

# REAL-TIME DISTRIBUTED SIMULATION ENVIRONMENT FOR AIR DEFENSE SYSTEM USING A SOFTWARE FRAMEWORK

## **Byunggyu Cho**

Weapon System Modeling & Simulation Center, Agency for Defense Development  
Yuseong P.O.Box 35, Daejeon 305-600, Korea  
82-42-821-2721

[bgcho@naver.com](mailto:bgcho@naver.com)

## **Dae Young Kim**

LIGNex1 Co., Ltd.  
148-1, Mabuk-ri, Gusung-eup, Yongin-city,  
Kyonggi-do 449-910, Korea  
82-31-288-1116

[dykima@lignex1.com](mailto:dykima@lignex1.com)

## **Sae Hwan Kim**

LIGNex1 Co., Ltd.  
148-1, Mabuk-ri, Gusung-eup, Yongin-city,  
Kyonggi-do 449-910, Korea  
82-31-288-1122

[shkimc@lignex1.com](mailto:shkimc@lignex1.com)

## **Cheong Youn**

Department of Computer Science, Chungnam National University  
220 Gung-Dong, Yuseong-gu, Daejeon 305-764, Korea  
82-42-821-6250

[cyoun@cnu.ac.kr](mailto:cyoun@cnu.ac.kr)

To solve the limitations of the test scope, test period, and budget, we have been adapting Modeling and Simulation (M&S) techniques for Test and Evaluation (T&E) of the state-of-the-art weapon systems. This paper describes the Real-time Distributed Simulation Environment (RDSE) for the Korea's medium-range surface-to-air missile system (Chelmae-II). The RDSE will be used to support the test bed and performance analysis tools for the Chelmae-II system. To develop the RDSE, this paper proposes the Air Defense Simulation Framework (ADSF) which supports the High Level Architecture (HLA) and the TCP/IP as well as a real-time simulation engine. An ADSF was successfully applied to the RDSE, and a satisfied test result was acquired through the Global Positioning System (GPS) timer.

**Keywords:** Software framework, real-time distributed simulation environment (RDSE), HLA, T&E.

## **1. Introduction**

The advent of the latest Information Technology (IT) has enabled upgrades to the Modeling and Simulation (M&S) infrastructure to provide a feasible development environment for the state-of-the-art weapon systems. Advanced countries are using a new paradigm based on Simulation Based Acquisition (SBA) technology in system development [1]. In the Korean Agency for Defense Development (ADD), we are developing the Korean medium-range surface-to-the-air missile system (Chelmae-II) and other weapon systems, using M&S technologies [2].

However, we face a number of problems in the Test and Evaluation (T&E) of the Chelmae-II system. First, the test range volume is not suitable for some T&E parameters, such as the maximum range test and the simultaneous multiple engagement capability. Second, the target drone is not appropriate for the maximum height tests. Third, the budget and the period of development apply pressure to the T&E activity. Fourth, safety control systems and test facilities in the test range are very coarse for T&E of state-of-the-art weapon systems. Finally, an atmosphere of current defense system development requires the latest T&E techniques. An alternative plan is needed [3].

Advanced countries have applied M&S technologies in T&E. The Simulation Test and Evaluation Process (STEP) and Model-test-Model techniques have been successfully utilized for T&E of the weapon systems [4]. Using the latest IT, we are able to integrate real weapon systems and simulators in the synthetic environment. Therefore, we are determined to develop a Real-time Distributed Simulation Environment (RDSE) that connects real Chelmae-II weapon systems and simulators for T&E [3].

However, due to lack of interoperability among simulation models developed for a variety of purposes, application of such models has not been fully satisfactory. To overcome these weaknesses, the U.S Department of Defense (DoD) proposed the High Level Architecture (HLA) for interoperability among simulations and registered an Institute of Electrical and Electronic Engineers (IEEE) standard 1516 in 2000 [1]. U.S. military M&S systems have adopted the HLA standard. The ChangJo21 was developed for the Army, the ChungHae for the Navy while the ChangGong and the ChunJaBong are under development for the Air Force and the Marines, respectively, in compliance with the HLA in Korea [5, 6, 7].

For better M&S results, the RDSE should follow the HLA rules and support Transmission Control Protocol/Internet Protocol (TCP/IP) and User Datagram Protocol (UDP) interfaces used in real weapon system interfaces. However, we had to learn complicated development techniques to obey the HLA rules and network protocols. For that reason, we can not easily program in the HLA field and the networking field [8, 9]. However, we need to develop a reliable communication service for the RDSE. Moreover, a real-time simulation engine that runs orders rapidly to produce precise simulation for the advanced weapon system is needed. To satisfy these requirements, developers who use the elements of the RDSE should study complex communication programming techniques, including the HLA compliance technique, and develop their own real-time simulation engine. To reduce redundancy, we developed the Air Defense Simulation Framework (ADSF) which supports several communication services, including the HLA, and supports real-time simulation for the air defense system [10].

