from the user. The attribute values are incorporated into modeling and simulation to achieve the expected functionality discussed as follows.

3 COMMON MODELING FUNCTIONS

3.1 Modeling the Idle State

Traditional modeling systems use idle state (e.g., queue nodes) to model entity waiting. For instance, if activity X requires four resources namely R1, R2, R3, and R4 to start its execution, the four needed resources are modeled by four individual idle states (circles) as shown in Figure 2(a). Idle times of the four resources are collected at the four waiting locations respectively.

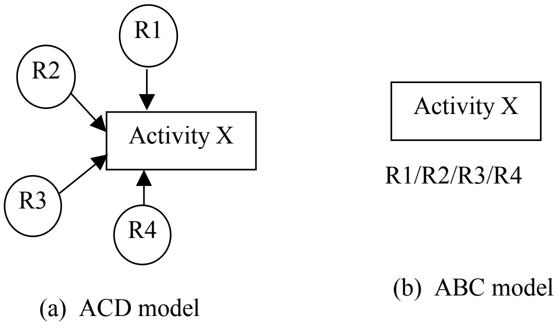

Figure 1: A Contrast of ACD and ABC

When required resources are treated as the activity attributes, one element is needed to model the same situation as shown in Figure 1(b). Therefore, removal of the idle states does not affect the modeling functionality, but it reduces the modeling elements from five in the ACD model to only one in the ABC model. Resources wait at the activity until the activity starts. Resource idle times are calculated using the following equation:

$$WT = ST - TA \qquad (2)$$

where, WT = waiting time of resource at activity X; ST = time that the activity starts; TA = time that the resource arrives at the activity.

From an engineer's perspective, required resources are an integral part of executing the activity. It is not natural to model the required resources separately from the activity. Our experience shows that the separation of an activity from its required resources has created difficulty for construction engineers to effectively grasp existing modeling methods.

3.2 Modeling Logical Relationships

An activity (e.g., X) may be preceded and be succeeded by multiple activities. Activity X may start when one entity reaches it ("OR" relationship), or it may not start until all of its preceding activities have released at least one entity to it ("AND" relationship). The number of active entities released from preceding activity I to activity X is expressed by EN(X, I). While activity X is being scanned for feasibility of starting, the user specified logical attribute value is used. The simulation algorithm is illustrated in Figure 2. The difference in the logical relationship will be reflected at the logically allowed start time (T1) for the activity.

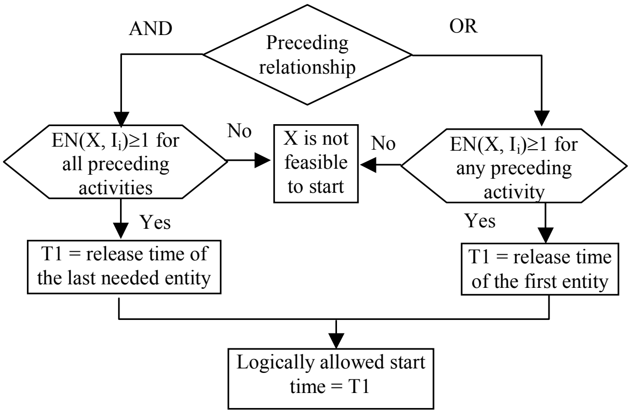
AND
Preceding relationship
OR
EN(X, I_i)≥1 for all preceding activities
No
X is not feasible to start
No
EN(X, I_i)≥1 for any preceding activity
Yes
Yes
T1 = release time of the last needed entity
T1 = release time of the first entity
Logically allowed start time = T1

Figure 2: Checking for Preceding Relationship

After its completion, activity X may release one single entity to one of its succeeding activities ("OR" relationship), and it may also release one entity to each of its succeeding activities ("AND" relationship). For "OR" relationship, a corresponding probability for taking each activity must be specified with $\{p_1, p_2, \ldots, p_n\}$ with $\Sigma p_i = 1$. The algorithm is schematically illustrated in Figure 3.

