Integrated Middleware for Flexible DIS and HLA Interoperability

Lawrence A. Rieger
US Army ARCIC
Ft. Monroe VA
lawrence.rieger@us.army.mil

Jennifer Lewis
SAIC
Huntsville, AL
jennifer.e.lewis@saic.com

ABSTRACT

The Army Battle Lab Collaborative Simulation Environment (BLCSE) operates a distributed laboratory network in support of Army Concept Development Experimentation Plan (ACDEP). While operating in an IEEE 1278 (DIS) technical mode, the BLCSE was required to become compliant to and interoperable with IEEE 1516 (HLA) federations, yet retain the ability to operate in DIS mode during the process, which would overlap two major experiments. Although use of DIS-HLA Gateways, on either a by-machine or by-location basis, was evaluated as feasible, cost, complexity, and latency concerns forced an embedded Middleware solution. This paper discusses the operating environment concerns that precluded the traditional Gateway interoperability, the development and integration of Middleware integrated into compiled executable software programs, and the management process that enabled the Middleware to maintain DIS functionalities while the disparate federation simulations and tools were enabled compliant to the HLA. Lessons learned on Middleware integration of DIS-HLA federates while making the system interoperable between two major operating environments are provided. Project development began in late FY05, and all federates will complete Middleware integration prior to the end of FY06.

ABOUT THE AUTHORS

Lawrence A. Rieger is Technical Configuration Controller for the TRADOC BLCSE, working in Simulations Division, Army Capabilities Integration Center. He received a BA from Belmont Abbey College in 1976 and an MS from Troy State University in 1982. He is also a graduate of the Army Command and General Staff College and the Army Management Staff College. Following active and reserve commissioned service with both light and mechanized forces, he has spent the last 21 years in the development and management of Army simulations, working in live, virtual, and constructive environments.

Jennifer Lewis is a software engineer developing technical integration solutions for TRADOC's Battle Lab Collaborative Simulation Environment. She holds a BS and a MS in Computer Science with an emphasis in Telecommunications and Networking from the University of Texas at Dallas. She has worked in the design and implementation of networking protocols for the telecommunications and defense industries for the past six years.

Integrated Middleware for Flexible DIS and HLA Interoperability

Lawrence A. Rieger
US Army ARCIC
Ft. Monroe VA
lawrence.rieger@us.army.mil

Jennifer Lewis
SAIC
Huntsville, AL
jennifer.e.lewis@saic.com

INTRODUCTION

This paper discusses the management and technical processes and changes in transitioning the TRADOC Battle Lab Collaborative Simulation Environment (BLCSE) from an IEEE 1278 (DIS) technical baseline to an IEEE 1516 (HLA) baseline, while still maintaining a full schedule of critical Army experiments.

The BLCSE operating environment

BLCSE consists of 18 TRADOC battle labs and supporting sites spread across the US, linked together within a virtual classified network hosted by the Defense Research and Engineering Network (DREN). Most sites use a DS3 (Digital Signal Level 3 – 45 Mbps/T3) interface. Selected sites use an OC3 (Optical Carrier 3 – 155.52 Mbps) interface, and one site uses an OC12 (622 Mbps) interface. Growing out of the earlier SIMNET-based TRADOC Core DIS Facilities (CDF's) of the 1990s, the BLCSE was enlarged and expanded with the mission of providing a continuous simulation and analysis capability in support of the Army Advanced Concepts Development and Experimentation Program (ACDEP).

BLCSE maintains two separate simulation federations, Brigade and Below (B2F) and Corps and Division, although when this HLA effort began, only the B2F was actually operating. All federates within B2F are entity level, and many are based on the Objective Force OneSAF Test Bed (OFOTB) and derivatives. The federation operates using IEEE 1278-1998, along with a number of BLCSE-developed expedient Protocol Data Units (PDUs) written to fill gaps between the IEEE 1278 standard and ACDEP needs.

BLCSE normally conducts two to four major experiments a year, with several mini-experiments conducted either over the DREN or by the various battle labs themselves. One of the great challenges of the effort was to develop, integrate, and test Middleware software at the various BLCSE locations without interfering with the ACDEP schedule and working around the availability of federates and personnel at the various locations.

