ANL/DIS/CP-95375

Ownership Transfer for Non-Federate Object and Time Management in Developing an HLA Compliant Logistics Model

Zhian Li, Charles M. Macal, and Michael R. Nevins
Decision and Information Sciences Division
Artificial Intelligence Section
Argonne National Laboratory
9700 South Cass Avenue
Building 900
Argonne, IL 60439
(630) 252-3388, (630) 252-3767, (630) 252-6091
li@dis.anl.gov, macal@dis.anl.gov, nevins@dis.anl.gov

Keywords:
logistics, port, cargo, non-federate object, object ownership transfer, simulation synchronization

ABSTRACT: *A seaport simulation model, PORTSIM, has been developed for the Department of Defense (DOD) at Argonne National Laboratory. PORTSIM simulates the detailed processes of cargo loading and unloading in a seaport and provides throughput capability, resource utilization, and other important information on the bottlenecks in a seaport operation, which are crucial data in determining troop and equipment deployment capability. There are two key problems to solve in developing the HLA-compliant PORTSIM model. The first is the cargo object ownership transfer problem. In PORTSIM, cargo items, e.g. vehicles, containers, and pallets, are objects having asset attributes. Cargo comes to a seaport for loading or unloading. The ownership of a cargo object transfers from its carrier to the port and then from the port to a new carrier. Each owner of the cargo object is responsible for publishing and updating the attributes of the cargo object when it has the ownership. This creates a unique situation in developing the PORTSIM federate object model, that is, the ownership of the object instead of the attributes needs to be changed in handling the cargo object in the PORTSIM federate. The ownership management service provided by the current RTI does not directly address this issue. The second is the time management issue. PORTSIM is an event-driven simulation that models seaport operations over time. To make PORTSIM HLA compliant, time management must be addressed to allow for synchronization with other simulation models. This paper attempts to address these two issues and methodologies developed for solving these two problems.*

1. Introduction

Model interoperability and reusability have become compelling issues in recent years, as more and more sophisticated simulation models are being developed for all disciplines of science and engineering. It is especially important to the simulation models developed for the Department of Defense (DOD), because various simulation models often need to be integrated together to form a joint simulation exercise. The High Level Architecture (HLA) provides a common interface architecture and implementation standard to the development of interoperable and reusable simulation models. To assure interoperability, HLA compliance is in order for all models developed for DOD. Logistics problems are important components of military operations and exercises. Logistics models should therefore be HLA compliant so that they are readily available for joint warfighting simulation integration[1].

A seaport simulation model, PORTSIM[2], has been developed for DOD at Argonne National Laboratory. PORTSIM simulates the detailed processes of cargo loading and unloading in a seaport and provides port throughput capability, resource utilization, and other important information on the bottlenecks in a seaport operation. These are crucial data in determining troop and equipment deployment capability and should be of interest to federate(s) in federation simulation involving troop and equipment deployment and mobility analysis and, therefore, be published by the PORTSIM federate.

A key problem to solve in developing an HLA-compliant PORTSIM model is the cargo object ownership transfer issue. In PORTSIM, cargoes,

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

e.g. vehicles, containers, and pallets, are objects with asset attributes. Cargo comes to a port for processing and then leaves perhaps with new attribute values and on new carriers. The port federate must take over the ownership of the cargo objects when the cargo arrives, change the attributes if necessary, publish and update the cargo objects and their attributes, and release the ownership when the cargo leaves. This creates a unique situation in developing the HLA federate object model, that is, the ownership of the object instead of the attributes needs to be changed in handling the cargo object in the PORTSIM federate so that the values of the cargo object's attributes can be updated by the new owner. The ownership management service provided by the current RTI does not directly address this issue[3], [4]. A technique for solving this problem is in order.

The second important problem to be solved is the time management issue. PORTSIM is an event-driven simulation that models seaport operations over time. It operates on an internal simulation clock. Reports on information such as port cargo processing queues, resource utilization rates, number of facilities available/busy, number of berths available, and the overall port throughput capability, are generated on a fixed time schedule according to the internal simulation clock. Because the current version of PORTSIM does not accept external time coordination, in making PORTSIM HLA compliant, the time management issue must be addressed to allow for synchronization with other simulation models.

This paper attempts to address these two issues and the methodologies developed for solving these two problems. Section 2 describes the fundamentals of the seaport simulation model, PORTSIM, and the HLA-compliant PORTSIM federate to be developed. Section 3 details the technical approaches to the solutions of the non-federate object ownership transfer and the time management issues for the PORTSIM federate.