This paper is organized as follows. Related work is presented in Section 2. Section 3 illustrates details of the ADSF. In Section 4 we explain research results related to on performance of the real-time distributed simulation when the ADSF is applied to RDSE; especially for system simulators. In conclusion, future work is presented in Section 5.

## **2. Related Work**

### *2.1. The Concept of Real-time Distributed Simulation*

A simulation is said to be executing in real-time when a constant relationship is maintained between the local simulation time and the Wall-Clock Time (WCT) [11]. Simulation time is the internal time maintained in the simulation. Generally, the WCT referencing the real-time is the system time. During the real-time simulation, the time constraint is very important. Time constraint refers to the completion condition of simulation computations during a constant period. Hard real-time simulation and soft real-time simulation are divided according to the degree of satisfaction of the real-time constraint. In hard real-time simulation, there will be a fatal error in function or missions if the time constraint is not strictly kept, such as for Hardware-in-the-Loop Simulation (HILS). In soft real-time simulation, there will be a deterioration of performance if the follow-time constraint is not kept.

Generally, a real-time distributed simulation exists when a real-time simulation can successfully inter-operate on different nodes in distributed computing environments. Real-time distributed simulation must guarantee satisfying the time constraint as well, in order for no causality errors to occur when delivering messages among distributed simulation nodes [11, 12].

### *2.2. The Concept of HLA*

The HLA is a general-purpose architecture for simulation reuse and interoperability. The HLA was developed under the leadership of the Defense Modeling and Simulation Office (DMSO, now called the M&S Coordination Office) to support the reuse and interoperability across the large numbers of different types of simulations developed and maintained by the DoD in the middle 1990's [1].

The HLA is defined by a set of Rules, the Object Model Template (OMT) and the Interface Specification in detail. The Run Time Infrastructure (RTI) is middleware that is derived from HLA interface rules and is used when the simulation is connected with other HLA simulation systems. To develop defense simulation in the U.S, developers observe the HLA rules, and attempt to apply HLA to commercial game simulation and educational simulation [11]. Individual distributed simulation systems that comply with the HLA standard are a group unit of a federation. The federation consists of each distributed simulation object (federates), the RTI, and the Federation Object Model (FOM) [1, 5].

### *2.3. Software Framework*

A Software framework can be defined as a set of interacting classes designed to reuse specific application field software [8]. Consequently, a software framework can be defined as a reusable, "semi-complete" application that can be specialized to produce custom applications [14]. A developer customizes the framework to a particular application by sub-classing and composing instances of framework classes. The benefits offered by the software framework can be summarized by the term: "reuse". Through the reuse of proven software providing higher-level services while encapsulating lower-level details, substantial gains in productivity across the development of multiple applications in the software domain supported by the framework are achieved. Indeed, software reuse can reduce time, defect density, maintenance costs, and overall development costs [8].

### *2.4. Previous Studies of HLA Related Software Framework*

Kevin Cox [6] proposed the HLA Federation Class (HFC) and the HLA Automation Tool (HAT) to support various kinds of HLA federates. Belanger et al. [14, 15] proposed the OSim framework as an

HLA federate development tool. The OSim framework consists of the OSim BulletinBoard supporting distributed communication, the OSim Executor supporting simulation engine and the simulation object, and the OSim Generator generating C++ and Java frame code taking advantage of the BulletinBoard and Executor information. Jay Graham et al. [16] suggested the FedProxy tool to generate a proxy federate formed automatically from the Java frame code in the object model. The FedProxy simulation engine supports time and event scheduling services, object management services, and simulation execution control services. Elias and Huiskamp [17] proposed an advanced simulation framework (ASF) which provides the flexibility for migrating from Distributed Interactive Simulation (DIS) to HLA without significant changes to the simulation module itself. The ASF is comprised of a software framework that is commonly used simulation functions to promote software reuse. Pilloud and Kanko [18] proposed SIMWORX, a reusable object-oriented application framework for distributed simulation which is compliant with both the DoD HLA for M&S and DIS standards. SIMWORX is implemented in Ada95 using a software engineering method. Yuan et al. [19] proposed a framework technique that equalized task burdens by using a Logical Process (LP). Kim et al. [5] proposed the RTI Object Model (ROM) framework to separate a RTI program from the simulation program. The ROM is composed of a RTI service portion, the state management of the object, and a portion composed of data management. The ROM framework is a successfully-applied development of interlocking models such as ChangJo 21.

Summarizing previous studies, a trend exists to separate communication service parts, including the HLA, from simulation engine parts, in the software framework. Deficiencies in existing studies include simulation engines to support precise, real-time simulation, and communication framework components to provide reliable message transmission.

