Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

USE OF THE ANALYTICAL SYSTEM AS THE DECISION SUPPORT SYSTEM FOR THE HLA JOINT TRAINING ENVIRONMENT

Jiung-yao Huang

Dept. of Computer Science and Information Engineering
National Taipei University
San Shia, Taipei, 237 TAIWAN

Ming-Chih Tung

Dept. of Computer Science and Information Engineering
Ching Yun University
Taoyuan, TAIWAN

Ji-Jen Wu and Shu-Shen Wai

Dept. of Computer Science and Information Engineering
Tamkang University
Tamsui, Taipei County, 25137, TAIWAN

Chung-Hsien Tsai

Dept. of Computer Science and Information Engineering
National Central University
Taoyuan County 32001, TAIWAN

ABSTRACT

This paper proposes an approach to join the analytical system to the real-time operational war game system. The interoperability issues of these two heterogeneous war game systems are fully explored in this paper. The purpose of interoperability among heterogeneous systems is to extend the capability of the joint systems beyond their originally designed purposes. This paper starts with the purpose of interoperability to categorize the types of the heterogeneous joint simulation. The technical issue and architecture of the heterogeneous joint simulation environment are then explored. Our experiment shows that the proposed method of using the analytical system as the decision support system is feasible and efficient approach to provide a heterogeneous joint training environment.

1 INTRODUCTION

There have been two principal paradigms to construct parallel and distributed simulations over the past few years (Ferenci 2000). The first method is widely used by the parallel discrete event simulation (PDES) research community. It defines a parallel simulation engine, programming languages, libraries, or tools to create proprietary high performance simulating environments. Numerous examples of this approach include Task-Kit (Xiao 1999), SPEEDES (Steinman 1992), and TeD/GTW (Bhatt 1998, Das 1994) et al. Since the simulation models are environmentally specific for this approach, it is difficult for this approach to port models into different environments.

A second paradigm that has emerged in the distributed simulation community is using a standardized protocol interface to interconnect the distributed nodes of simulation. This approach is used in Distributed Interactive Simulation (DIS) (IEEE 1278.1), Aggregate Level Simulation Protocol (ALSP) (Wilson 1994) and the High Level Architecture (HLA) (IEEE 1516.1). It places a few restrictions concerning the realization of individual simulators. This could create a common simulation environment, where the entire simulation is viewed as a black box and made up by many different simulation nodes. The standardized interfaces can be used to interconnect the simulation nodes within the black box to achieve the goal of concurrent processing. The HLA is the newest technical innovation which integrates the operation concepts of DIS and ALSP and other distributed simulation issues with standard protocols. This technical innovation has allowed a radical upward scaling of the number of the entities involved in the distributed simulation. Such scaling, combined with constant advances in computational horsepower, has opened up the promise of representing the interactions of hundreds of thousands of highly complex entities.

The HLA is a project initiated by the Department of Defense, USA, to support the interoperability among the distributed simulators. It has later become an international standard, IEEE 1516.1, in 2001. The HLA defines an infrastructure to support the reusability and interoperability among heterogeneous simulations. To achieve this goal, the HLA defines the Time Management services for the simulating system to coordinate its execution with others. The Time Management service defines the synchronization mechanism to ensure the attribute/event sequence among distributed simulation nodes. In HLA terminology, each distributed node of a simulation is called a federate. Furthermore, a federation is a simulation that consists of a set

of federates. Depending upon the simulating model, a federate can use the time-stepped or event-driven time advancing technique. For example, the analytical war game federate is a sequential flow of simulation. After the initial conditions are established, the simulation is executed in multiple iterations until the simulation result is converged to a stable range or value. To accomplish this objective, it is designed with the faster-than-real-time methodology to perform the simulation. Whereas the real-time operational war game system requires user intervention to perform the simulation, it uses the time-stepping mechanism to synchronize the wall clock. While the heterogeneous joint training simulation is a complex aspect of modern military operation, the HLA Time Management service defines a method to synchronize different types of federates within a joint federation.

