METHODOLOGY

DEVS/HLA-Based Modeling and Simulation for Intelligent Transportation Systems

Jong-Keun Lee
Intelligent System Research Laboratory
Department of Computer Engineering
Hangkong University
Koyang-shi, Hwajeon-dong 200-1, Seoul 411-791, Korea
leejk@mail.hangkong.ac.kr

Min-woo Lee
Sena Technologies, Inc.
210 Yangjae-dong Seocho-gu Seoul 137-130, Korea

Sung-Do Chi
Intelligent System Research Laboratory
Department of Computer Engineering
Hangkong University
Koyang-shi, Hwajeon-dong 200-1, Seoul 411-791, Korea

This article focuses on the study of the modeling and simulation of an intelligent transportation system that uses discrete event system specification (DEVS)/high-level architecture (HLA), which is the methodology of hierarchical modular modeling and distributed simulation. The authors performed distributed homogeneous traffic simulation by extending an existing developed DEVS-based I^3D^2 transportation simulation system to an HLA-based distributed simulation environment. First, the suggested methodology gives an object-oriented and hierarchical modular modeling and simulation environment of traffic models that have complicated, dynamic features of the real world. Second, it provides easiness and extendibility of a distributed traffic simulation system design/development. Third, it offers the whole simulation time management method between distributed simulation objects through the HLA distribution standardization technique. Finally, it can offer an effective and hierarchical simulation analysis environment of a large-scale road network.

Keywords: DEVS/HLA, distributed traffic simulation, I^3D^2 traffic simulation system, intelligent transportation system, system entity structure/model base

1. Introduction

The growth of urban automobile traffic has led to serious and worsening traffic congestion problems in most cities around the world. Intelligent transportation systems (ITS) appear to provide a set of intuitively promising tools to improve the increasingly complex and rapidly deteriorating transportation systems of today. *Intelligent transportation systems* is an umbrella term. It covers the application of a wide variety of computer, communication, positioning, sensing, control, and other information-related technologies. Major categories of ITS include traffic management systems and traveler information systems [1, 2]. In recent years, simulation has become a tool commonly used to understand the characteristics of a traffic system and to select an appropriate design for ITS [1]. Thus, modeling and simulation techniques provide a convenient way to (1) evaluate the alternative signal control strategies at the operational level of advanced traffic management systems (ATMS) and (2) generate simulation-based forecasting information for advanced traveler information systems (ATIS) [2, 3]. Potential success of such ITS is implicit and derived, to a large extent, from the success of the underlying technology in other fields such as the defense and aerospace industries. Comprehensive research tools for quantifying the expected benefits from ITS are, however, still absent. At the moment, the use of traffic simulation is regarded as

SIMULATION, Vol. 79, Issue 8, August 2003 423-439

DOI: 10.1177/0037549703040233

inevitable for anyone who needs to quantify potential benefits prior to any major investment in development and deployment. Simulators are needed not only to assess the benefits of ITS in a planning mode but also to generate scenarios, optimize control, and predict network behavior at the operational level [2]. In addition, recent ITS adds advanced electronic, communication, and computer techniques to existing transportation systems and tries to intellectualize the road environment, generalize the offer of transportation information, make vehicles and traffic management advanced, and so on, so the complexity of existing traffic components is growing. To receive these changes, existing simulation systems are trying to draw a parallel and distributed method.

Research of distributed simulation is rapidly progressing in various sectors as a measure to overcome limitations derived from nonbalanced system performance and complex applications. Lately, as the need increases for a tool that can perform a synthetic analysis and evaluation of the ATIS and ATMS, distributed and parallel simulation techniques that improve system performance, share system resources, enhance extendibility, and reduce costs are catching people's attention. For example, microscopic traffic simulation systems such as TRANSIMS (Los Alamos National Lab) [3, 4], THOREAU (The MITRE Corp.) [5], SIMLAB (MIT) [1], DYNEMO (PTV AG) [6–8], and PARAMICS (Edinburgh Parallel Computing Center & Quadstone Corp.) [9, 10] are trying to improve their system performance by using a parallel virtual machine (PVM) (see www.epm.ornl.gov/pvm) that offers a virtual parallel computing environment. However, this PVM technology has a shortcoming—it does not have elements needed for the whole simulation time management and its data management in a distributed simulation environment. Instead, it has an advantage in that a large computational problem can be solved more cost-effectively by using the aggregate power and memory of many computers in a homogeneous/heterogeneous simulation environment because it is a software package for exchanging distributed messages between systems built in a different system-developing environment. In addition, SmartPATH [11–13]—a simulation environment for the automatic highway system (AHS) of the PATH research center at the University of California, Berkeley—offers a distributed simulation environment by obtaining a distributed supervisor (DS) to manage messages among processors. However, although this system developed and applied independent distributed simulation techniques, thereby overcoming problems derived from overhead that the simulation of larger road networks generates, there are still difficulties in supporting the interoperability and reusability of various traffic simulation systems resulting from independent development and research of various traffic components. Helsinki University of Technology in Finland also implemented a distributed traffic monitoring and information system in HUTSIM [14] by using a distribution method such as the distributed shared memory system (DIME) [15], which Nottingham Trent University in Britain developed. Otto-von-Guericke University in Germany showed that its system could be available for interoperability among simulators operated in heterogeneous systems by applying the high-level architecture (HLA) interface to commercial simulation tools such as SLX and Simplex 3 [16, 17]. However, a study has not been attempted of direct distributed processing for internal traffic models to express effectively their dynamic features among various and complicated traffic models in a distributed environment. In the meantime, we have developed an I^3D^2 transportation simulation system [18, 19] and attracted considerable attention in that it reproduces real-world traffic phenomena according to discrete event formalism by using an advanced hierarchical modular modeling and simulation method. Although this system has the possibility of distributed processing among traffic models, it still has limitations in practical use.

Therefore, we addressed the complexity and extendibility of ITS by applying a discrete event system specification (DEVS)/high-level architecture (HLA)(DEVS/HLA) [20, 21]–based distributed modeling and simulation environment to the I^3D^2 transportation simulation system that we have developed and implemented it. The suggested DEVS/HLA-based distributed traffic simulation environment has the following advantages compared to the existing distributed traffic simulation environment by integrating DEVS formalism and the HLA-distributed simulation standardization technique:

1. It provides an object-oriented and hierarchical modular modeling and simulation environment of traffic models that have complicated and dynamic features of the real world.

2. It provides easiness in the design and development of a distributed traffic simulation system.

3. It provides the whole simulation time management method between distributed simulation objects through the HLA-distributed standardization technique.

4. It provides an effective and hierarchical simulation analysis environment of a large-scale road network.

In this study, we performed distributed homogeneous simulation by dividing 16 crosses, 78 two-way roads and their vehicles, and other components—which are included in the sample road network of an existing I^3D^2 transportation simulation system—into four federates on the basis of DEVS/HLA. Therefore, we proved that the suggested DEVS/HLA-distributed modeling and simulation environment gives an effective distributed traffic simulation environment of a large-scale road network made by the same simulation language/environment.

The remainder of this article is organized as follows. Section 2 provides the background on distributed and parallel traffic simulation systems. Section 3 gives

DEVS/HLA-based distributed traffic modeling and simulation environments, and section 4 provides a distributed traffic simulation example. Finally, section 5 provides conclusions and future work.

2. Background on Distributed and Parallel Traffic Simulation Systems

Recently, an approach method of traffic simulation systems for a distributed and parallel simulation environment that deals with the complexity and extendibility of the ITS has been largely classified in two ways. One way is to classify according to distributed and parallel computing techniques, and the other is to classify according to distributed targets. Table 1 shows the classification of traffic simulation systems based on distributed and parallel simulation techniques and distributed targets.

2.1 Classification by Computing Technology

As distributed and parallel techniques are developed, simulation techniques for the analysis and evaluation of traffic simulation systems have become various. First, there are ways to use a high-power supercomputer that has a distributed shared memory and multiprocessor and to enhance the speed of systems by using a parallel computing technique in the workstation system environment. Second, there is a method that supports not only the speed of systems but also the interoperability and reusability for various individual traffic simulation systems by using the distributed middleware technique. Therefore, flexible system extendibility and low simulation costs can be offered.

The PVM [13] is representative of one of the recent parallel computing techniques and has been applied to traffic simulation systems. PVM is the software technique that effectively converts the same or different kinds of computers that use UNIX connected to a computer network to one virtual parallel computer by the message-transmitting method. PVM1.0 was developed by Oak Ridge National Laboratory in Los Alamos in 1989 and was used for parallel calculating. PVM3.4 (3.6 beta), currently in the spotlight as a good tool for paralleled calculations, especially plays a crucial role in parallel computing through Linux-PC clustering. However, PVM technologies are unsatisfactory in the time and data management of the distributed simulation, such as the whole time-progressing method by conservative synchronization that HLA provides.