### *2.5. HLA-Based Air Defense System Test bed*

The Raytheon, in USA developed and has been using a Flight Mission Simulator (FMS) for T&E of the Patriot system. The FMS is a T&E tool based on M&S techniques for T&E categories that are not easy to obtain through live flight tests. The FMS demonstrated the possibility of changing the paradigm from live flight tests to M&S-based tests in T&E of the air defense system [20]. Additionally, the Raytheon company developed the Standard Missile (SM)-3 Test bed which is HLA-based and is using the system level test [21]. The Germany Army developed the Gepard Hardware-in-Loop (HIL) facility based on M&S techniques, especially the HLA, and has applied this approach to the Development Test (DT) and the Operational Test (OT) of the Gepard air defense system [22].

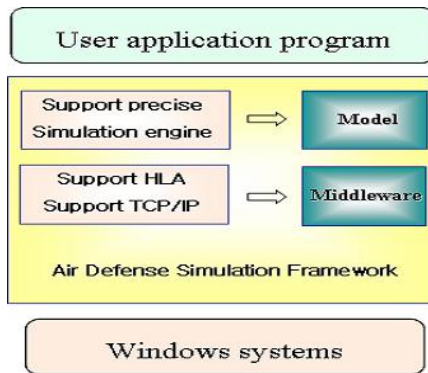
### *2.6. Necessity of ADSF*

We examined the HLA to as the communication protocol for simulators in the RDSE. Usually, real weapon systems communicate with each other using the TCP/IP or the UDP. Therefore, communication middleware to support all HLA, TCP/IP, and UDP communications is necessary. The RDSE is composed of a large number of component parts. These component parts create an Air Defense System Simulation Federation using network connectivity. Each component has a simulation engine sub-component to achieve the desired simulation. Necessarily, the developer enhances reuse by applying techniques to the common framework rather than developing the communication middleware and the engine of the real-time distributed simulation individually. Previously, HLA-related software frameworks did not implement applications for real weapons such as advanced air defense systems. Additionally, to develop simulation engines supporting over 100 Hz is requested because we could not

easily find frameworks that support 100 Hz accuracy. To support our real-time distributed simulation, we designed the delay time of messages among system simulators to be less than 50ms. Therefore, it is necessary to apply software frameworks including communication and simulation engines to meet the needs of the real-time distributed simulation in RDSE.

### 3. The ADSF

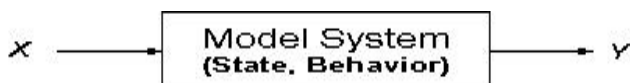
The purpose of the ADSF is to provide common functions used by each air defense system federate. An ADSF is composed of the Middleware parts (providing the communication function) of the main function in ADSF, and the Model Architecture part (providing the real-time simulation engine), as shown in [Figure 1].



**Figure 1.** The structure of the ADSF

#### 3.1. The Model Architecture

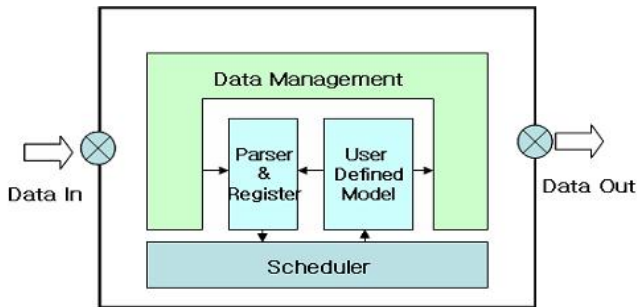
The proposed model architecture was developed based on the discrete model system, shown in [Figure 2]. Model state is defined by state variables, behavior is modeled by internal functions that operate on events.



**Figure 2.** The model system within the model architecture

##### 3.1.1. Composition of the Model Architecture

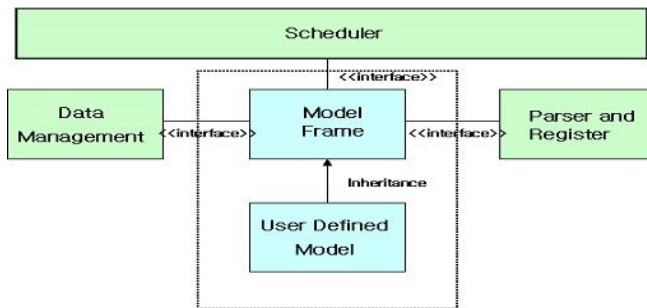
The primary composition of the model architecture is composed of the Data Management, the Parser/Register, and the Scheduler as shown in [Figure 3]. Data is entered using the interface in the model frame and is subsequently transmitted to the User Defined Model. Transmitted data is processed in the User Defined Model and is generated to the outside through the data interface.



**Figure 3.** The composition of the model architecture

### 3.1.2. The Function of the Model Architecture

The User Defined Model with a specified states and behavior can change its state through internal processing or an in response to an input message. The User Defined Model is a frame that gathers outer interfaces and common functions of a model. The User Defined Model developer should develop other sections based on the frame. The User Defined Model inherits the model frame as shown in [Figure 4]. See Table 1 for a detailed description of model frame.