Although the HLA specification enables time-stepped and event-driven federates to be joined in a federation, questions of why and how to let an analytical federation join a time-stepped federation are important research issues. The paper proposes an approach to use the analytical system as the decision support system for the joint operational training simulation. In the following sections, the classification of interoperability using the HLA approach is studied first. The technical issue of an analytical system jointly with a real-time operational federation is explored in the next section. The scenario and implementation of the proposed approach is the followed. The experiment result and discussion is given at the end.

2 METHODOLOGY OF JOINT SIMULATION OF HETEROGENEOUS SYSTEMS

The interoperability among the distributed simulations can be classified into four levels: Application Interoperability, Model Interoperability, Service Interoperability and Communication Interoperability (Myjak 1999), as shown in Figure 1.

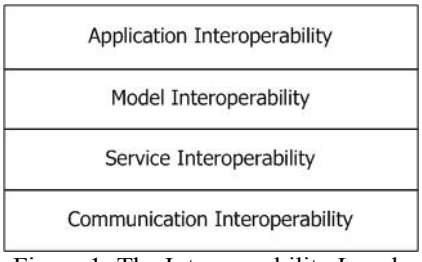

Figure 1: The Interoperability Levels

Here, the heterogeneous joint simulation is referred to the joint training simulation environment assembled by different types of simulation systems, including different levels of time-stepped simulators and event-driven simulating systems. We can categorize the heterogeneous joint simulations according to the levels of interoperability. The application level of interoperability for joint simulation exists between two different types of war gaming systems. The discrepancies may include different time advancing mechanisms employed, or different designed purposes with distinct functionalities. For example, the interoperability between the analytical and the real-time war gaming systems belongs to this category. The model interoperability level refers to the diversity of the object models employed in the federated simulating systems. The examples of this level may include two interoperating federations that use different FOMs or two federates with dissimilarity in simulated object resolutions. For example, the interoperability between the tank simulator and the battalion war game system has different resolution of entities. For the service level of interoperability, the simulating systems with different communication protocols are interconnected. For example, the interconnection between the DIS-based simulator and the HLA simulator belongs to this category. Finally, the communication interoperability level implies different communication protocols are employed to connect simulating systems. For example, different manufacturers build their RTI with distinct communication protocols, such as TCP and SCRAMNet, which can not interoperate directly.

The paper focuses on the issue of Application level of interoperability. Specifically, the study is concentrated on the interoperability of HLA specification of simulating systems. In the following subsections, methods of interconnecting HLA simulating systems are given first. Methods to connect an HLA simulating system with a non-HLA simulating system, such as a DIS simulator, then follow.

2.1 Heterogeneous environment with HLA simulation systems

The method to join the operation HLA federates and federation includes Federate Gateway, Federate Proxy, RTI broker, and RTI Interoperability protocol (Myjak 1999, Myjak and Sharp 1999). Their differences are depending upon whether they are in the Application level, Model level, Service level, or Communication level of interoperability.

2.1.1 Federate Gateway

Federate Gateway (Myjak and Sharp 1999) belongs to both the Application and the Model levels of interoperability. The purpose of this type of interconnection is to interoperate multiple federations into a federation community. This approach is to assign one federate from each federation as the representative of its federation. Representative federates from different federations then exchange messages through the Federate Gateway. Hence, the interoperability among federations is achieved through the representative federates and Federate Gateway.

The Security Guard Federate (Filsinger 1997) is an example of Federate Gateway. The main benefit of this approach is that it is capable of filtering or hiding important messages among federations to produce a secure simulation environment. With the Federate Gateway, we can construct the federation community in Intersecting, Adjacent, or Hierarchical architecture (Myjak 1999).