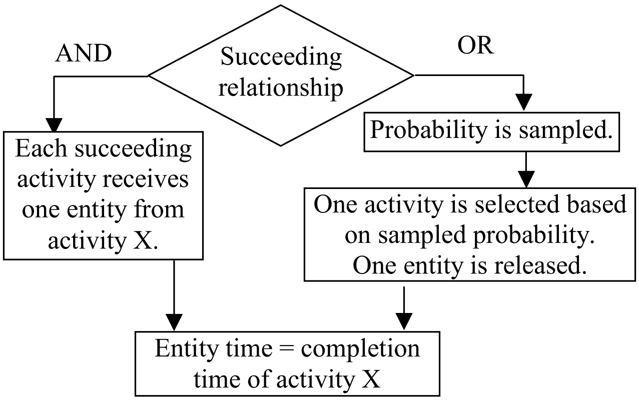
AND
Succeeding relationship
OR
Probability is sampled.
Each succeeding activity receives one entity from activity X.
One activity is selected based on sampled probability. One entity is released.
Entity time = completion time of activity X

Figure 3: Schematic Flow for Succeeding Relationship

Default "AND" relationships are used for both preceding and succeeding activities in the critical path method (CPM). Existing modeling systems model the succeeding relationship in the same manner as ABC does, but cannot conveniently model the "AND" or "OR" preceding relationships. Instead, additional elements (e.g., SELECT node) are used for such function. It is a tedious task to model a CPM diagram using existing simulation systems.

3.3 Modeling Resources

A resource is fully identified by its location (L) and its last event time (ET) over time. It stays at the current location until the next event which changes the location and status of the resource as shown in Figure 4.

$$TR = EX - ET \qquad (3)$$

where, TR = time that the resource stays at the current location L; L = 0 if the location is the resource pool, and L = number of the activity if the location is an activity; EX = next event time; ET = last event time.

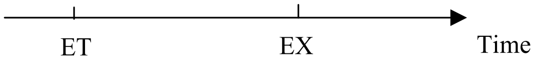

Figure 4: Discrete Event Times of a Resource

There are two possible locations for a resource: an activity (identified by the number of the activity) or the resource pool. A resource at the pool is always idle. Its state is the same as its involved activity while it is at an activity. A resource changes its state when its involved activity changes state. For instance, a resource changes from idle to busy if its involved activity starts operation; it changes from busy to idle if the activity completes operation.

The class of resource attributes of an activity contains required resources, required units, and rules governing the release of the resources after the activity completes its operation. While an activity is logically feasible for starting, the availability of required resources is then checked. If all needs are satisfied, the required resources are allocated and the activity starts its construction.

After the completion of the activity, occupied resources may stay at the activity, to be released to its succeeding activities, or to be released to the resource pools depending on the user-specified attribute value.

If an activity requires two or more classes of resources, a matching situation exists among resources. If all of these resources can be released at the same time after the completion of the activity, this situation is defined as the even matching. On the other hand, an uneven matching exists if involved resources cannot be released at the same time after the completion of the activity. Existing modeling methods model an uneven match by splitting and/or combining simulation entities. For example, a truck arrives at a warehouse and is loaded with six packages of product. Loading is conducted by a crew with one package at one time. Using existing modeling approach, a truck entity is first split into six entities to match the six packages. Then, six entities after loading are combined into one to recover the true truck entity as shown in Figure 5(a).

To model an uneven matching in ABC, a fraction (e.g. $1/x$) is used to express the required unit of a resource, and x is the iterations of the activity before the resource can be released. The same truck-loading example can be modeled by a required truck unit of 1/6 as shown in Figure 5(b), which means that the truck entity will stay at the activity until the activity has executed six iterations, but the crew is released after each operation of the activity. To correctly simulate an uneven match, the involvement of resource at an activity is initialized when the resource is allocated to the activity and be added by 1 after the completion of the activity; the resource is released after the given iterations have been reached.