Various distributed computing techniques have been developed for exchanging data effectively in a general network environment. Middlewares that support this distributed environment offer the technique of exchanging data on a different platform and interoperability in a client-server environment. They also give effective, flexible, and extensible information sharing by integrating different platforms. General middleware techniques such as these are the HLA standardization technique that the U.S. Department of Defense (DoD) recently suggested, the common-object request broker architecture (CORBA) of the Object Management Group (OMG), and the distributed-component object model (DCOM) of Microsoft. Here, the HLA-distributed middleware technique is a measure to enhance interoperability among various simulation models and stands in the spotlight of modeling and simulation in defense fields. Its availability is increasing in various applications [22, 23] (see also http://hla.dmso.mil and http://www.omg.org/gettingstarted/specintro.htm#CORBA).

2.2 Classification by the Distributed Target

Traffic simulation systems were developed independently according to their functions and purposes to perform their duties as a tool for various analyses and evaluations of ITS. However, their role as integrated systems became larger due to the need for a synthetic analysis and evaluation tool of the ATIS and ATMS of ITS. Therefore, the need for the integration of traffic simulation systems increased according to each analysis purpose and function. Also, the need for the distributed simulation of large-scale road networks increased. The method that classifies current traffic simulation systems according to a distributed target includes the following two ways.

One way is to apply the distributed and parallel simulation technique by integrating simulators developed independently according to analysis purposes and functions. It includes SIMLAB, developed by MIT, and the HLA-based distributed traffic simulation system by Otto-von-Guericke University in Germany. For example, SIMLAB [1] distributed, integrated, and implemented three simulation systems such as MITSIM, TMS, and MesoTS according to the PVM technique. In the integrated and distributed simulation environment of SIMLAB, simulators (MITSIM, TMS, MesoTS) implement communications according to the PVM technique by using message-transmitting shared memory and data files. The HLA-based distributed traffic simulation system by Otto-von-Guericke University integrated different simulators (driving simulator, traffic simulator) developed in a different simulation domain by using SLX and Simplex 3—commercial simulation tools—through the HLA interface. The distributed environment of individual simulation components is good for integrating different individual simulation components, but if one of the independent simulation components gets more complex, it is difficult to extend to the flexible system.

The other way is to apply the distributed and parallel simulation technique by dividing whole traffic networks into regional subnetworks for the analysis of large-scale traffic networks. For example, the simulation of large traffic networks with substantial numbers of cars requires considerable runtimes. The traffic flow model DYNEMO [8, 9, 24] has been parallelized to speed up this simulation process. The master process is responsible for the supervision of the slaves, the initialization and decomposition of the traffic network, and tasks that require information about

Table 1. Classification of distributed and parallel traffic simulation systems

Distributed Target Computing Technology	Traffic Network	Simulation Component
Parallel computing based	· TRANSIM (PVM) · DYNEMO (PVM) · THOREAU (PVM) · PRAMICS (PVM) · SmartPATH (DS)	· SIMLAB(PVM) · HUTSIM(DIME)
Distributed “middleware” based	· I^3D^2 (DEVS/HLA)	· Traffic simulation system [16] (HLA)

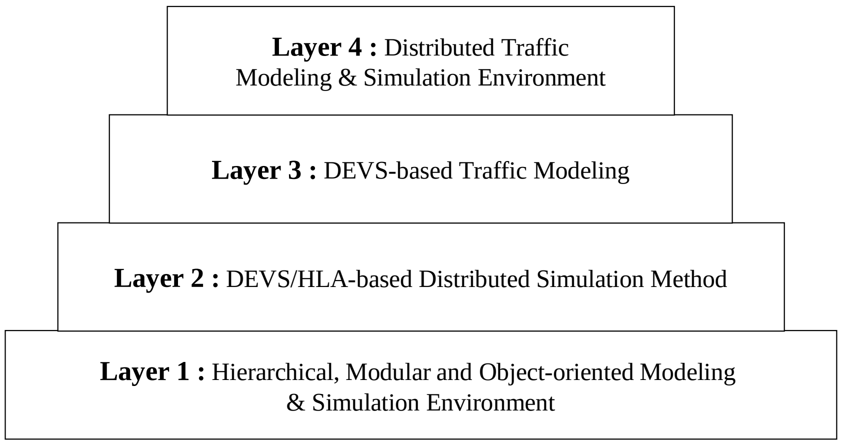


Figure 1. Layered approach for a distributed traffic modeling and simulation environment

the state of the complete network. During one time step, each slave calculates the spatial motion of the vehicles situated on its subnetwork. The decomposition should provide possibly large, coherent regions of the subnetworks with minimal interconnections (stretches) between the subnetworks. The performance enhancement method, by dividing the traffic network, is suitable for planning and managing a macroscopic simulation environment based on large-scale networks.

3. DEVS/HLA-Based Distributed Traffic Modeling and Simulation Environment

In this section, we suggest a layered approach such as the one in Figure 1 to get a distributed transportation modeling and simulation environment for receiving the complexity and extendibility of the ITS. In the first layer, a hierarchical, object-oriented, modular modeling and simulation environment of the system is applied through the system entity structure and model base framework. In the second layer, we receive the complexity and extendibility of systems according to the DEVS/HLA that offers a distributed modeling and simulation environment. In the third layer, a modeling environment for various traffic components of the ITS is built according to the methodology obtained from the low layers. Finally, the fourth layer shows a distributed traffic modeling and simulation environment obtained from these layers.

3.1 Layer 1: Hierarchical, Modular, and Object-Oriented Modeling and Simulation Environment

The system entity structure/model base [24, 25] framework serves as a basic hierarchical, modular, and object-oriented modeling and simulation environment for designing traffic simulation systems. The system entity structure/model base framework was proposed by Zeigler as a step toward marrying the dynamics-based formalism of simulation with the symbolic formalism of artificial intelligence. Basically, it consists of two components: a system entity structure (SES) and a model base (MB).

The *system entity structure*, declarative in character, represents the knowledge of decomposition, component taxonomies, coupling specification, and constraints. The *entities* of SES refer to conceptual components of reality for which models may reside in the MB. An entity may have several *aspects*, each denoting decomposition and therefore having several entities. An entity may also have

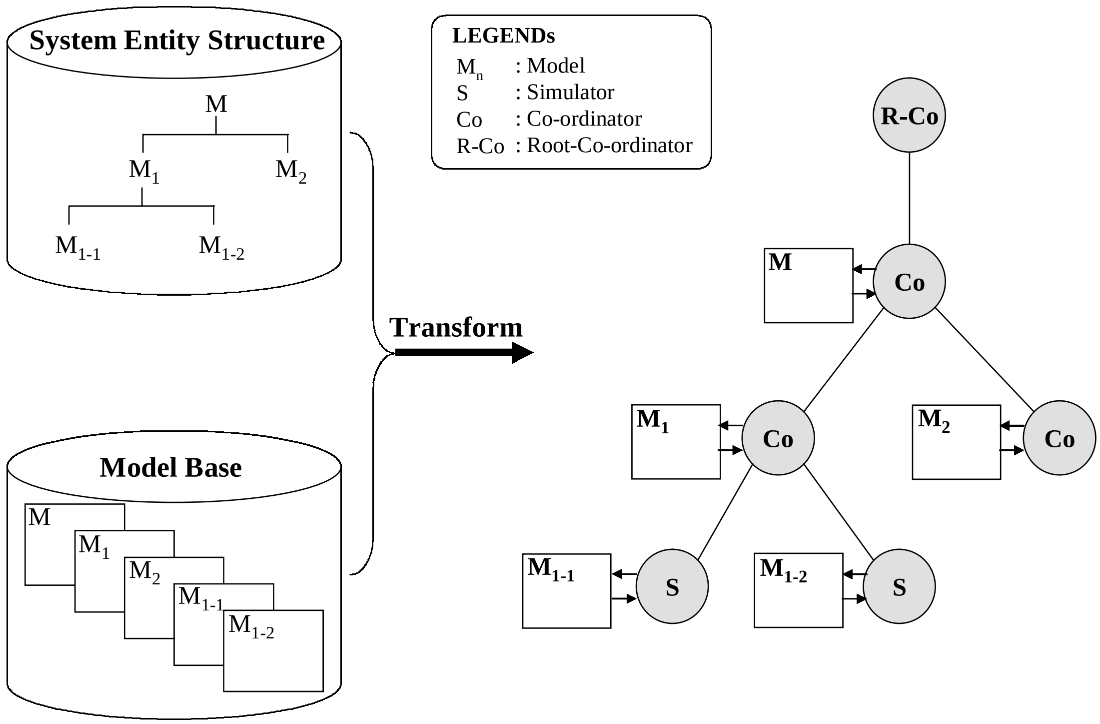


Figure 2. System entity structure/model base environment

several *specializations*, each representing a classification of the possible variants of the entity. The *pruning* operation extracts a substructure of the SES by selecting one aspect and/or one specialization for each entity in the SES. Hierarchical simulation models can be constructed by applying the transform function to pruned entity structures (PESs) in working memory. The *model base* contains models that are procedural in character, expressed in DEVS formalism [24, 25], which is a theoretically well-grounded means of expressing modular discrete event simulation models. As shown in Figure 2, the SES that has a structural feature of the system and the models that have dynamic behaviors in the model base are composed by the transform command. Each model is operated by the DEVS abstraction simulator. Detailed descriptions of SES/MB are available in Zeigler [24] and Zeigler, Praehofer, and Kim [25].