2.1.2 Federate Proxy

Federate Proxy is another technique for both Application level and Model level of interoperability. This approach allows a federate to join more than one federation to form a federation community. Hence, this federate is also called Federate Bridge or Bridging of Federation as suggested by SISO (Braudaway and Little 1997). This federate proxy acts like an agent among the interconnected federates. It helps a local federate to update its status to a remote object on a different federation to achieve interoperability. Through the federate proxy, federates in different federations will think all federates are in a unified federation. This approach can also be easily employed in the following cases of the federation community (Braudaway and Little 1997):

- Federations with different versions of RTIs;
- Federations with different time advancing techniques;
- Federations with different FOMs.

2.1.3 RTI Broker

RTI broker is similar to the Federate Proxy that lies between two federations. However, different from the Federate Proxy, RTI broker acts like a mediator that interchanges messages between two federations at the Service level. That is, it is designed to achieve the interoperability between the RTIs manufactured by different venders. Hence, it not only can interoperate federations but also can be used to harmonize different RTIs that are designed with different techniques. The main drawback of this approach is that it requires defining an interface between different RTIs.

2.1.4 RTI Interoperability Protocol

RTI broker is not an efficient approach since its interoperability is constrained by the differences of the programming languages, algorithms, and design principles among RTIs. The RTI Interoperability Protocol is a low-level approach to solve this issue. Compared to RTI Broker that interchanges messages through APIs of different RTIs, this approach allows RTIs to communicate their internal states as well as federate data.

2.2 Federating HLA simulation system with non-HLA simulation system

There are several methods to upgrade non-HLA systems to HLA-compliant simulation systems. When considering interoperating with HLA systems, Wood and Petty (1999) suggests those upgrade approaches can be classified into the following two categories:

- Reengineering the existing system so that its communication module is conformed to the HLA specification.
- Using an extra mechanism to convey messages between HLA and non-HLA simulation systems.

The first category is generally called an “Integration” approach and the other is called a “Gateway” approach. The Gateway approach is well-known in interoperating DIS simulators with HLA simulation systems, which was initiated by IST in 1996 (Wood and Petty 1999). With this approach, no modification is required on the non-HLA simulators and the Gateway is responsible for converting messages between two different protocols. When HLA Gateway receives a PDU from DIS simulators, it translates this message into corresponding RTI service to HLA federates. When HLA Gateway receives a callback from other federates, it then packs a PDU according to RPR-FOM(Real-time Platform Reference-Federation Object Model) and forwards this PDU to DIS simulators.

For the first category, it is often implemented either by the Native approach or by the Middleware approach. The Native approach redesigns the non-HLA system with the HLA specification, while the Middleware approach modifies the communication module of the existing system. The Native approach is the most straightforward method and can achieve efficient execution performance among all approaches yet costly. On the other hand, the Middleware approach is an equilibrium between the pervious two methods. The Middleware plays the role of an exclusive “Internal Gateway” (Paterson 2000) for the non-HLA system when it is interoperated with the HLA systems. Compared to the Native approach, Middleware costs less to implement but yet more efficient than the Gateway approach.

3 USING THE ANALYTICAL SYSTEM AS THE DECISION SUPPORT SYSTEM

Due to the property of the analytical system, once it is executed, the analytical war game system will not stop until the specified iterations are completed or a predefined condition is reached. Furthermore, hypothetic initial conditions must be given to the analytical war game system to perform the analysis. These assumptions may not be realistic enough and this limits their application to the real battlefield situation. For the real-time operational simulation,

status may change rapidly along with the ongoing engagement. Hence, for the analytical war game software to perform a realistic analysis, it must interactively receive the information of up-to-date status when the simulation is being performed. In this way, it can perform the course of action (COA) analysis during the simulation (Guleyuponglu and Ng 2001). However, since the analytical war game software and the real-time simulation are using different time advancing strategies, this approach may post the following questions:

- **Why**: Are there any specific benefits from interoperating the event-driven analytical war gaming system with the time-stepped real-time operational system?
- **When**: Since the analytical war gaming system and the real-time operational system were designed with different time advancing techniques for distinct purposes, should they interoperate at all time during the federation execution? If so, the event-driven federate has to be degraded into the time-stepped simulation in order to coordinate their time advancing techniques. Or, can the analytical war gaming system be triggered by the real-time simulation system as needed? If so, when?
- **How**: How to interconnect these two diverse war gaming simulation systems? What is the architecture to construct such a heterogeneous environment?