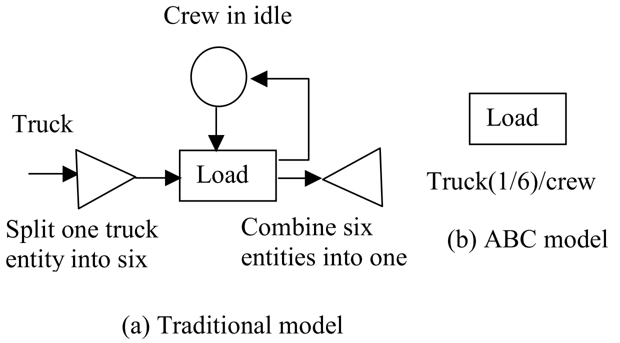

Figure 5: Modeling Unbalanced Matching

A contrasting of the two models suggests that the ABC can achieve the same modeling function by reducing two extra elements.

3.4 Generating Entities

Generating entities is the common simulation function. Additional elements are usually used by existing simulation systems to meet the need. In ABC, entities may be generated by an activity through the following manners.

3.4.1 Generation of processing entities

A processing entity is generated and flows to its succeeding activity if no resource entities are released in the same direction. Activity X is succeeded by Activities Y. If Activity X does not release its resources to Activity Y, a processing entity is generated and flows to Activity Y each time after the completion of Activity X.

3.4.2 Free activity

A free activity does not have constraints (e.g., no resource requirement and no preceding activities) to limit its execution. It can be conveniently used to generate processing entities for modeling customer arrival or other similar situations. If the maximum concurrent operation of the activity is limited to one, the activity can start again for generating the next entity after completing the execution of the activity. The activity duration models the interval between two adjacent entities.

3.5 Modeling a Simulation Entity

There are two classes of simulation entities: resource entities and processing entities. Resource entities are initialized by the user at the resource pool or activities before simulation starts. They change location and status as simulation proceeds. Processing entities are automatically generated during simulation as previously discussed.

If activity Y succeeds activity X and some resources engaged in activity X are released to activity Y, these resource entities will be released to activity Y after the completion of activity X. A resource entity represents a unit of resource or a unit of resource temprarily combined with a processing entity. For instance in an earthmoving process, the truck entity released from activity “dump” to activity “return” is a pure resource entity; the resource entity released from activity “load” to activity “haul” is the combination of “truck” and “soil”. A resource entity changes its location and status over time, but does not change its nature. One piece of equipment is always one piece wherever or whenever it may be in the process. Therefore, when resource entities are released whether or not they are combined with processing entities, the entity units remain unchanged, i.e., the released units equal to the received units.

A processing entity may have different physical meanings at different activities. For instance, the soil pushed in one cycle by a tractor serves two loader buckets in an earthmoving process. There is no common resources between activities “push” and “load”. Processing entities are generated after each pushing operation. In this case, one processing entity at “push” equals to two entities at “load”. On the other hand, five truck loads of soil dumped at activity “dump” are spreaded in one batch by a spreader. This time, five processing entities at “dump” equal to one entity at “spreading”. “Split” and “combine” are the additional modeling functions used by exsiting simulation systems to balance processing entities.

ABC uses two attribute values in the activity for the purpose. The iterations of the activity are accumulated since the last release of entity and no processing entity is released until given iterations have reached. The algorithm is illustrated in Figure 6.

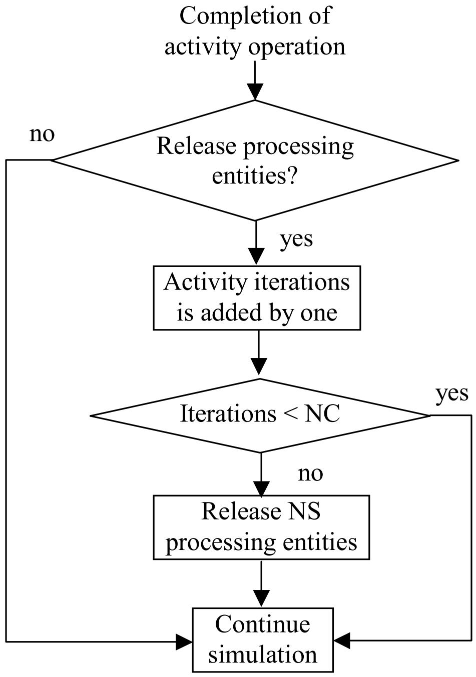
Completion of activity operation
Release processing entities?
no
yes
Activity iterations is added by one
Iterations < NC
yes
no
Release NS processing entities
Continue simulation

Figure 6: Flow Diagram for Releasing Processing Entity

For above "push" and "load" example, NI = 2, so that two processing entities are released to "load" after each pushing operation. For the "dump" and "spread" situation, NP = 5, so that one processing entity is released after "dump" has accumulated 5 iterations.