3.2 Layer 2: DEVS/HLA-Distributed Modeling and Simulation Environments

Traffic simulation models traditionally have been developed as monolithic, more or less stand-alone systems well suited for actual needs. However, what happens if these needs change, posing new requirements or new connectivity demands to other systems? *Interoperability* and *reusability* are the catchwords for open systems that provide new flexibility in a modular way. The HLA is a promising, upcoming standard in distributed simulation. Using the HLA as a basis for distributed traffic simulation, new synergetic effects could be used to gain even more flexibility [16].

HLA [22] is a middleware designed specifically for a distributed simulation environment. HLA, which defines major functional elements, interfaces, and design rules, is the technical architecture for DoD simulations. The HLA rules are a set of rules that must be followed to achieve a proper interaction of simulations in federations. The runtime infrastructure (RTI), which works as a logical layer, provides communications between a federation and several federates in a way that is analogous to how a distributed operating system provides services to applications.

DEVS/HLA [20, 21, 23, 26], suggested by the University of Arizona, is an HLA-compliant modeling and simulation environment formed by mapping the C++ version of the DEVS formalism to the C++ version of the HLA/RTI. DEVS is a mathematical formalism for expressing discrete event models that supports discrete event approximation of continuous systems and has object-oriented, substrate-supporting model implementation and repository reuse. Advantages of the DEVS methodology for model

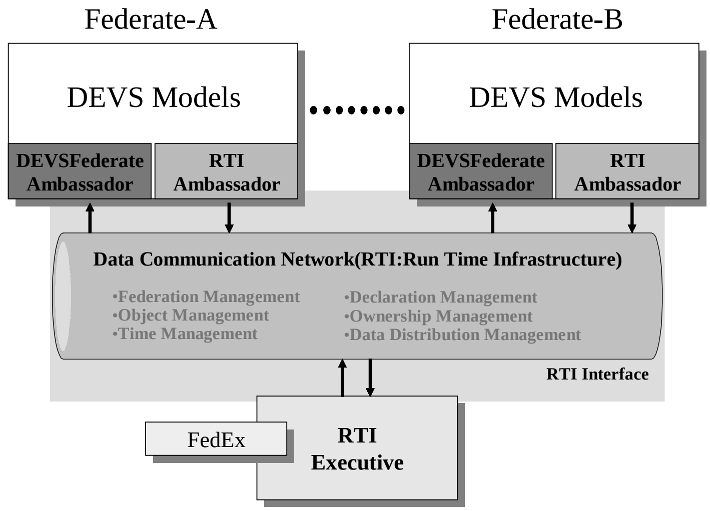


Figure 3. DEVS/HLA-based distributed simulation environments

development include well-defined separation of concerns supporting distinct modeling and simulation layers that can be independently verified and reused in later combinations with minimal reverification. Such formal properties of the DEVS methodology enable DEVS/HLA to support high-level federation development and execution. Modularity, in which component models are coupled together in input-output ports, allows messages to be sent from one federate to another using the underlying HLA interface messages. One of the key objectives in designing the DEVS/HLA simulation environment is portability of models across platforms at a high level of abstraction. Since a DEVS model can be reused based on object-oriented technology, such portability would enable a model to be developed and verified in a platform and be easily ported across distributed platforms. The DEVS formalism is expressed as a collection of objects and their interactions, with the details of the implementation hidden within the objects. The user interacts with only those interfaces that manifest the DEVS constructs while being shielded from the ultimate execution environment provided by the services of the HLA/RTI. Figure 3 shows a DEVS/HLA-based distributed simulation environment. The RTI of HLA performs federation management, object management, time management, declaration management, ownership management, and data distribution management to provide a distributed simulation environment among each federate. DEVS models in each federate transmit messages to the RTI through the DEVSFederateAmbassador. Events and messages to be input to outside federates perform distributed simulation while being input through the RTIAmbassador, the class that the U.S. DoD released. To deal with these capabilities, the DEVS/HLA-C++ [18, 19] code has been developed in Visual-C++ on the basis of the HLA-compliant DEVS formalism, as shown in Figure 4. Upper-case letters indicate the class, and a lower-case letters indicate the attributes that the class has. Each class is explained as follows.

- *ENTITIES:* The uppermost abstraction class of DEVS elements. The *ENTITIES* class has only the entity name and class name. All models have their names through this class.
- *MODELS:* The uppermost class of the DEVS model. This manages pointer of parent model, simulation time and combination information among models. Through MODELS, information about the models used in the actual simulation, as well as combination information among models, can be obtained. Also, this class declares functions that are expected to be defined in the lower class as virtual functions.
- *ATOMIC:* Basic atomic model of DEVS model inherited when the user DEVS model is materialized. This class contains the status and time information that each DEVS model has. That is, this class changes its status with time information and should maintain the status through sigma, time value, and phase that has status information. As mentioned in DEVS formalism, the internal status transition function, external status transition function, time delay function, and output function help these status transitions.
- *COUPLED:* Abstracted class of all DEVS models that has a children model. Unlike ATOMIC, this class simply transmits messages among models to the target model with information about the combined ATOMIC model.
- *DIGRAPH:* Member variable that has a list of the children model. Users should inherit one element of this class to define children as a combined relation.
- *ENTSTR:* A class that manages simulation. Users should have one element of this class for simulation.
- *FEDERATEAMBASSADOR:* A class that defines the function that the RTI calls. This class was released by the U.S. DoD.