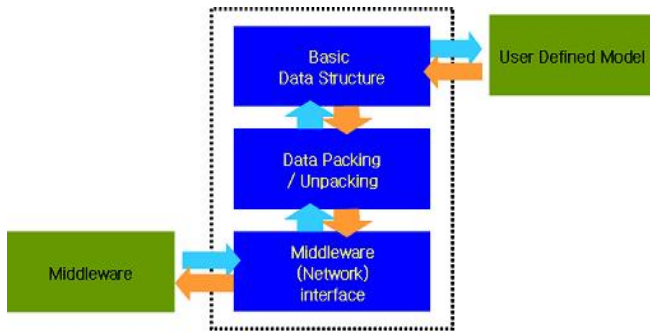
**Figure 4.** The User Defined Model

Detailed Function	Description
Basic data structure	Provide data structure through Data Management
Callback register	Using Parser/Register
Callback remove	Using Parser/Register
Data output	Through Data Management
Data initialization	Implementation of the base function in the model frame

**Table 1.** Detailed functions of the model frame

The function of Data Management is to supply the data structure for communication and packing/unpacking services for communication data. Figure 5 illustrates the data flow from the

middleware to the basic data structure in the model frame. See Table 2 for a detailed description of data management.

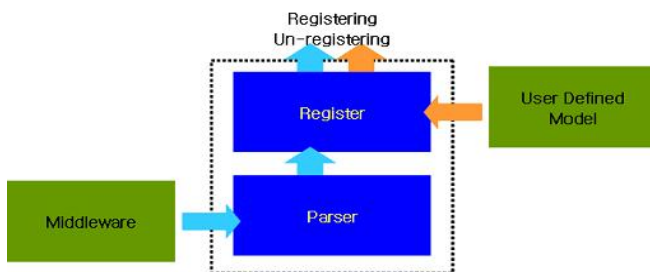


**Figure 5.** Data management

Detailed Function	Description
Provide basic data structure	Provide the basic data structure available to the User Defined Model
Provide packing/unpacking of data	Provide packing/unpacking of receiving or sending data through middleware
Provide interface to middleware	Provide an interface to the middleware for data communication

**Table 2.** Detailed functions of data management

The Parser parses the data from the outer system and registers the callback function that operates the User Defined Model in the Scheduler. The developer should use the register to start parsing the data and to finish parsing the data. To start periodic or non-periodic function calls, and to finish those in the User Defined Model, we applied the schedule to the Scheduler using the register. The common model frame provides the operational interface. Figure 6 illustrates the structure of the Register and the Parser while Table 3 provides additional information about these functions.



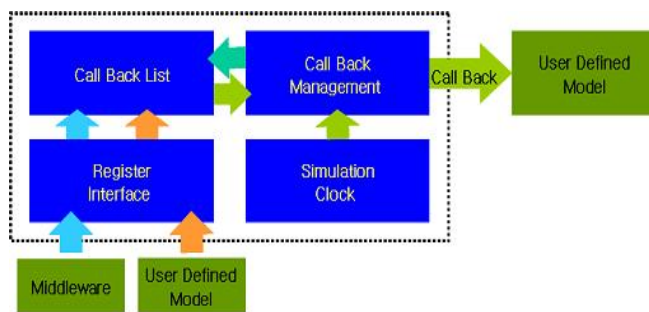
**Figure 6.** The structure of the Parser and the Register

Detailed Function	Description
-------------------	-------------

Parsing	Determine the type of external data from Data Management
Callback registration	Register the Callback in the Scheduler
Callback removal	Remove the Callback in the Scheduler

**Table 3.** The detailed functions of the Parser and the Register

Basic functions of the Scheduler control events that affect the User Defined Model when a simulation is running. The definition of an event is the user-defined (registered) function for running a simulation. A callback function can be registered or un-registered to the Scheduler using the register interface. The Scheduler can call a user-defined callback function based on the simulation clock. The registered user-defined callback function lists are arranged according to the simulation time. The callback manager determines whether or not to call the callback function by the callback priority. Figure 7 illustrates the registering process of the callback from the User Defined Model and the calling process of that callback. Table 4 describes the functions of the Scheduler in more detail. For measure of the ADSF Model operation accuracy use the Global Positioning System (GPS) board timer or multimedia timer.



**Figure 7.** The structure of the Scheduler

Detailed function	Description
Callback management	Inquiry Callback list and call User-defined model function based on the simulation time
Provide interface to Parser	Interface the register or remove the request through the parser

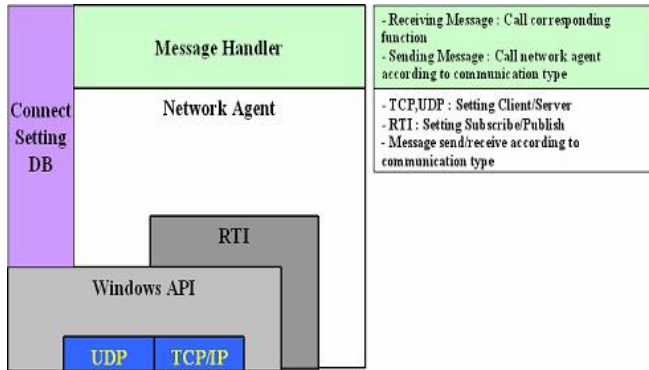
**Table 4.** The detailed functions of the Scheduler

### 3.2. Middleware

For ensure the consistent quality of the network interface service and to make connections with real-systems (TCP/IP, UDP) and simulators (HLA), we developed middleware with TCP/IP, UDP and HLA functions. Figure 8 illustrates the structure of the middleware.