2. PORTSIM Federate

Seaports are critical nodes in the transportation networks linking the United States and final destinations around the globe. Ideally, a complete military equipment deployment analysis should include all relevant network nodes and links. To perform this analysis, simulation models for all of these different nodes and networks must be integrated. To achieve this goal, all of the models involved have to be interoperable. HLA provides a common implementation standard for the development of interoperable models. Making PORTSIM HLA compliant is an effort toward this goal.

2.1 PORTSIM

PORTSIM is a discrete-event time-stepped simulation model for seaport military equipment deployment throughput capability analysis. PORTSIM simulates the detailed seaport embarkation and debarkation operations to provide critical information on port resource utilization, bottlenecks, cargo queues, as well as the detailed cargo queue status. PORTSIM takes two types of transportation inputs: land transportation inputs and waterway transportation inputs. The land inputs are prepared by a military equipment characteristics database query and transportation asset requirement system TARGET (Transportability Analysis Reports Generator), with a detailed break down of transportation unit and transportation asset requirements. The waterway input is provided by the JFAST ship database system[5] or possibly another source. The waterway input includes ship name, fleet designation, ship class, containers, and breakbulk capacities, etc., sufficient for identifying the profile of the arriving ship. Finally, the port infrastructure data for PORTSIM is furnished by the MTMCTEA (Military Traffic Management Command Transportation Engineering Agency) database[6].

The embarkation and debarkation processes in a seaport operate in different operational procedures. The embarkation operation processes a cargo item in four steps: reception, staging, ship berthing, and loading. The debarkation process on the other hand operates in four different steps, i.e., ship berthing, unloading, staging, and clearance. The cargo types that PORTSIM can process are: (1) vehicles, (2) containers, and (3) palletized cargoes. Each cargo type is processed differently in the port according to the transportation asset it arrives with as well as the type of operation, i.e., embarkation or debarkation. A cargo item with a railroad transportation asset, for example, will go to the interchange yard first, whereas the entry point for cargo with a highway transportation asset is the port gate. For debarkation, a ship must dock on a berth first in order for a cargo item to be unloaded. PORTSIM differentiates the embarkation and debarkation processes along with the cargo types to ensure the accuracy of the simulations. Figure 1 depicts the operational flow of the port convoyed-vehicle-embarkation process as simulated in PORTSIM.

