DESIGN AND IMPLEMENTATION OF THE BDS-D/HLA GATEWAY

Andy Cox, Douglas D. Wood
Institute for Simulation and Training
Orlando, FL

ABSTRACT

The High Level Architecture (HLA) is a project to develop a simulation infrastructure that will promote interoperability between simulations. One important consideration of HLA development is the ability to integrate legacy systems, preferably without incurring prohibitive costs in modifying existing software. One example of a legacy system in widespread use is the US Army Simulation, Training and Instrumentation Command's Battlefield Distributed Simulation-Developmental (BDS-D) M1 Tank Simulator. The simulator is a SIMNET-based training system being used in HLA prototyping efforts. The BDS-D M1 is the only crewed human-in-the-loop simulator involved in the HLA experiments.

This paper presents research conducted by the Institute for Simulation and Training (IST) to develop a stand-alone internetworking device allowing legacy applications to be interoperable with the HLA. The Gateway provides Run Time Infrastructure (RTI) communication on behalf of the M1 simulator, in a manner that is transparent to both the M1 and the HLA participants. Of particular interest will be the intricacies of converting from HLA attribute updates to SIMNET's PDU-based updates. Also of technical interest will be the rationale for maintaining remote entity approximation (dead reckoning) models within the interface.

ABOUT THE AUTHORS

Andy Cox is an Associate Computer Scientist at the Institute for Simulation and Training. He is currently involved in various projects in the areas of Distributed Interactive Simulation (DIS) and the High Level Architecture. Mr. Cox received a Bachelor of Science degree in Computer Science from the University of Central Florida and is currently pursuing a graduate degree. His background includes prior military service as an Infantryman, Quartermaster Officer, and Military Police Officer. His research interests are in distributed simulation and internetworking.

Douglas D. Wood is a Research Computer Scientist at the Institute for Simulation and Training. Mr. Wood has performed research primarily in the area of distributed simulation, including HLA experiments, electronic warfare protocols, and algorithms for computer generated forces. He has also performed research in simulation for emergency management training. Mr. Wood received a M.S. and a B.S. in Computer Science from the University of Central Florida.

DESIGN AND IMPLEMENTATION OF THE BDS-D/HLA GATEWAY

Andy Cox, Douglas D. Wood
Institute for Simulation and Training
Orlando, FL

INTRODUCTION

The Defense Modeling and Simulation Office (DMSO) is supporting several experimental applications in 1996 to test and refine the High Level Architecture (HLA) concept. One of those experiments is being conducted by the Platform Proto-Federation (PPF), a group of virtual real-time (i.e., DIS-type) simulations formed to test the applicability of HLA to that particular simulation category. A more detailed description of the HLA and PPF is provided in “High Level Architecture and the Platform ProtoFederation” (Harkrider, 1996). One member of the PPF is the US Army Simulation, Training and Instrumentation Command's Battlefield Distributed Simulation-Developmental (BDS-D) program. The Institute for Simulation and Training (IST) and STRICOM have chosen to connect the BDS-D M1 tank simulator to the larger HLA PPF via an interface node. The interface will convert HLA Runtime Infrastructure (RTI) services into SIMNET PDUs and forward them to the M1 Simulator. Similarly, SIMNET PDUs are translated into RTI service calls. This device is referred to as the IST HLA Gateway. The term *gateway* is used to describe the act of connecting two separate networks requiring translation at the transport layer and above. Such devices are also frequently referred to as protocol converters. IST and STRICOM have chosen to use the gateway approach to HLA integration so as to test its feasibility. This provides a mechanism for legacy applications, such as the BDS-D M1 crewed simulator, to utilize RTI services without requiring any modifications to existing software. If viable, the interface will enable the integration of the US Army's large inventory of existing SIMNET equipment into HLA exercises.

IST HLA GATEWAY

The BDS-D/HLA Gateway is an extension of IST’s Computer Generated Forces (CGF) Testbed (Smith, 1992). The Testbed provided a SIMNET and DIS compliant framework for coordinate conversions, dead reckoning, visual displays, and a protocol-independent simulation core. Gateway functionality entailed the addition of a Gateway Manager, RTI interface, and set of RTI services in accordance with the HLA Specification.