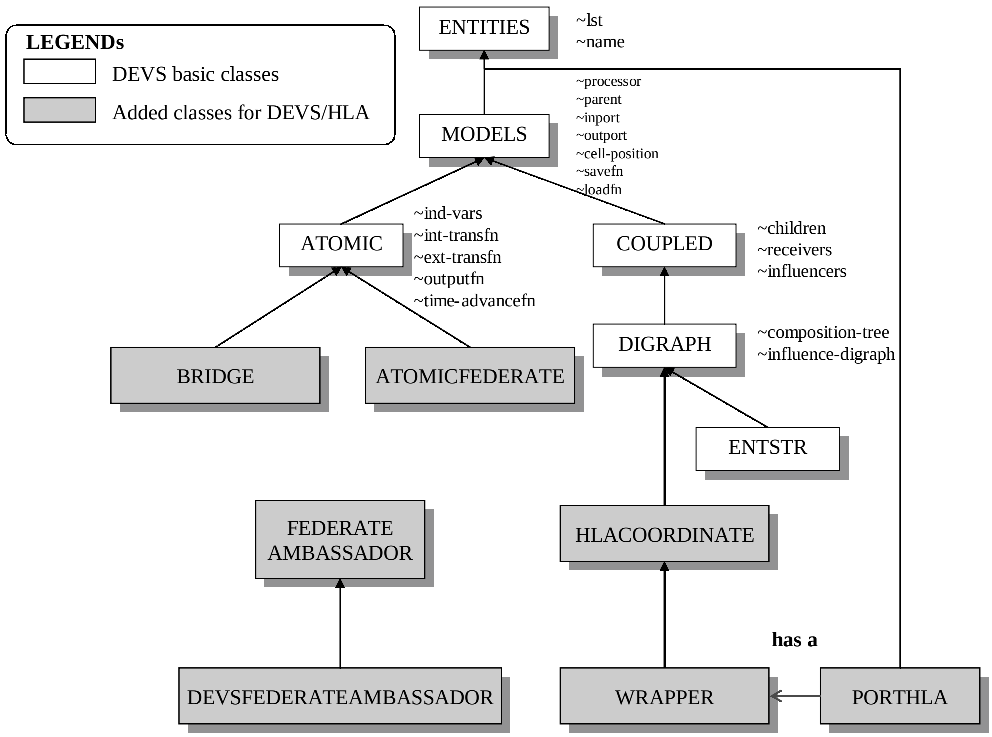


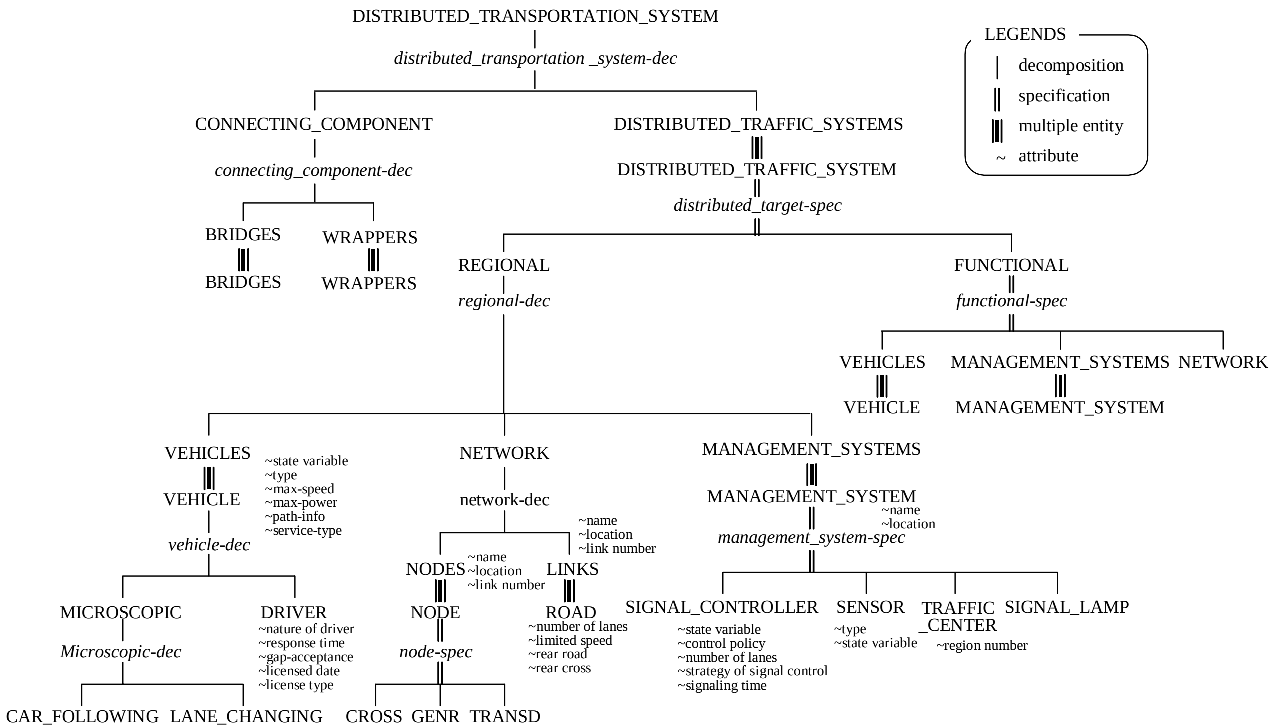
Figure 4. Basic class hierarchy of DEVS/HLA-C++

- *DEVSFEDERATEAMBASSADOR:* This class receives functions for communication with the RTI from the FEDERATEAMBASSADOR. This is defined as a callback function to the event that occurred by the RTI, and it is used to synchronize and set events to the external input message, the time advance grant signal, and the next grant time.
- *PORTHLA:* This class manages the interaction and parameter of HLA. It is equal to the port concept of the existing DEVS model, and it also defines publish-and- subscribe functions of the interaction class.
- *ATOMICFEDERATE:* A class for making the atomic model of DEVS one federate. This class defines member functions for the HLA federate.
- *HLACOORDINATE:* A class for making the combined model of DEVS one federate. Like ATOMICFEDERATE, this class has member functions for the HLA federate. However, the function for transmitting a message from outside to inside the atomic or coupling model is defined.
- *WRAPPER:* A class that has a port to communicate with outside. The WRAPPER model surrounds all DEVS models to make message transmission among federates easy.
- *BRIDGE:* A class that handles messages when models in each federate communicate with the external federate through the WRAPPER model. The BRIDGE model is used to analyze the overhead that has occurred in the distributed environment and the distributed simulation's performance.

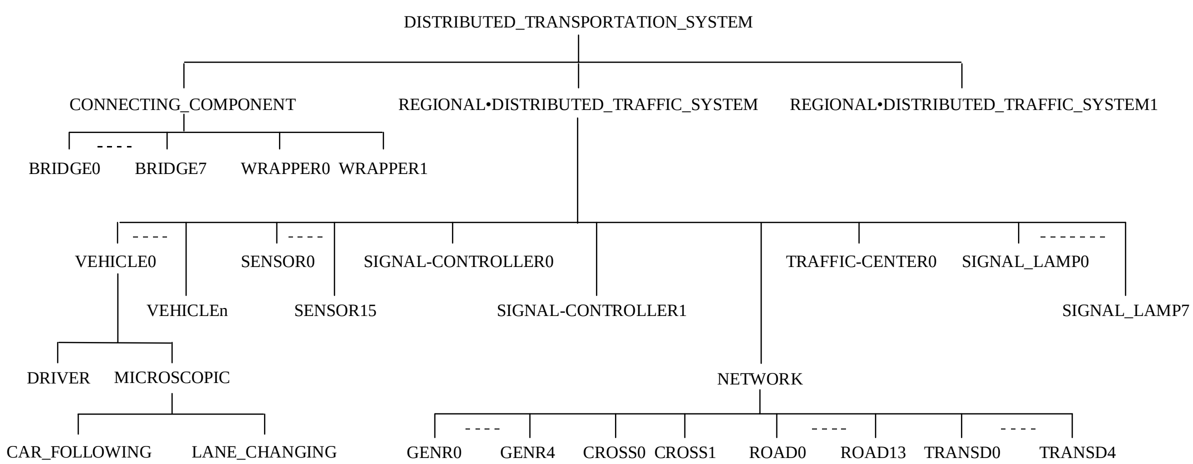
The suggested DEVS/HLA-based distributed traffic modeling simulation environment, as mentioned before, provides easiness and extendibility in the design and development of the distributed traffic simulation system compared to existing PVM-based distributed simulation models by giving the object-oriented and hierarchical modular modeling and simulation environment of traffic models, as well as the whole simulation time management method between distributed simulation objects. Also, HLA/RTI has an advantage in system portability compared to PVM, which supports C, C++, and FORTRAN language, because it supports various object-oriented programming environments such as ADA95, COBAL, IDL, C++, and Java.

3.3 Layer 3: DEVS-Based Traffic Modeling

The DEVS/HLA-based distributed traffic simulation system performs distributed modeling and simulation according to the complexity of traffic components. Figure 5a is an expression of the system entity structure of the distributed traffic simulation system. The DISTRIBUTED_TRANSPORTATION_SYSTEM is decomposed into the DISTRIBUTED_TRAFFIC_COMPONENT and CONNECTING_COMPONENT for effective interoperability in the distributed environment. The DISTRIBUTED_TRAFFIC_COMPONENT is decomposed

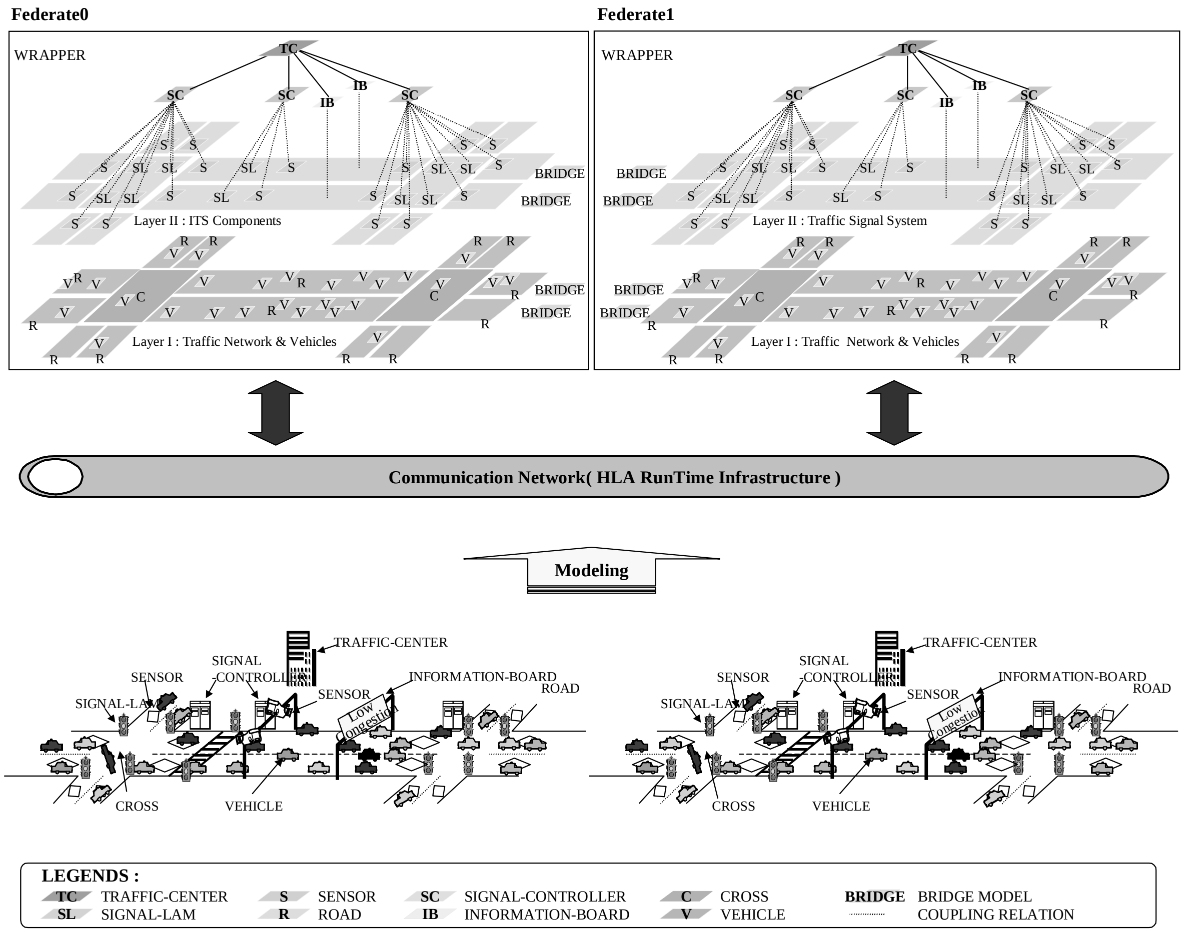


(a) SES representation of distributed transportation system



(b) Example of PES

Figure 5. Distributed traffic modeling example. (a) SES representation of distributed transportation system. (b) Example of PES. (c) Conceptual view of PES and the simulation model structure (continued on next page)