3.1 Purpose of Interoperability

The analytical and the real-time war gaming systems are two distinct simulation systems. Their differences range from the designed architecture, the time advancing mechanism, to the application domains. Hence, issues of the interaction between these two dissimilar types of simulating systems not only include technical problems but also the purpose of interoperability.

From the user's perspective, the analytical war gaming system often derives more accurate simulating result than the real-time one. In order to keep up with the timing constraint, the time-stepped real-time simulation system often uses fast but less accurate models to compute the object behavior as well as the combat assessment during the simulation. On the other hand, since the analytical war gaming system uses the event-driven method to execute the simulation as fast as possible, its simulation models are often more precise and the outcome is more convincing. Furthermore, since the real-time war gaming system is a tactical operational training system, its simulation is often dynamic with the plot and the simulated objects are often more than hundreds. On the contrary, the scenario for the analytical system is often restricted to a specific topic to analyze the effect of certain situations. Due to these innate differences, this paper categorizes the purposes of interoperation between the analytical system and real-time simulation system as follows:

1. **Using the analytical system to boost the simulation accuracy of the real-time system.** Since the real-time war game system aims to train the tactical skill of the user, the accuracy of the simulated model is not the subject. The user is only concerned about the result of the simulation. However, for certain scenarios, the precision of the model may profoundly affect the result of training. Hence, we can use the outcome of the analytical system to boost the accuracy of the simulating model in the real-time war game system. For example, the detonation effect in the real-time war game system is often modeled by a random number while the analytical system can give a more faithful assessment.
2. **Using the analytical system as the decision support system of the real-time simulation system.** During the real-time simulation, the trainer often faces the dilemma of making the "right' decision to derive a satisfying result. Based upon the time advancing mechanism of the real-time war gaming system, the trainer often has to make a decision on how to deal with an event in a short period of time with limited information for that particular event. Hence, within that short period of time, we can use the analytical war gaming system to analyze all possible actions to deduce a reasonable decision.

This paper focuses on the second purpose of interoperability. That is, this paper attempts to study the required technique and mechanism to interoperate the analytical and real-time simulation systems, so that the analytical war gaming system can be the decision support system of the real-time operational training system.

3.2 Interaction

After have concluded that the purpose of interoperability is to use the analytical war gaming system as the decision support system, the next question is when to launch the analytical system during the operational simulation? In addition, how much time does the analytical system can have to analyze the user's decision? If the analytical war gaming system uses too much time to analyze, the result may be too late to be useful to the real-time simulation system. Hence, the time to launch the analytical system, says t_1, and time for it to return the analysis result, says t_2, are two crucial factors. In other words, the interaction between the analytical system and the real-time system only take place

at the time t_1 when the real-time system launches the analytical system and the time t_2 when the analytical system returns the analytical result. If the interval $[t_1, t_2]$ is smaller than the time for the real-time system to launch the analytical system and the analytical system to return the analytical result, the analytic system does not have the time to perform any analysis.

To answer these questions, the following terms in the timeline are defined as follows. Define T_L and T_R as the time required for the real-time system to pass a command to launch the analytical system and the analytical system to return the analysis result to the real-time system, respectively. Furthermore, let T_A be the average simulation time of the analytical system per iteration. Hence, the minimum time for the analytical system to perform decision support is given by:

$$T_{min} = T_L + T_R + n*T_A \quad (1)$$

Where n is the minimum number of iterations for the analytical system to deduce an acceptable simulation result. Let t_s be the time when a specific event e which requires the user's attention is first spotted and t_d is the latest time that the user must decide how to deal with that event e. Hence, the interval $[t_s, t_d]$ is the time that the user can have to decide his move. Notice that time t_d is the function of event e, says $t_d = f(e)$, which depends upon the application domain. If $|t_d - t_s| > T_{min}$, then the user can use the analytical system to perform the COA analysis. Let t_l, $t_l \in [t_s, t_d]$, be the time when the user triggers the analytical system, the interval $[t_l, t_d]$ is the maximum time that the analytical war gaming system can have to analyze the user's intention. Notice that we must have $|t_d - t_l| > T_{min}$ for the analytical system to perform a convincing simulation. Hence, the real-time war game system sends an event to the analytical war game system at time t_l and the analytical war game system should reply the assessment before the deadline t_d.

Notice that, if the interval $[t_s, t_d]$ is much larger than T_{min}, i.e $|t_d - t_s| >> T_{min}$, the system may allow the user to perform multiple assessments by dividing this interval into several sub-intervals. The user may launch the analytical system within any of these sub-intervals and demand the analysis to be completed before that sub-interval expires.

3.3 Joint Simulation Using HLA

The design issue of the architecture for the heterogeneously joint simulation of the real-time war game system and the analytical system is to synchronize the timing policy among them. Due to the differences in the time advancing policy between the analytical and real-time systems, they must execute asynchronously to prevent the analytical one to scale down as a time-stepped simulation. On account of this constraint, the key issue of the joint simulation of real-time and analytical systems is the coordination of these two diverse time policies.

To answer this question, we need to further explore the essences of these two systems. For the time advancing technique, the real-time system is based upon the scaled time step of the simulation while the analytical system is decided by the next executable event. Furthermore, the result of the real-time simulation is manipulated by the user interaction and the analytical system is dominated by the statistic model with given parameters. Hence, in order to interoperate these two systems without interfering their innate features, the execution of the analytical system must be regulated by the real-time system. According to the HLA specification, the Time Management service of the real-time system must be set as a time-regulating federate while the analytical system a time-constrained federate. Hence, during the simulation, the real-time simulation is executed at its own pace and the analytic system is regulated by the real-time simulation. When an event occurs and the user decides to launch the analytical system, an interaction with time stamp t_d is then sent to the analytical system. After receiving this interaction event, the analytical system can use Equation (1) to decide the number of iterations for the analysis. Notice that the analytical system must reply the simulation result to the real-time system before td and followed by a NextEventRequest() function call with time stamp t_d.

The level of interoperability of this type of joint simulation belongs to the application level or the model level. According to Section 2, the federate gateway or federate proxy is a feasible approach to join them to a single federation. Moreover, since the legacy analytical system was often designed as a standalone simulating system, when joining the analytical system with the HLA federates, a middleware is often designed for it. Since the real-time system for our study is a proprietary theater-level operational training environment, a gateway federate is designed on its side as illustrated in Figure 2.

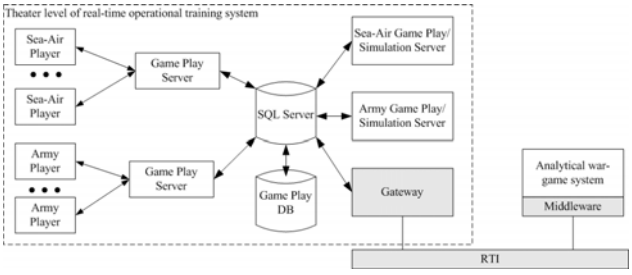

Figure 2: The Architecture of the Heterogeneous Joint Simulation Environment

The gateway federate is synchronized with the SQL server, accesses the data from the SQL server, and send the data directly to the analytical system. Before the gateway federate sends an interaction event to trigger the analytical

simulation, the analytical system only passively receives the up-to-date status of the operational training system. In this way, the overhead for the analytical system to acquire the latest information of the real-time war game system is minimized.