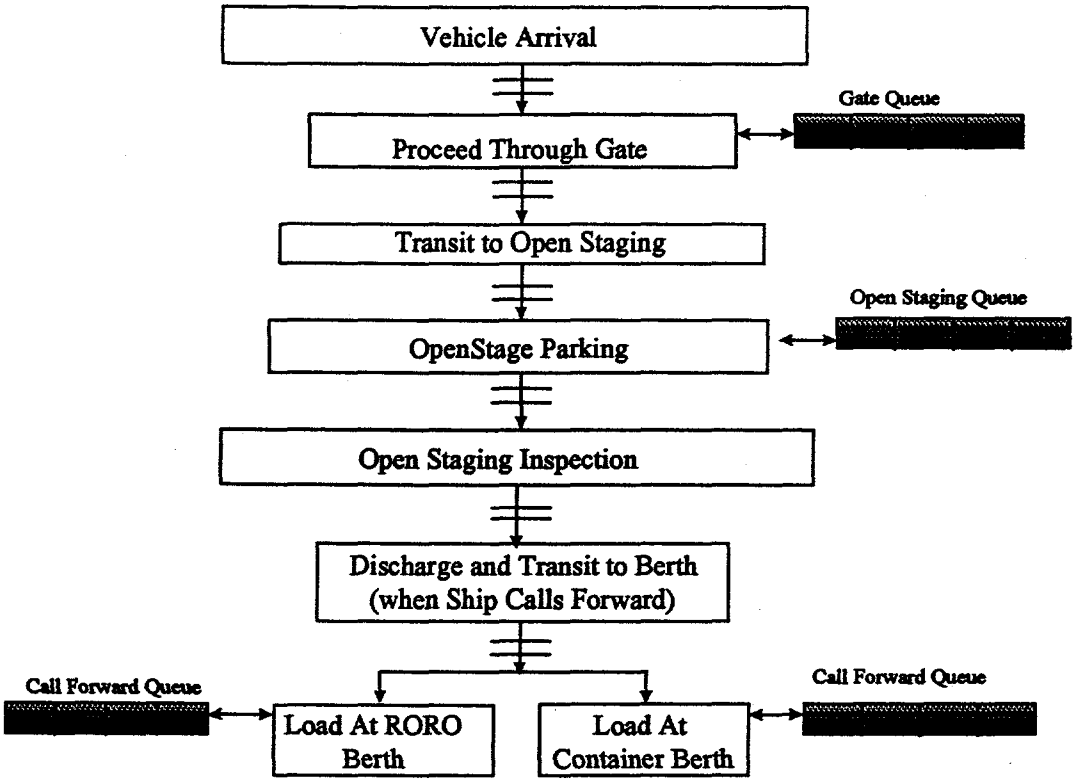

Figure 1. Operational Flow of the Seaport Convoyed Vehicle Embarkation Process Simulated in PORTSIM

PORTSIM is an event-driven simulation model. The entire simulation process follows an event list. The event list is generated dynamically based on an event hierarchy. An event time is assigned to each event in this list when the event is generated. The cargo arrival event time sequence is generated using a user-selected stochastic distribution over a given time interval. The currently available selections are exponential, beta, gamma, normal, triangular, and uniform distributions. An internal simulation clock is used to control the time advancement of the simulation. The simulation clock starts when the simulation commences. A simulation event is triggered when the simulation clock time reaches the event schedule time. The entire simulation process is carried out by internal control. No dynamic outside input is currently accepted or required.

PORTSIM is an object-oriented simulation system. All cargo items, ships, and port infrastructure resources are modeled as individual objects. The cargo items processed in a typical scenario can number over 10,000 pieces of equipment. This allows PORTSIM to perform a comprehensive analysis on all critical military seaport operations with reliable results.

2.2 PORTSIM federate and non-federate object

An HLA federate is a simulation model implemented in HLA compliant format. It consists of four essential components: federate object classes, class attributes, interaction classes, and interaction parameters. The federate objects are instances of object classes defined in the Federation Execution Data (FED) file. These objects are used by a federate to present itself to other federates in a federation.

The attributes are aspects that define an object class. The interaction classes and interaction parameters of a federate are used to define the cause and reaction relationship between federates. To make PORTSIM HLA compliant is to develop a PORTSIM federate that can facilitate communication between PORTSIM and other simulation models through the RTI.

The essential functions of a federate are to: (1) perform model simulation, (2) publish to the RTI, and in turn to other federate(s) in a federation, the data it wishes to provide to other relevant model(s) in form of object and attributes, (3) get data from other federate(s) by subscribing to object(s) and attribute(s) other federate(s) publish, (4) initiate and receive interaction(s) if there are any, and (4) coordinate with other federates, if necessary, through time management services that the RTI provides.

For the PORTSIM federate, the objects to be published are the port object and the cargo item objects. The port object represents the port that PORTSIM simulates. The attributes of the port object are the parameters, such as port resource utilization rates and port queues, that define the port status. Because there are many different types of cargo items that need to be loaded and unloaded, defining object classes for each type would make the FED file very complex. To simplify the FED file and our discussion, we indiscriminately use the cargo object to represent all cargo items, while using cargo object attributes to differentiate one cargo type from another. With this highly abstracted class, the number of object classes that need to be defined in the FED file is reduced to two. The term cargo object is hereafter used to refer to all types of cargo items in our discussion. Because the current version of PORTSIM does not accept dynamic input, no interaction class needs to be defined in the PORTSIM federate.

Of the two objects defined for the PORTSIM federate, the port object is a regular federate object that is owned and published by the federate. The object can be managed using object management services that are provided by the RTI. The cargo object, however, has to be handled differently. In the embarkation process, cargo items come to the port with various transportation assets. The port (1) unloads the cargo items, (2) changes cargo's dimensions, such as length, height, weight, and/or contents, if necessary, and (3) loads the cargo items onto a ship(s). In the debarkation process, the port (1) unloads the cargo items from a ship, (2) changes cargo's dimensions if necessary, and (3) then loads the cargo items onto a new carrier(s). Finally, the cargo items leave the port for new destinations. In either case, the ownership of the cargo items change during these processes. The attributes of a cargo object can also be changed by any of its owners. To simulate this process in military equipment deployment and a military mobility analysis simulation federation(s), it is clear that the ownership of the cargo object must also be able to be changed between the participating federates. The receiving federate, upon the reception of the object, must create a local copy of the cargo object it receives and resume responsibilities of publishing and updating the object and its attributes. The old owner must delete the local copy of the cargo object and the copy it published to the RTI when the object is transferred to a new owner.