(c) Conceptual view of PES and Simulation model structure

Figure 5. (continued from previous page)

into the REGIONAL entity and FUNTIONAL entity according to the distributed target. The REGIONAL entity is the system that performs distributed simulation, decomposing a large-scale traffic network into a regional subnetwork. The FUNCTIONAL entity explains the structure that can be made when integrated management is needed through the distributed simulation of simulation systems that have independent functions such as the traffic signal system and the simulation system for vehicles. The REGIONAL entity is decomposed into three elements, including the MANAGEMENT_SYSTEMS and the NETWORK, which are the actual traffic components, and the VEHICLES that occupy above NETWORK. Then, MANAGEMENT_SYSTEMS is decomposed into the SIGNAL_CONTROLLER, SENSOR, TRAFFIC_CENTER, and SIGNAL_LAMP, and NETWORK consists of many NODES and LINKS. As shown in Figure 5a, they can consist of various types. VEHICLE consists of MICROSCOPIC simulation, which reproduces behaviors of individual vehicles, and the DRIVER. In the figure, attributes (∼) belonging to each entity represent the common characteristic variable that each entity has. For example, a characteristic variable such as nature of the driver, among the attributes of DRIVER, affects choosing CAR_FOLLOWING, which performs the main operation in the microscopic simulation. By applying the pruning process to an SES built like this, the system structure for distributed simulation is completed. Figure 5b, which is an example of the PES of the SES in Figure 5a, decomposes one large-scale traffic network into two federates in the DEVS/HLA-distributed simulation environment and performs distributed simulation. The conceptual map and model structure of PES are similar to the ones shown in Figure 5c.

3.4 Layer 4: Distributed Traffic Modeling and Simulation Environment

The distributed traffic modeling and simulation environment that uses DEVS/HLA is obtained from the hierarchical approach of the three phases previously suggested. A distributed traffic modeling and simulation environment obtained such as this one can be available for the synthetic analysis and evaluation of each function of the ITS. Figure 6 explains a traffic simulation system development methodology that uses a DEVS/HLA-distributed traffic modeling and simulation environment with five phases, ranging from phase I (constraint and requirement specification) to phase V (application). Each phase is simply explained as follows.

Phase I. Constraints and requirements of each city/road that are needed for the analysis of large-scale networks and the traffic plan are input, and initial conditions of the distributed simulation target and simulation environment are given. Structural/dynamic models of traffic models to be created in the second phase are created. Data (size of traffic network, vehicle's generation distribution, number of subnetworks, simulation period, and attributes of various traffic components) needed for creating initial conditions for distributed simulation in the third phase are collected, and these data are input.

Phase II. In this phase, structural/dynamic models for distributed modeling and simulation of large-scale networks are built. The DEVS-based traffic model, explained in layer 3 of section 3.3, is created. The system entity structure of the distributed transportation system, as shown in Figure 5a, is created. The pruned entity structure is created based on data (size of traffic network and number of subnetworks, vehicles, sensors, signal controllers, bridges, and wrappers) obtained in phase I. Each traffic model is materialized according to the DEVS formalism introduced in layer 1 of section 3.1, and these models are mounted on a model base. Various information about traffic components needed for simulation runtime is constructed as a traffic database.

Phase III. The DEVS/HLA-based distributed simulation environment is built by (1) dividing the DEVS-based large traffic network (built in the second phase) and the traffic components into a subnetwork in each federate (see Fig. 3) and (2) mounting this on DEVS/HLA-distributed modeling and simulation environments introduced in layer 2 of section 3.2. The subnetwork divided into each federate is joined with the RTI of HLA at the same time when simulation starts. Also, it is combined into one federation that the RTI manages, and distributed simulation of one large traffic network is performed.

Phases IV and V. In phase IV, useful data for the traffic plan and the management of the large-scale network are created based on the result of the distributed simulation obtained in phase III. In phase V, research and development of various types of application systems that can perform a synthetic analysis of each function of the ITS are applied according to the simulation environment obtained in the previous phase. This article is a subsequent paper of "Hierarchical Modeling and Simulation Environment for Intelligent Transportation Systems" [19]. In Lee, Lim, and Chi [19], the method of traffic modeling and simulation, as well as the traffic network, is explained in detail. Their study also showed that "ATIS : The Next Generation" and the I^3D^2 transportation simulation system were developed as an application system of the ITS. Therefore, more details are in Lee, Lim, and Chi [19].

4. Distributed Traffic Simulation Example

In this section, we expand and apply the I^3D^2 transportation simulation system [18, 19], developed by the intelligent system lab at Hankuk Hangkong University, to the previously proposed DEVS/HLA-distributed traffic modeling and simulation environment. By this, we verified that the environment accepts the complexity and extendibility of ITS through a case study. The left-hand side of Figure 7 explains the approach method, which performs distributed simulation after dividing one large-scale traffic network into four subnetworks (dotted lines) and mounting them on four systems, such as the one shown on the right-hand side of the figure.

4.1 Model Description and Simulation Conditions

To test the distributed traffic simulation, we divided 16 crosses, 78 two-way roads, vehicles, and other components that the sample road network of Figure 4 includes into four subnetworks (federate). Each federate consists of 4 crosses (CROSS), 24 road models (ROAD), many vehicles (VEHICLE), and the component models of the ITS, as in the simulation model structure presented in Figure 8. The subnetworks are mounted on four computers, and distributed simulation is performed. Simulation models of Figure 8 are explained as follows. (The ROAD model, CROSS model, generator model, transducer model, car-following algorithm, and lane-changing algorithm are omitted because they are introduced in Lee, Lim, and Chi [19].)

- *Signal controller model (SC):* This model controls the signal light at the intersection, and there is one signal controller model at the signal intersection that has a signal light. This model collects information about the congestion situation from the sensor model and establishes signal policy. Also, it offers a signal to each *signal lamp* model according to the phase sequence, cycle, green signal time, and signal control variables. Figure 9 represents a simple pseudo-code of the signal controller model.
- *Sensor model (S):* This model is in charge of collecting traffic information, which is an important part of ITS research. It is possible to classify this model into various kinds of models. The kind of sensor, the location of the sensor, and the sensing rate play an important role in