Functions of the Middleware positioned program layer in the defined DoD 4 layer include:

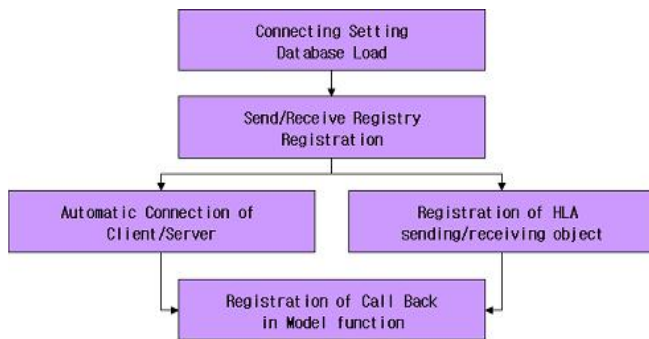
- Automatically sets TCP/IP communication
  - A. Creates server and client
  - B. Transfers the TCP/UDP message
  - C. Monitors and transfers (send/receive) the application message



**Figure 8.** The structure of the middleware

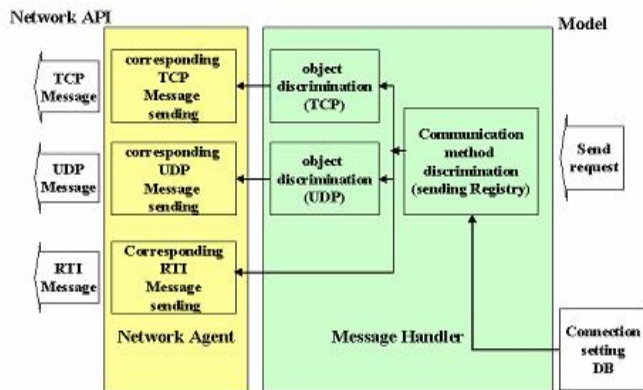
- Automatically sets HLA communication
  - A. Publishes and Subscribes
  - B. Transfers the HLA message
  - C. Sends and receives the application message

To publish refers to sending the message to other federates and subscribing the means to receive the message from other federates in the HLA simulation. Figure 9 illustrates the basic middleware settings proposed to communicate with other systems using the HLA and the TCP/IP.

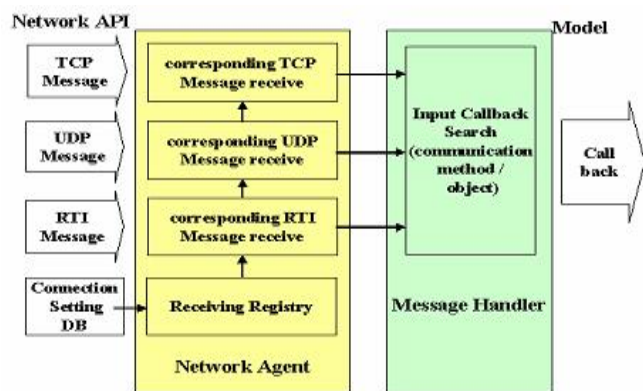


**Figure 9.** Setting of middleware communications

The middleware user can set the client-server HLA publish/subscribe object. The user can specify the corresponding callback function using the middleware Application Program Interface (APIs) and a script file. Figure 10 illustrates the data sending mechanism for the middleware. To send data to other systems, the user requests sending middleware with a script file or a connection name. The middleware sends data to other systems using the defined connection-setting information. Figure 11 illustrates the data receiving mechanism for the middleware. The middleware transfers the received data which corresponds to each different protocol with the user-defined callback function to the user application.



**Figure 10.** Sending data using the middleware



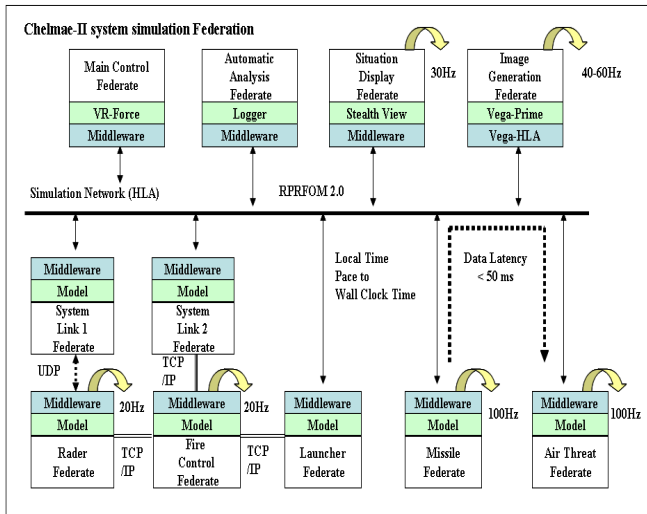
**Figure 11.** Receiving data using the middleware

#### 4. The RDSE

Using the ADSF enabled the design and implementation of the RDSE by requiring reduced resources (cost, manpower, and time period). In 2005, we completed the development of the exploratory version of the Chelmae-II System Simulator (CSS) in the RDSE. The CSS is an evaluation tool for the application M&S techniques that are used for systematic estimation.