The 3CE interoperability requirement

In March 2003, the Army Modeling and Simulation Executive Council formally recognized the need for a common operating environment to allow cohesive design, development, integration and testing of the Army's capabilities, systems and prototypes. As a result, representatives from three Army commands and the Future Combat Systems (FCS) Lead Systems Integrator (LSI) formed the Cross Command Collaboration Effort (3CE). (Altan, 2006) The 3CE brought together four disparate simulations environments: TRADOC BLCSE, RDECOM Modeling Architecture for Technology and Research Experimentation (MATREX), ATEC Test and Training Enabling Architecture (TENA), and the PM FCS (LSI) FCS Simulation Environment (FSE). The 3CE end state is a 3CE environment that meets the common requirements of all three commands and PM FCS BCT to conduct distributed development. (3CE, 2006) The principal way to achieve this end state is to identify, develop, and maintain a core set of M&S tools, data, and business processes that provide interoperable connectivity.

The initial intent of 3CE was to ensure that Army experimentation occurring within the BLCSE can exchange data, or conduct cooperative testing, with events occurring within the other Army experimentation or instrumentation facilities and networks. RDECOM brought into the 3CE program an assembly of simulations and tools, much as BLCSE did, specifically including a 1.3NG Run Time Infrastructure (RTI), which was adopted as the initial RTI by the BLCSE. This RTI is based on the DMSO 1.3NG RTI version 4.2.

CONSIDERATIONS FOR DECISION-MAKING

In determining the most effective means to HLA interoperability with the 3CE, we looked at three potential solutions. These were using DIS to HLA Gateways, installing Middleware within the various federates, or re-architecting the existing federates to become HLA compliant. We also had to take into consideration how the BLCSE would be interoperable with the 3CE.

Gateway considerations

In considering the Gateway option, each Gateway would be a two-way conversion mechanism. It would convert DIS PDUs into entity interactions and attribute changes and convert those interactions and changes back into DIS PDUs. Gateways were generally available, both as GOTS and COTS, with their cost and functionality being the major differences. Two Gateway approaches were reasonably available. The first, and simplest, was to maintain the existing LANs within the various BLCSE sites and have a single Gateway between that LAN and the WAN utilized for the experiment. As a subset of this approach, we also examined the potential to have a series of mini-LANs at our major facility, which typically hosts over 600 computer simulation hosts, both virtual and constructive. The second approach was to place a Gateway between each simulation and the LAN, allowing each LAN in the experiment, as well as the WAN/LAN, to operate as the RTI backbone in HLA mode.

The first course was explored at length, but two considerations finally caused the team to drop this approach: the fear of latency and the risk of dropped packets given the size of the customary experiment population. The second approach, individual Gateways, was very quickly dropped from consideration due to excessive cost.

Included in the consideration of the Gateway approach was the ability to continually "tune" the Gateways for expedient PDUs developed to meet experimentation needs, as well as the ability to keep Gateways updated based on regularly changing BCLSE FOM versions.

Middleware considerations

The Middleware course of action offered two particular advantages in operating efficiency and utility. The first advantage was the ability to reduce latency, always a major concern in large distributed events. The second was savings in BLCSE operating hardware, since there was no requirement for a separate Gateway host. The Middleware application also required far fewer CPU cycles than an embedded Gateway to operate on the simulation host, which is commonly already being used to its computational capacity by simulation demands. The decision was made to develop an optimized Middleware application to be linked into the compiled executable of each DIS federate. By deliberate decision, a sub-contractor was brought in to help produce the Middleware, which became government proprietary. Handed off to the development team, the Middleware has demonstrated its ability to be updated by simple patch and to be flexible enough to be adapted to new PDUs and FOM changes.

Re-architecture considerations

Although considered, converting all existing federates to native HLA standard was not a viable course of action. There were 12 primary simulations within the BLCSE federation plus several servers and tools. The author's experience in one re-architecture program, a relatively simple code rewrite, cost over a million in FY99 dollars. (Rieger, 1999) Two other factors against re-architecture were the already-intended replacement of OFOTB, a DIS-based SAF, with HLA-based OOS during FY07 and the deadline to be HLA compatible by the end of FY06.