From the above analysis, one may find that the ownership of the cargo object transfers from one federate to another. This factor signifies that the cargo object in the port federate is not a regular federate object because the ownership of a regular object never changes in a federation. To distinguish this object from the true cargo object in PORTSIM model as well as regular federate objects in the RTI, the term *non-federate object* is used here to refer to objects, such as the cargo object, that has to be published but can not be permanently owned by a unique federate.

2.3 Non-federate object ownership transfer and time management problems in developing the PORTSIM federate

To develop a comprehensive PORTSIM federate, in practice, is not trivial. A complete analysis of the interactions between potential federates must be made so that the objects and their attributes to be published and subscribed can be determined. Because the development process of the PORTSIM federate involves many complicated issues, a detailed discussion on the design of the PORTSIM federate is out of the scope of this paper. This paper will address only the non-federate object ownership transfer issue and the time management problem encountered in developing the PORTSIM federate.

As discussed earlier in section 2.2 of this paper, the cargo object is a non-federate object with attributes such as the cargo length, width, height, and weight.

A cargo item comes to a seaport for loading or unloading. The ownership of the cargo object instead of the attributes in the federation needs to be changed during this process. In the HLA Run-Time Infrastructure (RTI), however, only the attribute ownership transfer service is provided. The object ownership management issue is not directly addressed. There is no service provided to the object ownership management. A solution must be found.

The other important issue to be resolved in developing a PORTSIM federate is the time management issue. Because the current version of PORTSIM is a stand alone model, it does not accept dynamic cargo input or allow for external time coordination. Only an internal clock is used to track the time of the port operation. A port status and resource utilization report is generated at each hour of this internal clock. To make PORTSIM able to coordinate with other potential federates without major modification, a method must be developed so that the PORTSIM federate can update objects and attributes at the right time with the right data.

3. Methodologies

Solutions have been found for both the object ownership management problem and the time management problem encountered in developing the PORTSIM federate. The methodology developed for the object ownership management issue is to convert the ownership transfer problem into an interaction problem. The time management problem is solved by mapping federate time to federation time.

3.1 Non-federate object ownership management for the cargo object

The non-federate object ownership management problem encountered in developing the PORTSIM federate can be solved by converting the direct ownership transfer problem into a federate object interaction problem. To achieve this goal, first we create an extra interaction class, send_cargo_object, that is auxiliary to the PORTSIM federate. The initiator of the interaction is the federate who intends to give up the ownership of a cargo object. The receiver is the federate which has to take over the ownership of the incoming cargo object. The parameters of the send_cargo_object interaction class are the cargo name, cargo type, the cargo dimension attribute values, and the RTI object ID of the receiving federate. When a cargo object leaves a federate, the federate initiates a send_cargo_object interaction to the receiving federate. The receiver then creates a local copy of the cargo object with the name and attribute values provided through the interaction parameters. The receiving federate then publishes and updates the object and its attributes to the RTI. At this time, there are two duplicate copies of the cargo object. The federate which has given up the ownership of the cargo object must now delete the local object it holds and the copy it published to the RTI. The ownership of the cargo object is thus transferred from one federate to another. The non-federate object, the cargo object, ownership management is hence effectively converted to an interaction problem.

In fact, this approach depicts exactly the actual cargo loading and unloading processes in a port. A cargo item comes to the port with its initial carrier, the owner federate of the cargo object. The port unloads it from the carrier. The port now owns the cargo. After some operations, the cargo item is then loaded, probably with new attribute values, to a new carrier and moved to a new destination, i.e., a new federate. There is no such object in the port anymore. The port federate releases the ownership of the cargo object. The loading/unloading operation is complete. Although the cargo has to be loaded by some port assets (crane, material handling equipment), collectively, the port can still be considered as the one that obtains and releases the ownership of a cargo object and so the owner can be the PORTSIM federate. The steps and RTI services used in implementing this method are summarized as the follows.