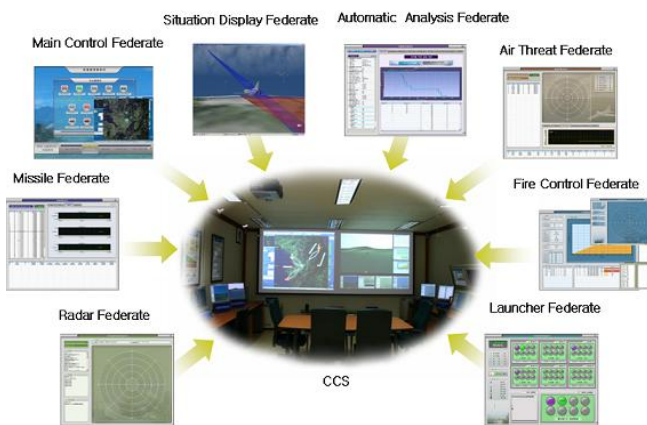
##### 4.1. The Development Environment

The CSS was programmed using the C++ language under Microsoft Windows 2000. Figure 12 illustrates the construction of the entire federation.



**Figure 12.** The CSS federation

We applied the ADSF to every component of the system simulation federation except the image generation device that uses a commercial product. Figures 13 and 14 illustrate sample displays of federates in the system.



**Figure 13.** The sample display of the CCS

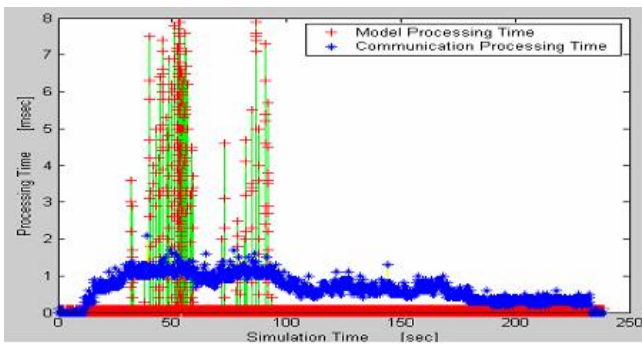


**Figure 14.** The sample display of others federates

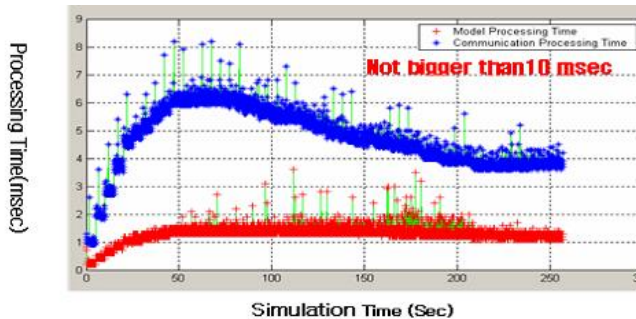
#### 4.2. Test and Result

To estimate the performance of the general simulation software framework, it is necessary to measure the renewal cycle of a side view of the communication delay time and the simulation engine [14, 15]. The ADSF must support the real-time distributed simulation. To test this, we simulated the next three related subjects. Moreover, the Network Time Protocol (NTP) and the window timer for measuring time according to a previous publication resulted in a 1ms~30ms time accuracy error [13]. The former method was unsuited for the performance estimation of the ADSF. Therefore, we used the bc637PCI GPS board timer that has 1 $\mu$ s up precision [23].

Generally, if one cannot conclude logically whether the distributed simulation operates in real-time or not, the property of real-time is determined by confirming that the requirement is satisfied under the worst condition in the system. The following simulation results were obtained when the maximum numbers of missiles were fired. According to the requirements of the CSS, the missile federate and the air threat federate were processed at 100Hz (handling time is less than 10ms). As the result of applying the ADSF, we obtained satisfactory values demonstrated in Figures 15 and 16. Table 5 describes the processing time of others important federates in the CSS.