BLCSE-3CE interoperability considerations

At the center of the team's initial attempts for interoperability was the selection of certain common tools, primarily the MATREX 1.3NG RTI, the MATREX FOM, and certain tools for integration, notably HLAResults™.

An initial issue was the difference between the MATREX FOM, based on the SIW Engineering Proto-Federation, and BLCSE, based on a real-time platform DIS baseline. After discussion, the development team created an initial BLCSE FOM, based on the MATREX FOM but modified to include a series of expedient PDUs, which meet experimentation needs by the Unit of Action Maneuver Battle Lab (UAMBL) at Ft. Knox, the BLCSE primary battle lab and systems integrator.

Although all Middleware development and integration testing was accomplished using the MATREX RTI with the BLCSE FOM, the objective was a common FOM with the 3CE. Early in 2006, coordination began between BLCSE and 3CE to develop a common FOM. These efforts continue in a formal working group, with an expected objective common FOM completion date in August 2006.

Development in an interim environment

A guiding factor in the overall transition process was the intent to move BLCSE to an objective HLA, full IEEE 1516, in the relatively near future. This meant the Middleware software needed to function in both HLA environments and allow maximum flexibility in the interim. Related to this was the need to design the Middleware and the FOM for significant changes, both as major federates were replaced and the FOM was changed to reflect the migration to the 3CE FOM.

MIDDLEWARE DEVELOPMENT

With the decision to use Middleware in an interim 1.3 NG MATREX environment, the development team went to work, dividing the development process into three major phases with specific deliverables for each phase (Jones, 2005). The following sections discuss each of the major deliverables.

BLCSE interface control documents

Before any work could begin, the team needed to document the current state of the federation. The team created interface control documents (ICD) for each BLCSE federate and for the federation as a whole. The ICDs were the first consistent set of documents to capture the BLCSE federation at a single point in time. They provided a detailed description of the purpose and structure of each PDU used in the BLCSE federation, including valid data types and enumerations used within the PDUs. They also clarified which federates populate and process which PDUs. Not only did these documents allow the development team to formalize and verify their knowledge of the existing federation, they also provided immediate results to the customer and lent credibility to the work still to be completed.

BLCSE federation object model

In conjunction with the ICDs, the team created a BLCSE Federation Object Model (FOM) and related Simulation Object Models (SOMs). Knowing the 3CE federation would use the MATREX FOM, the team decided to use it as a baseline for the BLCSE FOM. The development team compared the data in the MATREX FOM to the data being transmitted in each of the 61 BLCSE PDUs. In many cases, the MATREX FOM used identical or similar data to represent battlefield, C2 and other conditions.

The team also produced a detailed mapping document showing federate developers how to find the information from each data field of the 61 BLCSE PDUs in the new BLCSE FOM. The development team used this document extensively during a series of five weekly review sessions, which were conducted to obtain the community's final approval of the BLCSE FOM. During these sessions, a member of the development team walked through a pre-determined portion of the FOM, pointing out potential issues for consideration and action. Each Monday, a new version of the FOM and mapping document, containing updates from the previous week's meeting, was distributed along with a list of objects and interactions to be discussed at the next meeting. The detail provided in the mapping document, and the systematic approach to the approval process, contributed extensively to the confidence the community had in the development team. This, in turn, greatly increased the support the community gave to the transition process as a whole.

BLCSE Middleware software library

The Middleware software is a multi-layer library, which is linked into each federate application via shared objects or dynamic link libraries. The architecture of the library is extremely flexible and provides incremental incorporation of legacy simulation clients. Multiple plug-ins make it platform independent and adaptable to any communications library, including HLA 1.3, IEEE 1516, a proprietary protocol, or any future communications architecture. The two main conceptual layers of the library are the traditional Middleware, which abstracts and simplifies the RTI application program interface (API), and the DIS Adapter, which acts as a front-end to the traditional Middleware for DIS-centric applications.