1. Create interaction parameter pair;

2. Assign the cargo object ID and attribute values to the interaction parameters;

3. Send the send_cargo_object interaction to the receiving federate using the RTI service:

   RTIAmbassador::sendInteraction(...);

4. RTI calls the federate ambassador method:

   FederateAmbassador::receiveInteraction(...)

   to

   - Create a local copy of the cargo object with given attributes;

- Publish the cargo object using **RTIambassador::PublishObjectClass(...);**

5. Call **RTIambassador::deleteObject(...);**

6. Call **Federate::deleteObject(...);**

7. Call **FederateAmbassador::removeObj(...).**

It is important to point out that the FED_RELIABLE message transportation mechanism must be used in using this technique. This is to guarantee that the receiving federate will get this interaction and then create and publish the newly received cargo object. Otherwise, the interaction message may be lost and so would the cargo object.

A concern that may be raised is the timing of the removal of the cargo object from the RTI. Because the sending and receiving federates may be running on different machines across a computer network, there can be some time delay in getting the interaction message from the initiator to the receiver. Thus, the sender federate may have removed the cargo object from the RTI before the receiving federate receives the interaction. The RTI may temporarily lose tracking of the cargo object. This problem is in fact not as important as it appears to be. This is because the attribute values of a cargo object should not be available to other federates during the simulation of loading and unloading processes. For non-real-time applications of PORTSIM, it is therefore not critical whether the RTI temporarily loses tracking of a cargo object or not during an ownership transition period. This can, however, be a problem for a real-time applications, in which every movement of an object has to be traced. Solutions to this problem will be considered in our future work as this need develops.

3.2 Time coordinated update of the port and cargo objects

The time management problem encountered in the development of the PORTSIM federate can be alleviated by adding an auxiliary port object attribute buffer array to the federate while leaving the simulation logical flow intact. The PORTSIM federate uses this auxiliary array to store the port status report data while running on its internal simulation clock. When the PORTSIM federate joins a federation, the simulation starts and so does the internal simulation clock. The port attribute data, i.e. the simulation output, are generated at specific points on the time axis of the simulation clock. A time stamp based on the simulation clock is attached to each attribute data. Assuming that the PORTSIM federate joined the federation at time T_f of the federation time and a series of port attribute data are generated at simulation times $t_0, t_1, t_2, \ldots, t_n$, the time stamps represented in terms of the federation time can be calculated as:

$$\begin{aligned} T_0 &= T_f + t_0; \\ T_1 &= T_f + t_1; \\ T_2 &= T_f + t_2; \\ &\vdots \\ T_n &= T_f + t_n; \end{aligned}$$

Where T_0, T_1, T_2, ... and T_n are the time stamps attached to the port attribute update data in term of federation time. Thus, when the PORTSIM federate receives an attribute update request for a specific federation time, it can convert the federation time to the simulation time, retrieve the requested data from the auxiliary array, and respond to the request. In this way, a quasi-time coordination scheme is obtained. Figure 2. illustrates the relationship between the federate clock and the federation time axis.

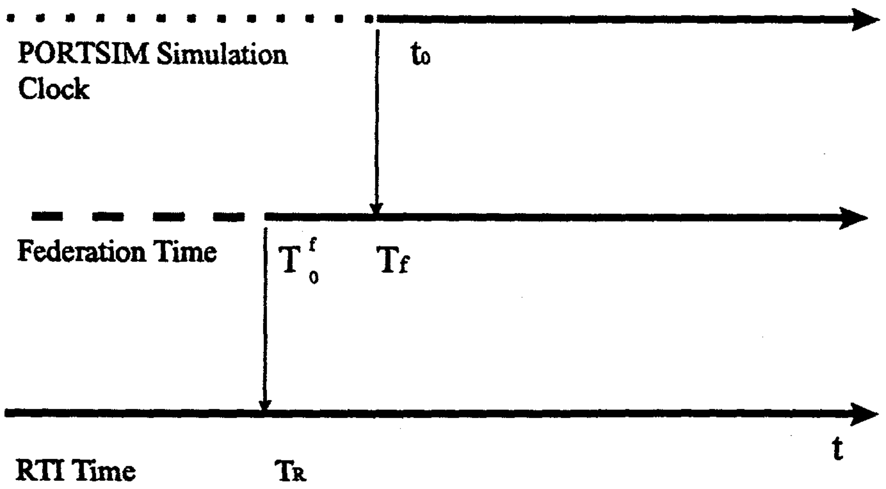

Figure 2. The Relationship Between RTI Time, Federation Time, and PORTSIM Simulation Clock

In Figure 2, t, T_R, and T_0^f denote the RTI time axis, the RTI time at which the federation starts, and the zero point of the federation time in the federation time axis, respectively.