**Figure 15.** The processing time of the missile federate



**Figure 16.** The processing time of the air threat federate

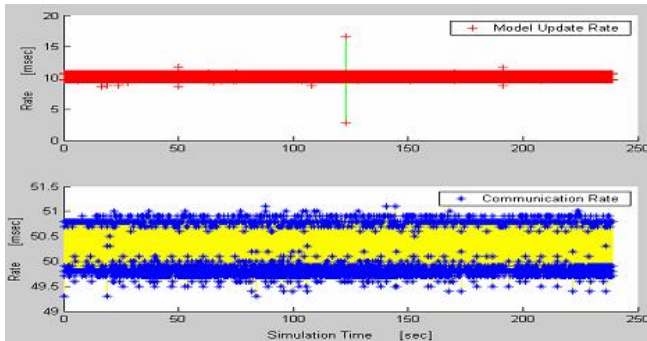
Federate	Processing time(worst case)
Fire Control Federate	7.8 ms
Launcher Federate	6.7 ms

Radar Federate	8.1 ms
----------------	--------

**Table 5.** The Processing time of others important federates

#### 4.2.1. The Accuracy of the Periodic Simulation Time

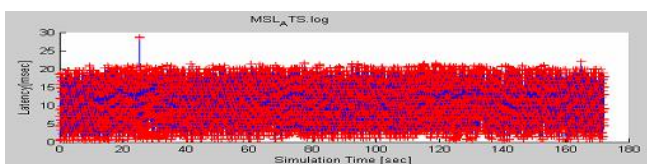
The information of the missile federate must be renewed on a 100Hz basis. The result of the testing of the renewal period accuracy is presented in Figure 17.



**Figure 17.** The accuracy of the periodic simulation time and the periodic data sending time

#### 4.2.2. The Accuracy of the Periodic Data Sending Time

For real-time distributed simulation, much information will be transmitted or received. The information of the missile federate is transmitted periodically as shown in Figure 17. To follow the HLA, communication was performed by using the RTI. However, the RTI generates a high calculation load, resulting from the internal working load, which can result in a communication delay. Currently multiple studies for the performance analysis of the RTI are in progress [13, 24, 25]. The CSS missile federate requires that the message delay time is less than 50ms. As the result of applying the ADSF, we obtained satisfactory results, as demonstrated in Figure 18. In general, the message delay time was less than 20ms, and under the worst condition, a satisfactory delay time of less than 50ms was generated.



**Figure 18.** The missile-to-air-threat message delay time

#### 4.3. The Effect of the CSS in the RDSE

The CSS was the first system in Korea to link real weapon systems and simulators. The CSS is a system used to link live simulation and virtual simulation. Existing domestic HLA linkage systems generated constructive simulations. By applying the ADSF to the CSS, the general M&S engineers do not need to learn complicated HLA techniques. Moreover, by supporting the framework form of established the HLA application model, the system offered greater convenience for verification and confirmation and enabled expansion to the surface-to-air missile system in the M&S field.

## 5. Conclusion

We applied the HLA that is a standard of M&S for RDSE development. Additionally, to link with real weapon systems, TCP/IP and UDP standards required support. The CSS in the RDSE is a real-time distributed simulation system, structured of various components, which supported simulation engine development that required distributed simulation. Hence, we developed the ADSF as a software framework that supports the HLA, the TCP/IP and the UDP, in addition to the real-time distributed simulation. The ADSF developed was applied successfully to the CSS in the RDSE. We received a certificate of compliance from the DMSO in August, 2004. The ADSF will be applicable to other real-time distributed simulation systems, and, in the future, we will actively apply the ADSF to other Chelmae-II M&S applications. The ADSF, presented in this manuscript, is supported in the Windows 2000, or the Windows XP environments. In the future, it will need to be improved to a framework that supports other operating systems for the spread of applications.

## 6. References

- [1] "Defense Modeling and Simulation Office." Available from: <http://www.dmsomil.com>, <http://www.modelbenders.com/sba/SBA-Preview.PDF> .
- [2] Cho B. G. "The Application of the M&S in the KMSAM." In *Proceeding of the 2nd Conference of the Weapon System Modeling and Simulation Korea*, C1-C17. 2003.
- [3] Cho B. G. "The Development Chelmae-II System Simulator for T&E based on M&S." In *Proceeding of the 3rd conference of the Weapon System Modeling and Simulation Korea*, C51-C69. 2006.
- [4] Jones P. "Model-Test-Model Methodology A critical component of Simulation Based Acquisition.", *Tutorial on 9th Annual Modeling and Simulation workshop*, 2003.
- [5] Kim D. S. "A Design and Implementation of ROM Framework for Developing HLA Federate." *Journal of Korea Information Science Society: Computing Practices*, VOL. 9-D NO.06, 1137-1144. 2002.
- [6] Cox K. "A Framework-based Approach to HLA Federate Development." *Simulation Interoperability Workshop*, 1998. Available from: <http://www.sisostds.org>
- [7] Kim J. H. et al. "Design and Implementation of Simulators Interoperation Layer for DEVS Simulator." In *Proceeding of the 2006 SCS International Conference on Modeling and Simulation – Methodology, Tools, Software Applications*, 95-100. 2006.
- [8] Kim T. G. et al. "DEVS Framework and Toolkits for Simulators Interoperation Using HLA/RTI." In *Proceeding of Asia Simulation Conference/the 6th International Conference on System Simulation and Scientific Computing*, 16-21. 2005.
- [9] Douglas C. S. "The ADAPTIVE Communication Environment (ACE)." Available from: <http://www.cs.wustl.edu/~schmidt/ACE.html>
- [10] Cho B. G. et al. "A Design and Application of HLA-Based Air Defense Simulation Framework." *Journal of Korea Information Science Society*, VOL. 12-D, NO.5, 709-718. 2005.
- [11] McLean T. "Hard Real-Time Simulation using HLA." *Simulation Interoperability Workshop*, 2001. Available from: <http://www.sisostds.org>
- [12] Tietje H. "Benchmarking of RTIs for Real-Time Applications." *Simulation Interoperability Workshop*, 2003. Available from: <http://www.sisostds.org>