Because 90% of the BLCSE federates are DIS-centric, the design of the DIS Adapter API was critical to success of the entire project. This API had to be easy to understand and quick to implement. As such, the API adheres to the design of the traditional socket APIs. Each routine in the DIS Adapter API is designed to mimic the behavior of the original socket call. For example, MwaConnect() mimics socket(). It opens a connection to the DIS Adapter and returns a Middleware adapter handle, which is used in all future calls to the API. MwaSendPDU() mimics send(). It transmits data across the HLA network and returns the number of bytes sent across the network.

Wherever possible, identical arguments, behaviors and return values were used to simplify the integration process. One example where identical behaviors could not be implemented was in MwaReceivePDU(), which

mimics recv(). recv() uses the input socket handle to determine whether the routine should block if no data is immediately available on the socket. Since MwaReceivePDU() must use the handle returned from MwaConnect(), it cannot determine whether the user expects the socket to block. By default, the routine will not block, forcing the user to use MwaSelect(), which mimics select(), to obtain the same behavior as a single call to recv(). Such differences in the APIs are minor and have caused no significant delays in integrating the Middleware libraries with the federate application.

THE INTEGRATION PROCESS

The key to the entire integration process was scheduling. Middleware testing had to blend into the already overwhelming experimentation schedule BLCSE maintains. The planning and execution of some BLCSE experiments require months of more-than-full-time work for federate developers. This forced the development team to schedule many months in advance of any integration trip to ensure no last minute technical questions arose to jeopardize the schedule.

In light of these issues, the team chose to conduct a series of spiral test events. This allowed integration and testing to begin almost one year earlier than would have been possible under a strict waterfall development lifecycle. Not only did these events provide early feedback, they also greatly lengthened the development schedule, giving the small development team adequate time to complete the work. During the first test event, approximately 10% of the development was complete. However, this event gave an important proof of concept to the team and to the customer. It proved the DIS Adapter easily integrated into the federate application, basic functionality was available, and the performance was in-line with other available HLA applications.

Middleware preparations

Although the team designed the Middleware to limit impact to the federate application, source code changes are necessary for its implementation. Depending on the complexity of the federate source code and the skill of the developer making the changes, these changes have required one to 24 man-hours to complete.

In addition, the federate developers must prepare themselves for integration testing. The team decided that testing individual PDUs, rather than operational threads, was the best approach to fully testing the system. If the data sent in the PDU were the same data received in a PDU after crossing the RTI, the change in network communication would be transparent to the federate application. This decision allowed the team to focus on network and communication issues, rather than having to become deeply involved in the operation of the federation.

The choice did create some unexpected work for the federate developers, however. In many cases, the developer was intimately familiar with only a few of the PDUs the federate application used. Many others were rarely used or rarely modified, so it took some time for the developer to reacquaint himself with those PDUs. Before the team arrived, they asked the developer to know exactly which PDUs the federate application sends and how to stimulate the federate to send it. For example, the developer might need to know which commands to give to cause an entity to lase another entity. The developer also needed to know which PDUs the federate application receives and how the application should respond to receiving it. For example, when an Entity State PDU is received, the application should display a specific graphic on the GUI.

To ensure there were no complications caused by testing multiple federates at once, the team first conducted black box testing with each federate. That meant there was no other BLCSE federate application available to stimulate responses in the federate under test. In many cases, the only way to stimulate the application to send a PDU is to first receive a particular PDU. While the team created several test applications to accommodate this case, the federate developer needed to know what constitutes valid data for the incoming PDUs for the scenario under test. For example, many applications filter data based on location. Therefore, the data being sent by the test application may need a valid location for the specific scenario before the federate under test will process it. Understanding what needed to be tested and what constituted valid inputs into the application was the single most critical part of preparing for Middleware integration.