It must be pointed out that this method is based on a fundamental assumption that the PORTSIM federate is capable of finishing analysis for a time step and putting the results into the buffer port history file before a request of port attribute update for this time interval is made by other federate(s). Therefore, the minimal lookahead value, which is the minimum time for PORTSIM to finish the analysis for the requested update, has to be determined and considered in developing a federation.

4. Conclusions

Two important problems were encountered in developing the HLA federate for the seaport military equipment deployment operation simulation model PORTSIM. The first is an object ownership transfer problem for the non-federate cargo object. The second is a time management issue for the federate models, such as PORTSIM, that are not coordinated through an external or federation clock.

Solutions have been found for both of these problems. First, a method is developed for solving the cargo object ownership transfer problem. In this method, the object ownership problem is converted into an interaction problem. When a cargo leaves a port, the port federate holding the cargo object initiates a send_cargo_object interaction. The attribute values are passed as interaction parameters to the receiving federate. The receiving federate instantiates and publishes a new local copy of the object with the attribute values received when it receives the interaction. The original owner of the cargo object then deletes both the local copy and the copy it published to the RTI of the cargo object. The cargo object ownership transfer problem is effectively converted into an interaction problem.

The time management problem has also been solved by mapping the PORTSIM internal simulation clock to the federation clock. In this method, the port attribute data, generated during the simulation, are stamped with the internal clock and pushed to an auxiliary array. A time mapping scheme is then used to convert the simulation time stamp into the federation time stamp. The PORTSIM simulation can then update its attributes to the RTI with the correct data. An important assumption made in this method is that the PORTSIM federate is capable of its finishing analysis for a time step and putting the results into the buffer port history file before a request for updating port attributes for this time interval is made by other federate(s). There is, however, no mechanism in this method to prevent this from happening. The designer of the federate must pay special attention to this matter.

5. Acknowledgment

This work is supported by the Military Traffic Management Command Transportation Agency (MTMCTEA) of the Department of Defense through U.S. Department of Engery contract. Mr. Melvin J. Sutton, Jr. from MTMCTEA has provided invaluable guidance and help in the course of this work. The authors would like to express their sincere appreciation to MTMCTEA and to Mr. Sutton.

6. References

[1] M. J. Sutton, "Force Projection Modeling", Proceedings of the 1997 Fall Simulation Interoperability Workshop, Orlando, FL, September, 1997

[2] M R. Nevins, C. M. Macal, J. Joines, "A Discrete Event Simulation Model for Seaport Operation", Simulation Journal, Society for Computer Simulation (Pending publication).

[3] Defense Modeling and Simulation Office(DMSO), "High Level Architecture Interface Specification Version 1.1", February 12, 1997.

[4] Defense Modeling and Simulation Office(DMSO), "High Level Architecture Run-Time Infrastructure Programmer's Guide Version 1.1", May 15, 1997.

[5] Oak Ridge National Laboratory, "Joint Flow and Analysis System for Transportation (JFAST) User's Guide Version 5.0", Oak Ridge, TN, 1993.

[6] Military Traffic Management Command Transportation Engineering Agency (MTMCTEA), Ports for National Defense, MTMCTEA report SE 91-3d-31.

Author Biographies

ZHIAN LI is a research software engineer at Argonne National Laboratory. He has been working on a broad spectrum of research areas. His main research interests are computer simulation and modeling, operation research, and Geographic Information System. He received his Ph.D. from The Pennsylvania State University, M.S. from Institute of Atomic Energy, and B.S. from TsingHua University in nuclear engineering. He is currently working on the development of HLA federates for the logistics models developed at Argonne National Laboratory.

CHARLES M. MACAL directs the Artificial Intelligence Applications Section and leads the Logistics Modeling and Simulation Program at Argonne National Laboratory. His research interests include simulation modeling and architectures, and agent-based modeling, He has a Ph.D. in operations research from Northwestern and degrees from Purdue. He is a registered professional engineer in the state of Illinois and a member of INFORMS, the American Association for Artificial Intelligence, the Society for Computer Simulation, and Tau Beta Pi.

MICHAEL R. NEVINS is a software engineer in the Decision and Information Sciences Division of Argonne National Laboratory. He received a M.S. in Computer Science from Depaul University and a B.S. degree in Computer Science from Elmhurst College. He is the developer of the PORTSIM model. His research interests include simulation modeling and object-oriented software design and development. He is a member of Phi Kappa Phi, Pi Mu Epsilon, and the Society for Computer Simulation.