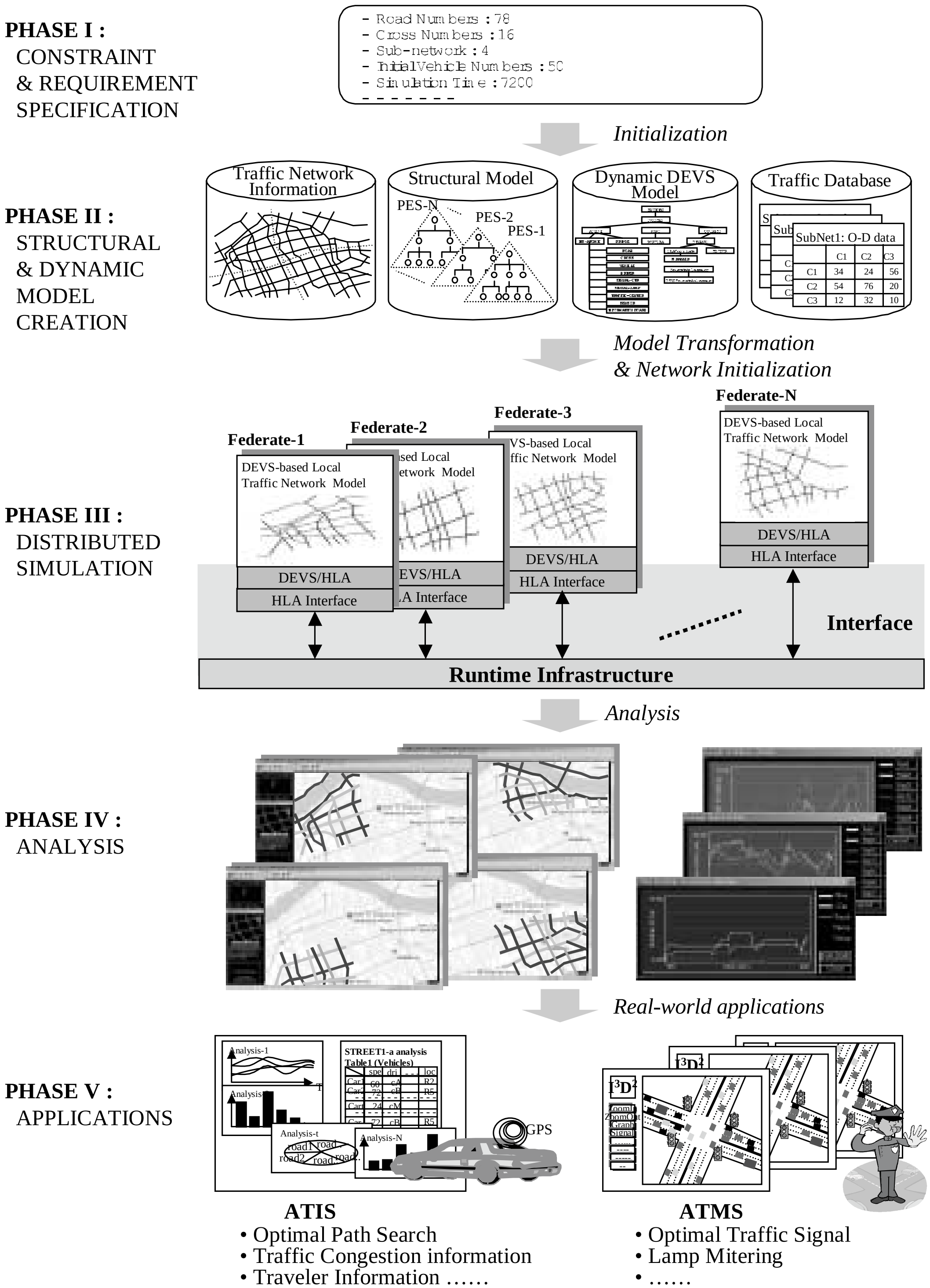


Figure 6. Distributed traffic simulation methodology

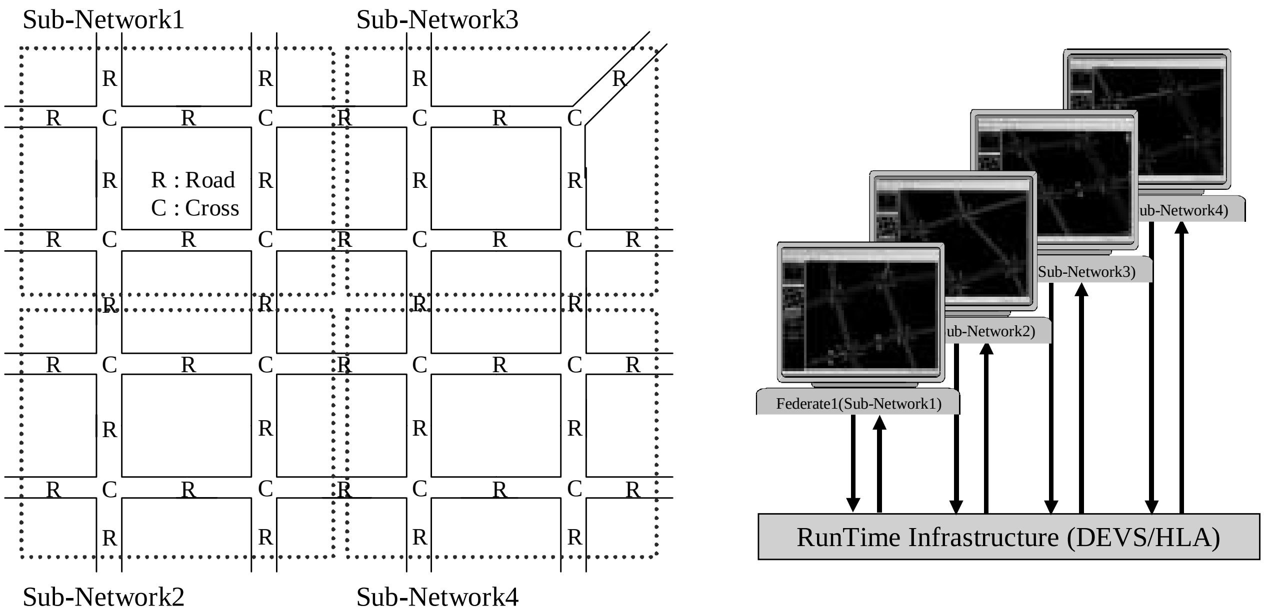


Figure 7. Decomposition concept of the large-scale traffic network

collecting information about vehicle flow. Figure 10 is a simple pseudo-code of the sensor model, which senses the vehicle's speed.

- *Signal lamp model (SL):* This provides signal information to the ROAD model; it is installed by turning on the signal according to the signal controller model and performs intersection control, such as stopping vehicles and passing them through. Figure 11 represents a simple pseudo-code of the signal lamp model.
- *Bridge model (BRIDGE):* This performs encoding and decoding of messages when models in each federate exchange messages through the WRAPPER model. Therefore, the BRIDGE model is suitable for applying the performance enhancement method, such as message reduction, to enhance performance of the network overhead that occurred in a distributed simulation environment. Lee, Chi, and Zeigler [27] performed research on the network performance enhancement method in a distributed traffic simulation by using predictive quantization-based filtering. Figure 12 represents a simple pseudo-code of the BRIDGE model.

The initial conditions for distributed simulation are as follows. The amounts of generating vehicles of each federate rely on the exponential distribution that λ is 1. The number of vehicles on the roads for the first time is 50. The signal phase system of crosses consists of simultaneous signals, including going straight and turning left. The signal phase order is clockwise, and the length of simultaneous signals on the roads is 30 seconds. Of the vehicles generated in vehicle-creating models, 75% are designed for going straight and 25% for turning left. After setting these test conditions initially, simulation was done in 7200 seconds.

4.2 Simulation Result

We built a test environment by using four PCs with the Windows 2000 operating system on an Intel Pentium III 700-MHz processor and a 125-Mbyte main memory according to test conditions outlined previously. Each PC installed four subnetworks onto an I^3D^2 transportation simulation system based on the DEVS/HLA-distributed simulation, which was performed in an RTI (version 3.33) environment. The test for simulation is divided in two ways: (1) when there are no special occurrences such as accidents on roads and (2) when there are no special occurrences but accidents do occur on a Federate4 during simulation. Figure 13 explains the result of the distributed simulation that uses the I^3D^2 transportation simulation system. Arrows show the direction of flow of the vehicles on roads, which were extracted to graph the simulation result. The cross-shaped plate in the center shows boundaries for separating the distributed four systems. Figure 13a-d shows that four federates in a normal state share information about vehicles and roads in an RTI environment of the HLA and that the flow of vehicles is normal. Figure 13a'-d' shows the riffle effect of congestion when accidents occur at Federate4 at simulation time 500. Figure 13b', which represents the road situation at simulation time 600, shows that congestion is mounting due to the riffle effect of the accident at simulation time 500. This riffle effect of congestion is effectively transmitted to Federate2 and Federate3 about

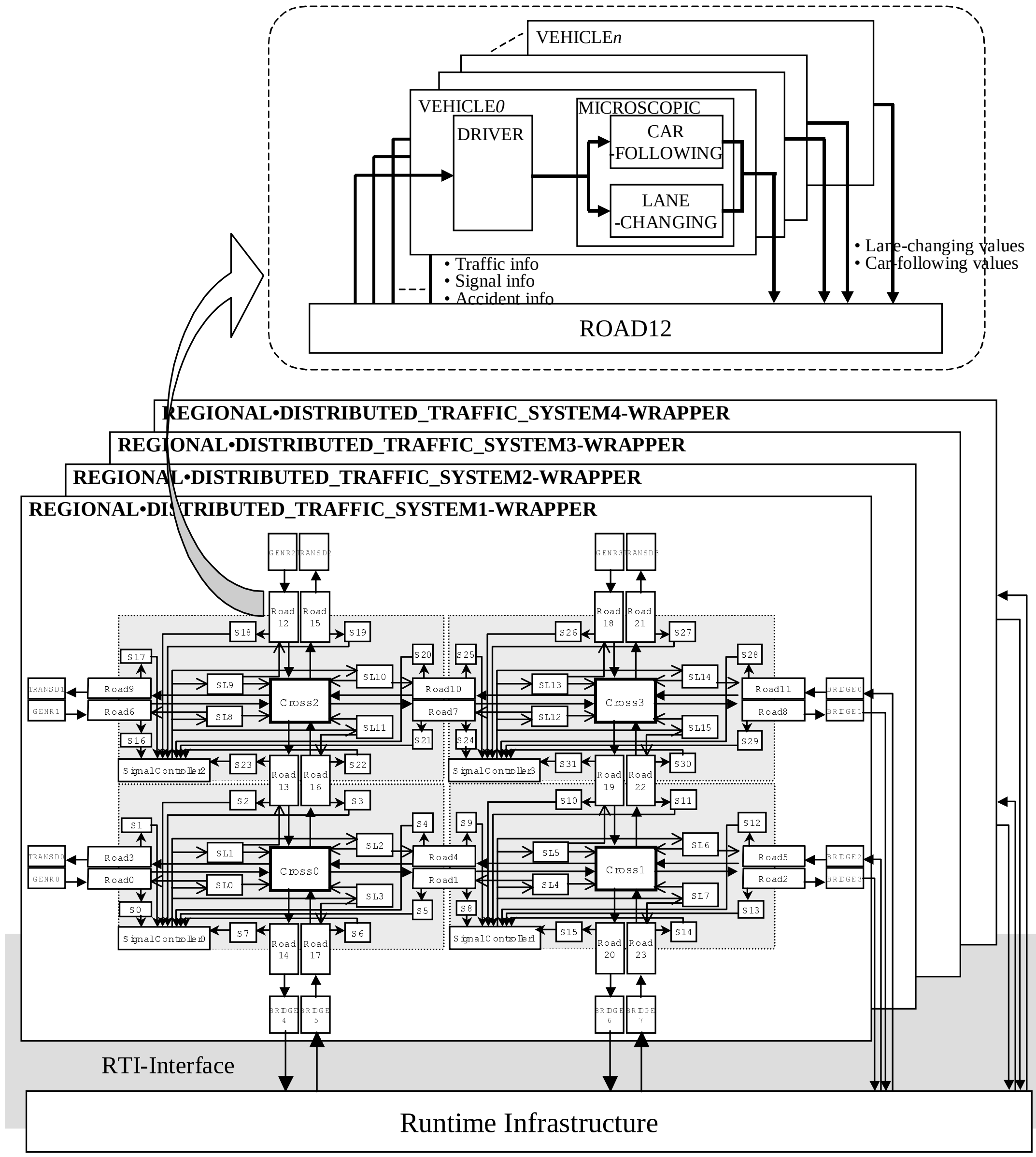


Figure 8. Distributed simulation model structure of the I^3D^2 transportation simulation system