Technical integration

Once all preparations were completed, the test events provided invaluable feedback to the team. Most often, these events showed the ever-changing face of the BLCSE federation. During the 15-month development process, approximately 25% of the PDUs in use changed format and nine new PDUs were introduced. Because BLCSE analyzes future combat systems, the DIS entity type enumeration values changed between

almost every experiment. And with only a few months left in the development process, the communications effects model changed, introducing new operating procedures to both the situational awareness and the communication architectures. These issues were dealt with on a case-by-case basis, but because the team learned about the changes early on, none of the issues caused delays to the schedule.

The test events also gave continual feedback concerning the performance of the software. Initially, reports were good, and the team found the Middleware software introduced a minimal level of overhead to the federation. However, as testing progressed, several of the more complex federates reported unacceptable levels of network latency and processing time. This early feedback allowed the development team to make significant improvements in these areas in a timely manner.

The most significant area of improvement was in the use of the RTI. Minor modifications to the RTI Initialization Data (RID) file resulted in faster return times for object creates, object updates and interaction sends. In addition, the team introduced a number of filters into the software to prevent unnecessary calls to the RTI. Each DIS-centric federate sends heartbeat data for each of its entities. Because major BLCSE events can include up to 120,000 entities, half of which are static, filtering heartbeat data on the send side greatly decreases network latency. This decrease in latency becomes more significant as more static entities are introduced to the scenario due to increased detail in urban terrain databases. In addition, the use of static variables to represent attribute and parameter names and handles prevented the Middleware from needing to interrogate the local RTI client (LRC) during the creation of each new attribute instance. This technique produced measurable performance improvements, which varied per object and interaction, but averaged about 30%-40% increase in the number of updateAttribute and sendInteraction calls that could be processed in a given time.

LESSONS LEARNED

RTI environment

Possibly the most important lesson learned during the development process was the importance of a thorough understanding of the RID file parameters and communications architecture. Integration testing showed a significant delay in object creates within the Middleware. However, the problem was later determined to be a single parameter, LockType, in the RID file the federate was using. Changing this single value from its default value of StandardMutex to FifoMutex increased the object creation rate seven times. The team added additional performance improvements during the development process, but by the end, the performance of the Middleware and RTI was still inadequate to support the large experiments conducted by the BLCSE federation. Again, changing a single RID parameter value, StrategyToUse, from ClassPartioned to Simple made the difference between unacceptable and good performance. This change increased object update and interaction send rates by up to 95%.

Separate from the RID file issues, the RTI communications architecture, itself, has not been rigorously stressed in the typical Army experimentation environment. Although many HLA-based experiments have been run, the largest being the JFCOM-sponsored Urban Resolve series, those have not fully implemented either the reliable data transport typical of objective 1516 standards, nor provided the range of services inherent in the objective standard. (Helfinstine, 2005) For that reason, the development team planned from early inception to advance the RTI from a 1.3NG version (the MATREX RTI) to an objective 1516 version. Although initial point-to-point evaluations have been very successful, these tests have not provided the number of entities and interactions expected in large scale distributed experiments. For that reason, a major stress test has been scheduled in the BLCSE system for August 2006. This will test both the 1.3NG and the objective 1516 RTI. Following that stress test, the BLCSE will operate a significant sized force under HLA Middleware during the March 2007 experiment, shadowing DIS forces, and permitting a thorough comparison between the RTIs and DIS-based performance.

Middleware development

The most significant issue the team faced during software development was manpower. The multi-layer, flexible design of the software required at least four C++ classes to be developed for each of the objects and interactions used in the FOM. In addition, supporting classes for PDUs, enumeration mappings and complex data types, resulted in approximately 12 MB of source code. This was a huge amount of work for the two developers working on this code at any given time during the six-month work window. The lack of manpower, combined with integration scheduling and

other issues, eventually resulted in a three-week schedule slip from the initial integration timeline.

Because the source code tended to follow specific patterns based on data from the FOM, the team discussed creating Perl scripts to generate some of the files required by the Middleware library. However, due to scheduling and manpower issues at the beginning of the development period, the team decided there was not enough time to dedicate to this task. Three months later, it became obvious that these scripts would create an invaluable tool. Even if the generated code was not ready to compile, the scripts could generate a good framework for each class, saving days of work per C++ class. At that point, some basic scripts were developed, which eased the development team's workload over the balance of the integration period.