- [13] Choi S. Y. et al. "Performance Measurement and Analysis of RTI in the HLA-based Real-time Distributed KMSAM Simulation." *Journal of Korea Information Science Society: Computing Practices*, VOL 11 NO. 2, 149-156. 2005.
- [14] Belanger J. P. et al. "Lessons Learned from a framework approach to HLA-Compliant Federation Development." *Simulation Interoperability Workshop*, 1998. Available from: <http://www.sisostds.org>
- [15] Belanger J. P. et al. "OSim Framework. Experience of an In-flight Refueling Federation Development Using Commercial Tools." *Simulation Interoperability Workshop*, 1997. Available from: <http://www.sisostds.org>
- [16] Graham J. et al. "HLA Object Models as Software Object Models." *Simulation Interoperability Workshop*, 1998. Available from: <http://www.sisostds.org>
- [17] Elias R. J., Huiskamp W. "Advanced Simulation Framework: A Generic Approach to Distribution Simulation." In *Proceeding of the 8<sup>th</sup> International Training and Education Conference (ITEC '97)*, 1997.
- [18] Capt. Pilloud E. C., Maj. Kanko M. A. "SIMWORX: An Ada 95 distributed simulation application framework support HLA and DIS." In *Proceeding of Aerospace and Electronics Conference*, 732-739. 1997.
- [19] Yuan Z. et al. "A framework for executing parallel simulation using RTI." In *Proceedings of the Seventh IEEE international symposium on Distributed Simulation and Real-Time applications*, 2003.
- [20] Capt. Yuliano A. "Simulations changing the paradigm for Air Defense Operational Testing." Available from: <http://airdefense.bliss.army.mil/adamag/April2001/Simulate.htm>, 2001.
- [21] Manthy R. S. "Navy Theater Wide Hardware-in-the-Loop End-to-End Simulation Using Raytheon's HLA based Standard Missile-3 System Testbed." *Simulation Interoperability Workshop*, 1999. Available from: <http://www.sisostds.org>
- [22] Lessmann K. et al. "Germany/USA Transatlantic Distributed Simulation Project." *Simulation Interoperability Workshop*, 2002. Available from: <http://www.sisostds.org>
- [23] 2003. *Symmetricom bc635/637/PCI/CPC/PMC Time and Frequency Process Revision J User's Manual*, 1-1 3-3.
- [24] Knight P. et al. "WBT RTI Independent Benchmark Tests: Design, Implementation, and Updates Results." *Simulation Interoperability Workshop*, 2002. Available from: <http://www.sisostds.org>
- [25] Kuiper P. et al. "Performance Measurements of a HLA Component-based Fighter Pilot Station." *Simulation Interoperability Workshop*, 2002. Available from: <http://www.sisostds.org>

## Author Biographies

**Byunggyu Cho** is a senior researcher in the Agency for Defense Development (ADD), Korea since 1990. He graduated with a B.S. in Computer Science from Inha University in 1988 and he completed an M.S in Computer Science from Inha University in 1990. Additionally, he is studying for a Ph.D in Computer Science at Chungnam National University. He was the project manager for the Chelmae-II System Simulator development at the 3rd System Development Center, ADD. The primary focus of his career has been air defense system M&S.

**Dae Young Kim** is a senior researcher at LIGNex1, Korea since 1998. He graduated with a M.S. in Control Engineering from Ajou University in 1998. He is currently the system engineer for the Chelmae-II System Simulator, the Chelmae-II Training Simulator and the Chelmae-II Testbed at System Research Center, NEX1. The primary focus of his career has been M&S.

**Sae Hwan Kim** is a project manager at LIGNex1, Korea since 1998. He graduated with a M.S. in Electronics from Kyungbook National University in 1987. He is currently the project manager for the Korean RTI, the Chelmae-II M&S System and Lynx EW Software. The primary focus of his career has been M&S and Electronic Warfare.

**Cheong Youn** is a professor in the department of Computer Science, Chungnam National University. He graduated with a B.S. in Physics from Seoul National University in 1979. He completed an M.S. in Computer Science from Illinois University, 1983, and a Ph.D in Computer Science from Northwestern University in 1988. He was a senior researcher at Bell Communication from 1988 to 1993. He has been working in the software engineering field for 13 years.