the accident that occurred at Federate4. Congestion information about the vehicles (shown in Fig. 13c',d'), is distributed, and therefore we know Figure 13 represents effectively the riffle effect of congestion that can occur in accidents. Based on these results, traffic simulation in a DEVS/HLA-distributed modeling and simulation environment will be used as an effective tool for analyzing the traffic flow of large-scale traffic networks and building traffic plans. Also, by using distributed processing, performance was two times more efficient than the result from which we performed the simulation after setting 16 crosses, 78 two-way road networks, and 50 vehicles per road initially in the same system. However, we could not obtain ideal improved performance due to the overhead of the distributed environment, derived from the conservative asynchronous method of RTI, compared to that of distributed simulation based on resource use. Therefore, the study of how the DEVS/HLA environment can effectively reduce overhead

```
State variables
   phase, sigma, green signal time, yellow signal time, etc.
External Transition Function
   Receiving Sensor Info on port 'InSensor
            Update Congestion of Road
            Compute Signal Policy
Internal Transition Function
   Case phase 'yellow
            Compute green signal time
            Changing state 'green and set sigma = green signal time
   Case phase 'green
            Compute yellow signal time
            Changing state 'yellow and set sigma = yellow signal time
   - - - - - -
Output Function
   Case phase 'yellow
            Selecting signal lamp
            send yellow signal to selected signal lamp port 'YellowSignalOut
   Case phase 'green
            Selecting signal lamp
            send yellow signal to selected signal lamp port 'GreenSignalOut
  - - - - - -
```

Figure 9. Pseudo-code of the signal controller model

```
State variables
   phase, sigma, averages speed of road, collecting time, etc.
External Transition Function
   Receiving vehicle speed on port 'InRoad
            Update average speed of road
            Continue phase Collecting
Internal Transition Function
   Case phase 'Collecting
            Changing state 'SendingInfo and set sigma = 0
   Case phase 'SendingInfo
            Compute collecting time
            Changing state 'Collecting and set sigma = collecting time
Output Function
   Case phase 'SendingInfo
            send average speed of road to signal controller port 'SensorOut
```

Figure 10. Pseudo-code of the sensor model

generated during communications due to distributed simulation needs to be continued by taking advantage of the interoperability and reusability among the different simulation systems that has the HLA technique.

5. Conclusions and Future Work

This study suggested a DEVS/HLA-based distributed traffic simulation methodology that is a hierarchical modular modeling and simulation environment. To do this, we performed distributed homogeneous traffic simulation by extending an existing developed DEVS-based I^3D^2 transportation simulation system to an HLA-based distributed simulation environment. Therefore, the suggested methodology provides the following features compared to the distributed traffic simulation environment, which uses a distributed/parallel computing technology such as an existing PVM. First, it gives an object-oriented and hierarchical modular modeling and simulation environment of traffic models that have complicated dynamic features of the real world. Second, it provides easiness and extendibility of distributed traffic simulation system design/development.

```
State variables
   phase, sigma, averages speed of road, collecting time, etc.
External Transition Function
   Receiving yellow signal on port 'FromSignalController
            Changing state 'ChangingYellowSignal and set sigma = 0
   Receiving green signal on port 'FromSignalController
            Changing state 'ChangingGreenSignal and set sigma = 0
Internal Transition Function
   Case phase 'ChangingYellowSignal
            Changing state 'yellow and set sigma = INF
   Case phase 'ChangingGreenSignal
            Changing state 'green and set sigma = INF
Output Function
   Case phase 'ChangingYellowSignal
            send yellow signal to road model port 'YellowSignalOut
   Case phase 'ChangingGreenSignal
            send green signal to road model port 'GreenSignalOut
```

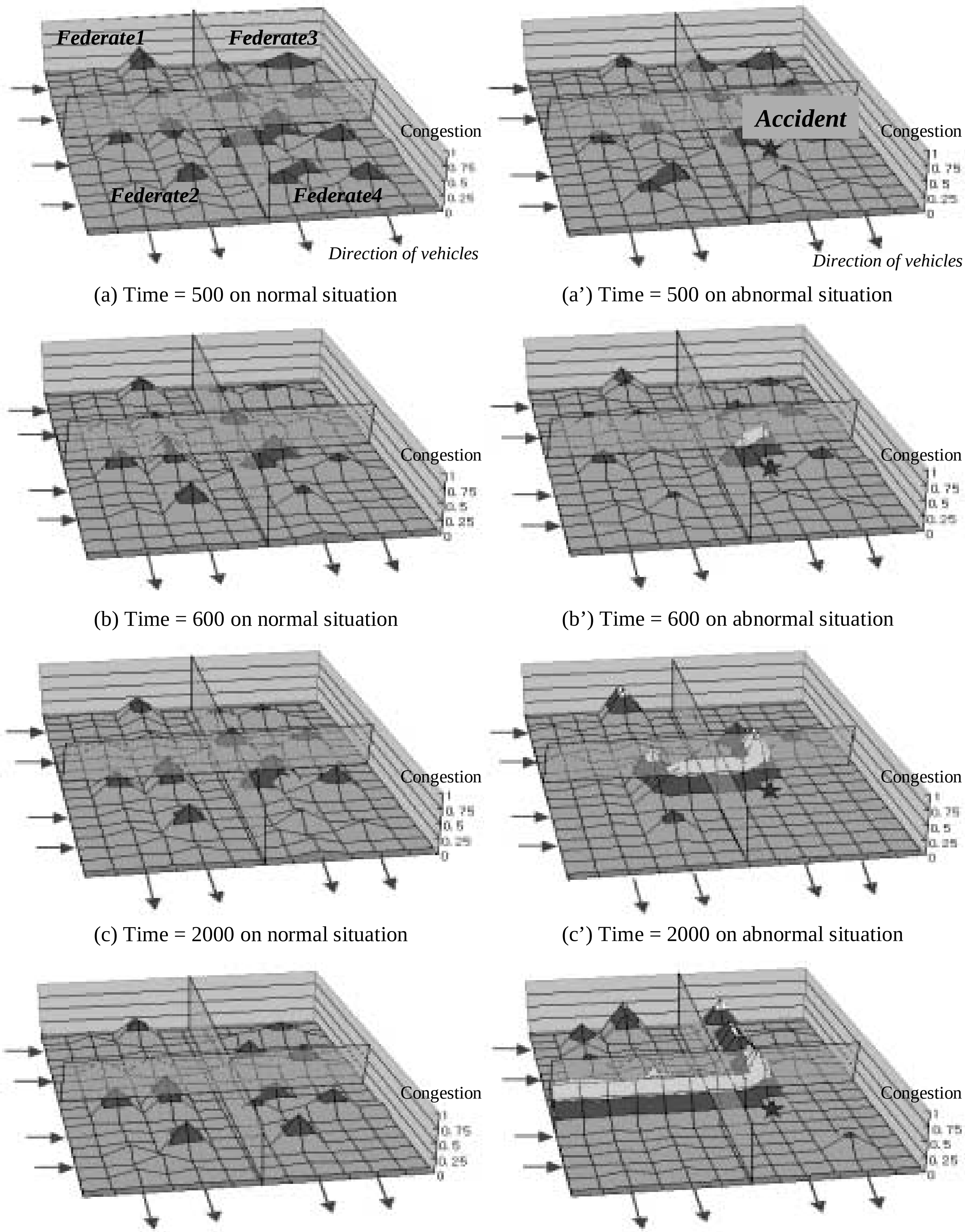
Figure 11. Pseudo-code of the signal lamp model

```
State variables
   phase, sigma, RemoteOut, RemoteIn, VehicleList etc.
External Transition Function
   Receiving Vehicle List on port 'inFromLocal
            Encoding Vehicle Info (Vehicle List ) ;
            Queue Add from Local ( Vehicle List ) ;
            Changing state 'passOut and set sigma = 0
   Receiving Last Vehicle Info on port 'inFromeRemote
            Decoding Last Vehicle Info (Last Vehicle Info );
            Queue Add from Remote ( Vehicle List ) ;
            Changing state 'passIn and set sigma = 0
Internal Transition Function
   Case phase 'passOut or 'passIn
            Changing state 'Idle and set sigma = INF;
Output Function
   Case phase 'passOut
            Queue Delete from Local(Vehicle List ) ;
            send Vehicle List to neighbor road in external federate port 'OutToRemote
   Case phase 'passIn
            Queue Delete from Remote (Last Vehicle Info )
            send Last Vehicle Info to connected road with external federate port 'OutToLocal
```

Figure 12. Pseudo-code of the BRIDGE model

Third, it offers the whole simulation time management method between distributed simulation objects through the HLA distribution standardization technique. Finally, it can offer an effective and hierarchical (road, local, city, etc.) simulation analysis environment of a large-scale road network. Therefore, the DEVS/HLA-based distributed traffic simulation methodology we suggest accepts complexity and extendibility of the ITS and offers interoperability and reusability so that it can be used as a tool to analyze, plan, and evaluate large-scale traffic networks as well as to conduct synthetic analysis and evaluation.