Integration issues

The biggest issue the development team ran into during integration was the issue of proprietary software as part of the various BLCSE federates. Also encountered was reluctance by several government proponents to release source code sufficient for the development team to map the Middleware and libraries into the host software. Eventually, two federates were not fitted with Middleware due to commercial vendors not releasing software code or requesting excessive fees to implement the Middleware. In the first case, the federate was simply dropped from use, and a different simulation was used in its stead. In the latter case, we developed a workaround using a link to a mirror simulation which did have the Middleware implemented.

Planning for evolution

As the HLA transition project was begun, it was clearly understood that the initial effort, achieving HLA compliance by the end of FY 06, was only an interim step. Once the Middleware was developed and installed, three major evolutions were expected. These are movement to the objective IEEE 1516 RTI, movement to a common FOM with the 3CE, and the integration of the Objective OneSAF (OOS). OOS is intended to replace the OneSAF Testbed (OTB), not only as the standard SAF constructive simulation, but also as the underlying SAF engine in various virtual simulators and several server tools.

As was mentioned earlier, migration to both the objective 1516 RTI and toward a common 3CE FOM was begun early in the development program. Development license versions of commercial 1516 RTIs have been acquired, and testing is well advanced with all federates with that RTI. The HLA Middleware stress test scheduled for August 2006 will include a stress sequence with the objective RTI to provide an analytical basis for further migration.

The last programmed evolution, replacement of OFOTB and its related servers by OOS, has also been well prepared for. The FOM developers have received an early copy of the OOS SOM, as well as the operating software, and tested integration issues with both DIS and HLA operating mode.

SUMMARY

Overall, the HLA Middleware integration was a nearly complete success, accomplished within budget and ahead of the mandated completion date (end FY 06). Although proprietary software system issues have prevented two federates from being integrated with the Middleware, one of those was fitted with a Gateway workaround, and the other was simply replaced as a working federate. The program was marked by a couple of false starts, notably the Perl scripts and the RID files, plus insufficient early coordination that caused the very late discovery of undocumented expedient PDUs. Those, in turn, required some minor modification of the Middleware. On the plus side, having to face new PDUs and to modify libraries during the development process caused the team to develop robust processes for changes that would be necessary in our follow-on work as the BLCSE evolved.

The creation of the BLCSE FOM was a challenge successfully met early in the development effort. This stood the team in good stead when the need to participate in 3CE common FOM development meetings arose. Although not planned initially, significant manpower assets were devoted to the 3CE program for common M&S approaches, a need that was met only because of the early FOM successes.

Active Middleware integration ended the first week of June 2006, as all but one site completed integration testing. At that time, a single site with two federates remained to be integrated, pending adoption of a workaround for incompatible operating system versions of Linux for one federate and the RTI.

With the successes of the Middleware development and integration, and subject to a successful stress test under operational conditions, the BLCSE is poised to prepare for migration to the HLA technical standard, as well as

full integration within the 3CE. Planning has begun for a large-scale shadow operation in Omni Fusion 07 during March 2007, followed by migration to an HLA operating environment during the summer of 2007.

REFERENCES

Minutes of the 3CE Advisory Council, 24 Feb 2005, Approved briefing 3CE Mission and Intent. (presenter, Tom Hurt)

Rieger, Lawrence and Pearman, Gerald, Re-engineering Legacy Simulations for HLA compliancy; *Proceedings of the 1999 I/ITSEC*

Altan, Engin and Lewis, Jennifer; Engineering a Common Operating Environment for the Army; *Proceedings of the fall 2006 Simulations Interoperability Workshop*

Jones, Jennifer; Transitioning Large-Scale Federations from IEEE 1278 to IEEE 1516, *Proceedings of the fall 2005 Simulations Interoperability Workshop*

Helfinstine, Bill, et al, RTI-s: A scalable HLA 1.3 RTI implementation Revision: 160, Copyright © 2005 Lockheed Martin Corporation, retrieved from https://166.25.29.26/ur2015/rtis-book/index.html