4 SCENARIO AND IMPLEMENTATION

Since the time t_d is application-dependent, a scenario is designed to further explore the issue of joint simulation of the analytical system and the real-time simulation. To simplify our study, the scenario is set as a cruise missile attack on a military base and the base tries to intercept this attack by antiaircraft missile and artillery. So, the user has to decide how many antiaircraft missiles should be used to intercept this cruise missile attack. The timing relation of this scenario can be depicted as a distance-time diagram as shown in Figure 3.

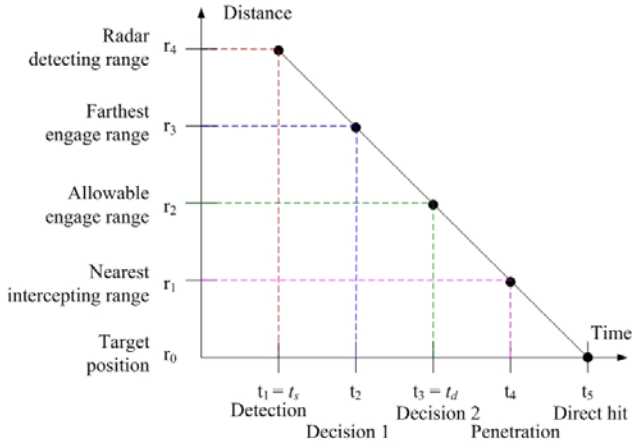

Figure 3: Distance-Time Relation of the Scenario

According to Section 3.2, the time t_1 in Figure 3 is the spotting time t_s of this attack event. Furthermore, we assume the cruise missile will hit the base at time t_5 if the interception fails. Based upon the speed of the cruise missile and the antiaircraft missile, we can assume the nearest intercepting range is r_1 and the penetration time t_4. Counting the time required to lock and launch the antiaircraft missile, the allowable engage range is r_2 which gives us the latest decision time t_3. Notice that t_3 is the time t_d in Section 3.2. That is, the time interval $[t_1, t_3]$ is the maximum time that we can use the analytical system to assess the user's intention. Since the antiaircraft missile has the maximum range of trajectory, we can further assume that the farthest engage range is r_3 and the corresponding time is t_2. Hence, we can set the time t_2 as the first decision time and t_3 as the second decision time that the user can decide how to intercept the attack. In other words, $[t_1, t_2]$ and $[t_2, t_3]$ are two possible time intervals that the user can activate the analytical system to assess his intercepting decision.

Finally, based upon the scenario, the object hierarchy for the FOM can be depicted as shown in Figure 4 and the interaction hierarchy in Figure 5.