3.6 Modeling Operational Characteristics of Activities

The class of the characteristic attributes of an activity contains three parameters: priority of the activity, maximum concurrent operations, and maximum iterations of operation. Priority is used when activities are selected for starting. The activity with a higher priority will be given a higher priority for competing resources.

An activity may allow multiple concurrent operations. For instance, two services can take place simultaneously if there are two bank tellers in the process. A maximum of two trucks may pass a bridge simultaneously. In existing modeling systems, the maximum concurrent operations are limited by the number of severs by default although this may not always be true. Sometimes, the concurrent operations allowed for an activity are less than the number of severs, typically for shared resources. For example, two crews are shared by three activities in a construction process, but they may not work on the same activity concurrently because of site limitations. This requirement is modeled by giving a value of the maximum concurrent operations for the activity. The concurrent active states of an activity are collected over time and must not exceed the given value at any time. If the active states of an activity have reached the allowed value, the activity is not selected for entering the feasible activity set for start.

The maximum target iteration of operations will be used to terminate the operation of the activity; therefore, simulation of the process can be controlled from the activity as necessary. Existing modeling systems control the expected iterations of an activity by additional elements.

4 THE SIMULATION ALGORITHM

The effective simulation algorithm corresponding to the ABC modeling is the activity scanning method (Pidd 1992). The three stage activity-based simulation algorithm was presented (Shi 1999) with: 1) selecting activity, 2) advancing simulation, and 3) releasing simulation entities. Above discussed functions are incorporated into the three stages accordingly.

5 AN EARTHMOVING EXAMPLE

For the earthmoving example, the ABC simulation model is constructed and is shown in Figure 7, which contains 6 activity elements. The first activity "prepare soil pile" requires resources "dozer" and "ground", which are not required by the succeeding activity "load". The dozer can push 6 m^3 of soil in each cycle of operation. The bucket capacity of the loader is 3 m^3. Therefore, two processing entities are generated following the execution of "prepare soil pile".

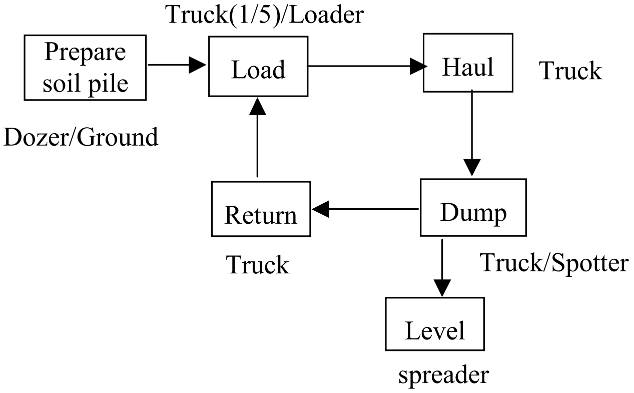

Figure 7: ABC Model for the Earthmoving Process

Activity "Load" has an uneven matching situation, at which five loading passes are needed to fill one truck box. A truck entity is held at "Load" until the activity has executed five iterations. After a truck entity is released, activity "Haul" starts immediately. When the truck entity arrives at the dumping area, it requested a "Spotter" and activity "Dump" starts.

Activity "Dump" has two succeeding activities. After its execution, the "truck" entity is released to activity "Return", and one processing entity is ready to be released to activity "Level". However, five truck loads are leveled in one batch. Therefore, the processing entity is held at activity "Dump" until five iterations have reached. Production values are collected at activities "Dump" and "Level".