We showed that the DEVS/HLA-distributed modeling and simulation environment proposed in this study can be effectively applied to enhance the performance of a homogeneous traffic simulation system by dividing a large traffic network. Therefore, as a future work, the distributed simulation approach of a heterogeneous traffic system should progress through additional research of the DEVS/HLA-based heterogeneous simulation environment introduced in Kim, Kim, and Kim [28]. Also, overhead problems derived



(a) Time = 500 on normal situation

(a’) Time = 500 on abnormal situation

(b) Time = 600 on normal situation

(b’) Time = 600 on abnormal situation

(c) Time = 2000 on normal situation

(c’) Time = 2000 on abnormal situation

(d) Time = 5000 on normal situation

(d’) Time = 5000 on abnormal situation

Figure 13. Distributed transportation simulation result

from synchronization and data transmission among each federate of the distributed simulation should be solved.

6. Acknowledgment

The authors thank Y. H. Lim and Professor B. J. Kim at Hangkong University and Tae-Ho Cho at Sungkyunkwan University for their helpful comments. This research was supported by the Ministry of Science & Technology under contract 2000-N-NI-03-A-02, "Development of Simulation System Architecture and Application Systems."

7. References

[1] Yang, Q. 1997. A simulation laboratory for evaluation of dynamic traffic management systems. Ph.D. diss., Department of Civil and Environmental Engineering, MIT.

[2] Abdulhai, B., J. B. Sheu, and W. Recker. 1999. Simulation of ITS on the Irvine FOT area using "Paramics 1.5" scalable microscopic traffic simulator: Phase I. Model calibration and validation. California PATH Research Report UCB-ITS-PRR-99-12, University of California, Berkeley.

[3] Stretz, P., and K. P. Berkbigler. 1998. Transportation analysis simulation system (TRANSIMS). Research Report, Los Alamos National Laboratory, March.

[4] Nagel, K., R. Frye, R. Jakob, M. Rickert, and P. Stretz. 1998. Dynamic traffic assignment on parallel computers. In *Proceedings of ISCOPE-98*, May, Santa Fe, New Mexico.

[5] Niedringhaus, W. P., and P. Wang. 1993. IVHS traffic modeling using parallel computing. In *Proceedings of the IEEE-IEE Vehicle Navigation and Information Systems Conference*, Ottawa, pp. 157-62.

[6] Schwerdtfeger, T. 1994. DYNEMO: A model for the simulation of traffic flow in motorway networks. In *9th International Symposium on Transportation and Traffic Theory*. Utrecht, The Netherlands: VNU Science Press.

[7] Schmidt, M. 2000. Decomposition of a traffic flow model for a parallel simulation. In *Proceedings of AI, Simulation and Planning in High Autonomy Systems*, March, pp. 199-202.

[8] Schmidt, M., R. P. Schäfer, and K. Nökel. 1998. SIMTRAP: Simulation of traffic-induced air pollution. *Transactions of the Society for Computer Simulation International* 15 (3): 122-32.

[9] *Paramics v1.5 manuals*. 1997. Edinburgh, UK: Paramics Traffic Simulation Ltd.

[10] Gordon, D. B. 1996. PARAMICS—Parallel microscopic simulation of road traffic. *Journal of Supercomputing* 10 (1): 25-53.

[11] Eskafi, F. H. 1996. Modeling and simulation of automated highway system. Ph.D. diss., University of California, Berkeley.

[12] Eskafi, F. H., D. W. Hsu, and L. Baja. 1996. Distributed simulation of the automated highway system. In *Proceedings of the Sixth Annual Conference on AI Simulation & Planning in High Autonomy Systems*, pp. 11-17.

[13] Eskafi, F. H., and A. Göllü. 1997. Simulation framework requirements and methodologies in automated highway planning. *Transactions of the Society for Computer Simulation International* 14 (4): 167-80.

[14] Kosonen, I., and A. Bargiela. 1999. A distributed traffic monitoring and information system. *Journal of Geographic Information and Decision Analysis* 3 (1): 31-40.

[15] Argile, A., E. Peytchev, A. Bargiela, and I. Kosonen. 1996. DIME: A shared memory environment for distributed simulation, monitoring and control of urban areas. *Proceedings ESS'96*, Genoa, vol. 1, pp. 152-6.

[16] Schulze, T., U. Klein, and S. Straβburger. 1999. Migration of HLA into civil domains: Solutions and prototypes for transportation applications. *SIMULATION* 73 (5): 296-303.

[17] Klein, U., T. Schulze, S. Straβburger, and H.-P. Menzler. 1998. Distributed traffic simulation based on the high level architecture. In *Proceeding of Simulation Interoperability Workshop*, Orlando, FL.

[18] Chi, S. D., and J. K. Lee. 2001. DEVS-based modeling and simulation for intelligent transportation systems. In *Discrete event modeling and simulation: A tapestry of systems and AI-based theories and methodologies*, edited by H. S. Sarjoughian and F. E. Cellier, 215-27. Arizona: Springer-Verlag.

[19] Lee, J. K., Y. H. Lim, and S. D. Chi. Forthcoming. Hierarchical modeling and simulation environment for intelligent transportation systems. *Transactions of the Society for Computer Simulation International*.

[20] Zeigler, B. P., and J. S. Lee. 1998. Theory of quantized systems: Formal basis for DEVS/HLA distributed simulation environment. In *Enabling Technology for Simulation Science (II), SPIE AeroSense 98*, Orlando, FL.

[21] Zeigler, B. P., G. Ball, H. J. Cho, and J. S. Lee. 1999. Implementation of the DEVS formalism over the HLA/RTI: Problems and solutions. In *Simulation Interoperation Workshop (SIW)*, Orlando, FL.

[22] Kuhl, F., R. Wearherly, and J. Dahmann. 1999. *Creating computer simulation systems: An introduction to the high level architecture*. Upper Saddle River, NJ: Prentice Hall.

[23] Cho, H. J. 1999. Discrete event system homomorphisms: Design and implementation of quantization-based distributed simulation environment. Ph.D. diss., University of Arizona, Tucson.

[24] Zeigler, B. P. 1990. *Object-oriented simulation with hierarchical, modular models: Intelligent agents and endomorphic systems*. New York: Academic Press.

[25] Zeigler, B. P., H. Praehofer, and T. G. Kim. 1999. *Theory of modeling and simulation*. 2d ed. New York: Academic Press.

[26] Zeigler, B. P., S. B. Hall, and H. S. Sarjoughian. 1999. Exploiting HLA and DEVS to promote interoperability and reuse in Lockheed's corporate environment. *SIMULATION* 73 (5): 288-95.

[27] Lee, J. K., S. D. Chi, and B. P. Zeigler. 2003. Predictive, quantization based filtering for distributed traffic simulation. Paper presented at the Summer Computer Simulation Conference, July, Montreal, Canada.

[28] Kim, Y. J., J. H. Kim, and T. G. Kim. 2003. Heterogeneous simulation framework using DEVS BUS. *SIMULATION* 79 (1): 3-18.

Jong-Keun Lee *is currently a manager at Telematics Research Department in Cartronics R&D Center of Hyundai MOBIS. From September 2002 to August 2003, he was postdoctoral researcher of ACIMS in the University of Arizona. He received BS and MS degrees in 1996 and 1998, respectively, in computer engineering from Hangkong University, Seoul, Korea, and a PhD degree in computer engineering in 2002 from the Hangkong University, Seoul, Korea. From 2000 to 2001, he worked as project manager at the Mirae ITS Corporation, Seoul, Korea. His research interests include Intelligent Transportation Systems, Telematics Systems & Driver Information Systems, Distributed discrete-event modeling and simulation, War games, intelligent system design methodology.*

Min-woo Lee *is a software engineer at Sena Technologies, Inc., Seoul, Korea.*

Sung-Do Chi *is an associate professor at the Intelligent System Research Laboratory, Department of Computer Engineering, Hangkong University, Seoul, Korea.*