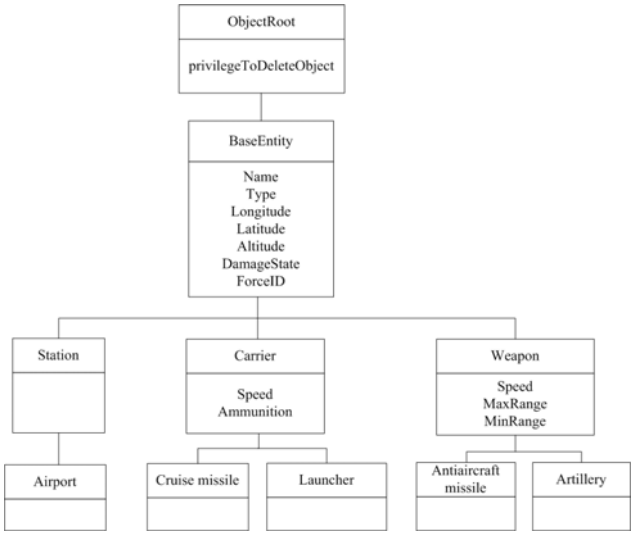

Figure 4: The Object Hierarchy of FOM

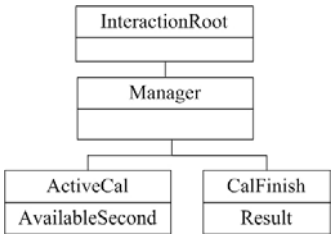

Figure 5: The Interaction Hierarchy of FOM

5 THE EXPERIMENT

Since the real-time operational war gaming system in our experiment was built with the server/client architecture, as shown in Figure 2, a gateway federate is implemented on a Celeron 2.4GHz computer with 512 MB memory. On the other hand, the analytical software and its middleware run on a Celeron 2.4GHz computer with 512 MB memory. The RTI version used in our experiment is DMSO RTI 1.3 NG V6. For the Time Management service, the gateway federate for the real-time war gaming system is set as a time-stepped federate with each time step equal to 5 time units. Moreover, it accesses data from the SQL server every 10 time units. Notice that, due to the feature of the real-time war gaming system, one time unit is equal to one second in our experiment.

There are three factors that will affect the performance of using the analytical system as the decision support system for the real-time war gaming environment. The first one is the time required for the gateway federate to send

the initial setup information to the analytical system. Another factor is the performance of the gateway federate to interact with the middleware to launch the analytical system. Notice that the first and the second factors constitute the time T_L in Equation (1). The other factor is the performance of the middleware and it is corresponding to the time T_R in Equation (1).

The first factor is concerned about the time required for the analytical system to receive the parameters for it to perform the analysis. If this reception is done right after the user has launched the analytical system, the receiving time may consume too much available time. That is, T_L in Equation (1) may become a significant element which leaves little time for $n*T_A$. To solve this problem, we allow the middleware to request the up-to-date information from the gateway federate regularly. Since the middleware is an event-driven federate, it will issue NextEventRequest() every 30 time units before the analytical system is activated. In this way, it can not only receive timely information of the real-time system but also synchronize with the gateway federate every 30 time units. Hence, with this design, the performance of the proposed approach is reduced to the performance of the gateway federate and the middleware.

To evaluate the performance of the gateway federate and the middleware, experiments were conducted to estimate their overhead. Our experiments show that the gateway federate took around 0.015 to 0.0020 second per frame to access the SQL database and issue updateAttributeValues() function call per frame. For the middleware, it took around 0.0015 to 0.0017 second per frame to receive the update, write the data into the file, and issue NextEventRequest(). Since the time step for the gateway federate is 5 seconds, our experiments show that the gateway federate and middleware will not cause any significant overhead to the entire simulation.

6 CONCLUSION

This paper presents an approach to join the analytical war game system with the real-time federation. The purpose of this joint simulation is to study the method of using the analytical system as the decision support system during the real-time simulation. The infrastructure and the Time Management service for such joint simulation are also fully discussed. The result of our research shows that this approach can effectively improve the tactical skill of the trainer since he can evaluate his intention online before actually making his move.

The only problem of this approach is the calculation of the decision deadline t_d which is application dependent. Several application-domain related parameters have to be considered when computing t_d. However, since those parameters can be pre-estimated, the online calculation of t_d during the simulation is still possible. Although it is application-domain specific, further study of dynamical computation of t_d is an interesting research topic.

Our research shows a promising approach of adopting the analytical war game system to a threat-level real-time war gaming system. Restricted by the available systems, this study focuses on a proprietary client/server real-time system and a stand alone analytical system. Further extension of this approach to other HLA-based real-time operational simulation systems and analytical systems is currently under investigation.

ACKNOWLEDGEMENT

This research is sponsored by Joint Training Simulation Center, Minster of National Defense, Taiwan.

REFERENCES

Bhatt, S., et al. 1998. Parallel Simulation Techniques for Large-Scale Networks. *IEEE Communications*, 36(8): 42-47.

Braudaway, W., and R. Little. 1997. The High Level Architecture's Bridge Federate. In *Proceedings of the Fall Simulation Interoperability Workshop*, paper no. 97F-SIW-078. Orlando Florida, U.S.A.

Das, S., R. Fujimoto, K. Panesar, D. Allison, and M. Hybinette. 1994. GTW: A Time Warp System for Shared Memory Multiprocessors. In *Proceedings of the 1994 Winter Simulation Conference*, 1332-1339. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Ferenci, S. L., K. S. Perumalla, and R. M. Fujimoto. 2000. An Approach for Federating Parallel Simulators. In *Proceeding of the 14th Workshop on Parallel and Distributed Simulation*, 63-70. Bologna, Italy.