Simulation starts at time 0, i.e., Tnow = 0. After scanning all activities, only "Prepare soil pile" can start. It is selected, its operation is advanced, and its completion time is sampled. No other activities can be scheduled for construction because "Prepare soil pile" is in operation and no entity has been released to activate activity "Load". Simulation goes to Stage 3. Activity "Prepare soil pile" is the only activity that can be selected for releasing entity. It is selected, and two processing entities are generated to activity "Load".

Simulation goes back to stage 1 to scan activities and choose one for construction. This time, both activities "Prepare soil pile" and "Load" are feasible candidates. Because both activities have the same start time, "Prepare soil pile" is chosen again for execution. Its operation is simulated with its completion time sampled again. Simulation process continues to scan for feasible start activities. Only "Load" is feasible because "Prepare soil pile" is still active. Activity "Load" is selected, and advanced. No other activity can be scheduled for start. Simulation goes to Stage 3 to release simulation entity. Either "Prepare soil pile" or "Load" may be selected. The activity with the least completion time is selected. If "Load" is selected, resource "Loader" should be released, but "Truck" should stay because five iterations are needed to fill its box.

The process continues until expected simulation time has reached. The simulation results can then detail activity productions, utilization and distribution of resources. The hourly production cost of the fleet is calculated by adding up the hourly operating costs of involved resources, and unit production cost can be computed accordingly. After experimenting with different number of trucks, the simulation results can be summarized in Figure 8. The chart shows that five trucks should be chosen for the process.

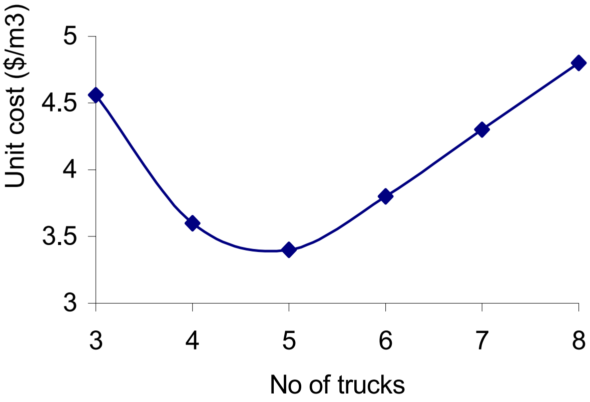

Figure 8: Summary of Simulation Results

6 CONCLUSION

This paper discussed the integration of the object-oriented technology with the activity-based construction (ABC) modeling and simulation method. Each modeling element, e.g., activity, contains six classes of attributes describing the characteristics of the activity including duration, logical sequence, resource requirement, etc. These attributes are relevant to individual activities, and can be easily understood and specified by an engineer who is familiar with the process. These attributes are then incorporated into modeling and simulation to achieve needed functionality. The enhanced ABC has achieved both flexibility and ease-of-use essential for a construction simulation system.

REFERENCES

AbouRizk, S.M., K.C. Er, J.Y. Ruwanpura, and S. Fernando. 1999. Special purpose simulation template for utility tunnel construction. *Proceedings of the 1999 Winter Simulation Conference.* Farrington, P. A., H. B. Nembhard, D. T. Sturrock, and G. W. Evans, eds. IEEE, Piscataway, N.J. 948-955.

Chang, D. 1987. *RESQUE*, Ph.D. thesis, University of Michigan, Ann Arbor, Mich.

Hajjar, D. S.M. AbouRizk. 1999. Simphony: an environment for building special purpose construction simulation tools. *Proceedings of the 1999 Winter Simulation Conference.* Farrington, P. A., H. B. Nembhard, D. T. Sturrock, and G. W. Evans, eds. IEEE, Piscataway, N.J. 998-1006.

Halpin, D.W. 1977. CYCLONE - A method for modeling job site processes. *Journal of Construction Division*, ASCE. 103(3). 489-499.

Halpin, D.W. 1993. Process-based research to meet the international challenge. *ASCE Journal of Construction Engineering and Management*, 114(2). 417-425.

Halpin, D.W., and L.H. Martinez. 1999. Real world applications of construction process simulation. *Proceedings of the 1999 Winter Simulation Conference*. Farrington, P. A., H. B. Nembhard, D. T. Sturrock, and G. W. Evans, eds. IEEE, Piscataway, N.J. 956-962.