Filsinger, J. 1997. HLA Security Guard Federate. In *Proceedings of the Spring Simulation Interoperability Workshop*, 1015-1023, paper no. 97S-SIW-163. Orlando Florida, U.S.A.

Guleyuponglu, S., and H. Ng. 2001. Linking Real-Time and Faster-Than-Real-Time Federations. In *Proceedings of the Spring Simulation Interoperability Workshop*, paper no. 01S-SIW-131. Orlando Florida, U.S.A.

IEEE Std 1278.1. 1995. IEEE Standard for Distributed Interactive Simulation – Application Protocols. New York, NY: Institute of Electrical and Electronics Engineers, Inc.

IEEE Std. 1516.1. 2000. IEEE Standard for Modeling and Simulation (M & S), High Level Architecture (HLA) - Federate Interface Specification, i-467.

Myjak, M.D., D. Clark and T. Lake. 1999. RTI Interoperability Study Group Final Report. In *Proceedings of the 1999 Fall Simulation Interoperability Workshop*, paper no. 99F-SIW-001. Orlando Florida, U.S.A.

Myjak, M. D. and S. T. Sharp. 1999. Implementations of Hierarchical Federations. In *Proceedings of the Fall Simulation Interoperability Workshop*, paper no.99F-SIW-180. Orlando Florida, U.S.A.

Paterson, D. J., E. S. Hougland, and J. J. Sanmiguel. 2000. A gateway/middleware HLA implementation and the extra services that can be provided to the simulation. In *Fall Simulation Interoperability Workshop*, paper no. 00F-SIW-007. Orlando Florida, U.S.A.

Steinman, J. S. 1992. SPEEDES: A Multiple Synchronization Environment for Parallel Discrete Event Simulation. *International Journal on Computer Simulation*, 2(3): 251-286.

Xiao, Z., B. Unger, R. Simmonds, and J. Cleary. 1999. Scheduling Critical Channels in Conservative Parallel Discrete Event Simulation. In *Proceedings of the Workshop on Parallel and Distributed Simulation*, 20-28.

Wilson, A. L. and R. M. Weatherly. 1994. The Aggregate Level Simulation Protocol: An Evolving System. In *Proceedings of the Winter Simulation Conference,* 781-787. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Wood, D.D., and M. D. Petty. 1999. HLA Gateway 1999. In *Proceedings of the Spring Simulation Interoperability Workshop*, paper no. 99S-SIW-051. Orlando Florida, U.S.A.

AUTHOR BIOGRAPHIES

JIUNG-YAO HUANG is Associate Professor of Department of Computer Science and Information Engineering at National Taipei University, Taiwan. His research interests include pervasive computing, computer graphics, networked virtual reality, distributed system and multimedia system. He has been a member of IEEE and ACM since 1990 and joined the Society for Computer Simulation in 1996. His e-mail address is <jyhuang@mail.ntpu.edu.tw> and his Web address is <http://web.ntpu.edu.tw/~jyhuang/>.

MING-CHIH TUNG is Assistant Professor of Department of Computer Science and Information Engineering at Ching Yun University, Taiwan. His research interests include modeling and simulation system, distributed simulation and networked virtual environment. His e-mail address is <mctung@gmail.com>.

JI-JEN WU is Navy Captain of Taiwan. He is also a Ph.D. candidate of Department of Computer Science and Information Engineering at Tam-Kang University, Taiwan.

SHU-SHEN WAI is Navy Captain of Taiwan. He is also a Ph.D. candidate of Department of Computer Science and Information Engineering at Tam-Kang University, Taiwan.

CHUNG-HSIEN TSAI is Air Force Major of Taiwan. He is also a Ph.D. candidate in Computer Science Information Engineering National Central University.