Huang, R., A.M Grigoriadis, and D. W. Halpin. 1994. Simulation of cable-stayed bridges using DISCO. *Proceedings of 1994 Winter Simulation Conference.* J.D. Tew, S. Manivannan, D.A. Sadowski, and A.F. Seila, eds. IEEE, Piscataway, N.J. 1130-1136.

Ioannou, P.G. 1989. UM_CYCLONE User's Guide. Dept. of Civil Engineering, The University of Michigan, Ann Arbor, Mich.

Law, A.M. and W.D. Kelton. 1991. *Simulation Modeling and Analysis.* 2nd edition. McGraw-Hill, Inc. New York.

Liu, L. Y. and P. G. Ioannou. 1992. Graphical object-oriented discrete-event simulation system. *Proceedings of 1992 Winter Simulation Conference.* Swain, J. J., D. Goldsman, R. C. Crain, and J. R. Wilson, eds. IEEE, Piscataway, N.J. 1285-1291.

Martinez J.C. and Ioannou P.G. 1999. General purpose systems for effective construction simulation. J*ournal of Construction Engineering and Management*, ASCE, 125(4), 265-276.

Paul R.J. 1991. Recent development in simulation modelling. *Journal of Operations Research Society*, 42(3), 217-226.

Paul, R.J. 1993. Activity cycle diagrams and the three Phase Method. *Proceedings of the 1993 Winter Simulation Conference.* IEEE, Piscataway, N.J. 123-131.

Paulson, B.C. Jr. 1978. Interactive graphics for simulating construction operations. *Journal of Construction Division*, ASCE, 104(1), 69-76.

Pidd, M. 1984. Computer simulation for operational research in 1984. *Developments in Operational Research* (R.W. Eglese and G.K. Rand, Eds.). Pergamon Press, Oxford.

Pidd, M. 1992. *Computer Simulation in Management Sciences*. 3rd edition. Chichester: John Wiley and Sons.

Pritsker, A. A. B., O'Reilly, J.J., and LaVall, D.K. 1997. *Introduction to Simulation and SLAM-II.* 2nd edition. John Wiley and Sons, Inc., New York, N.Y.

Shi, J. and S.M. AbouRizk. 1997. Resource-based modeling for construction simulation. *Journal of Construction Engineering and Management*, ASCE. 123(1). 26-33.

Shi, J. and S.M. AbouRizk. 1998. An automated modeling system for simulating earth-moving operations. *Journal of Computer-Aided Civil and Infrastructure Engineering*. 13(2). 121-130.

Shi, J. 1999. Activity-based construction (ABC) modeling and simulation method. *Journal of Construction Engineering and Management*, ASCE. 125(5), 354-360.

Shi, J. 2000). Activity based construction (ABC) simulation modeling vs. critical path method (CPM). *Construction Congress VI,* ASCE, Orlando, FL. 131-139.

Teicholz, P. 1963). A simulation approach to the selection of construction equipment. Tech. Rep. No. 26, The Construction Institute, Stanford University, Palo Alto, Calif.

Tocher, K. D. 1963. *The Art of Simulation*. London: English Universities Press.

Tommelein, I. D., and A. M. Odeh. 1994. Knowledge-based assembly of simulation networks using construction designs, plans, and methods. *Proceedings of the 1994 Winter Simulation Conference.* J.D. Tew, S. Manivannan, D.A. Sadowski, and A.F. Seila, eds. IEEE, Piscataway, N.J. 1145-1158.

AUTHOR BIOGRAPHY

JONATHAN JINGSHENG SHI is an Associate Professor in the Department of Civil and Architectural Engineering at Illinois Institute of Technology in Chicago. He received his Ph.D. in Civil Engineering from the University of Alberta in 1995. His research interests in simulation are focused on automated modeling, simulation, and optimization in construction applications. His other research interests include neural networks, construction scheduling and planning method, and construction productivity and improvement. His email address is `<jonathan.shi@iit.edu>`.