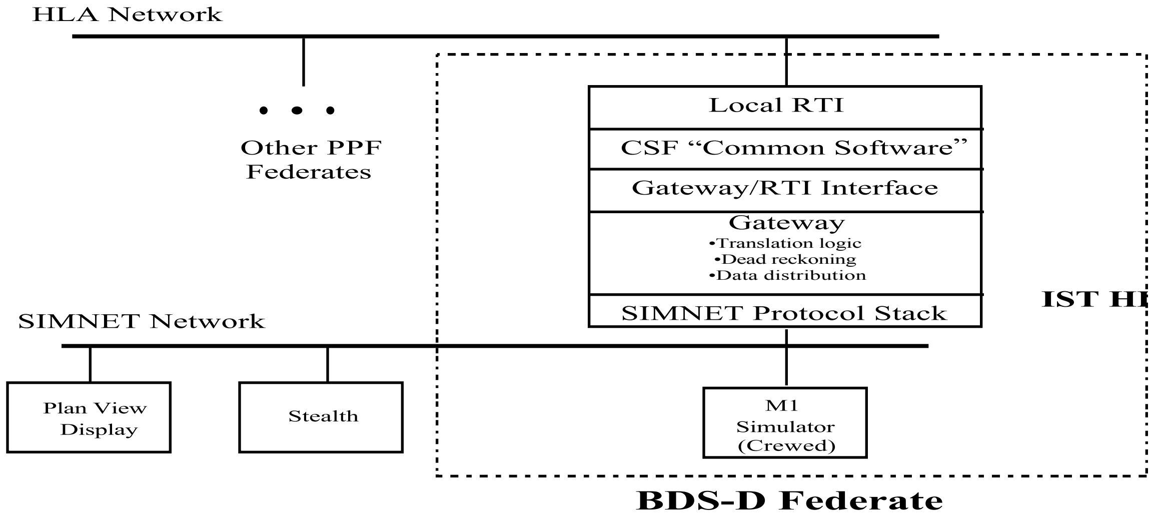

Figure 1. IST HLA Gateway and PPF Network.

The Gateway's RTI interface was developed using TASC's Component Services Framework (CSF), a set of middleware that encapsulates and extends RTI services in C++ classes (Bachinsky, 1996). To support translation between networks, two network interfaces are supported within the gateway. The use of separate networks has several benefits, including reduced processing at the simulation hosts and a lower average utilization on each network. A byproduct of the gateway's translation is that the entire HLA exercise may be observed remotely on the SIMNET network by other passive monitoring devices such as Plan View or Stealth visual displays. A diagram of the gateway and the overall PPF network is shown in Figure 1.

Gateway Design

The Gateway comprises several "manager" processes, each providing a specific type of service. The major processes used in the Gateway are listed in Table 1.

Manager	Function
Console Manager	Process console input
Display Manager	PVD/3D Display
Distribution Manager	Distribute information from the network
Executive	Message Scheduler
Gateway Manager	Provide RTI and Gateway services
Initialization Manager	Initialization and removal of entities
Protocol Manager	Provide protocol specific interfaces
Radio Manager	DIS Radio Handler
Simulation Manager	DIS Simulation Management services

Table 1. Gateway Manager Processes

The Protocol Manager was the only existing manager significantly effected by the addition of the gateway functionality, other than additional console input processed by the Console Manager. The Protocol Manager provides an interface between the internal application and the external protocols. The Protocol Manager receives incoming data from the simulation network and performs any necessary data transformations to create an internal representation. Similarly, outgoing data from the application is transformed into the appropriate simulation protocol format. Incoming data is forwarded to the Distribution Manager via the Executive message queue for further processing (i.e., dead reckoning model update). An additional communication path was added to the Protocol Manager to pass the data to the Gateway Manager via a direct function call. Output to the legacy simulation network via direct function calls already existed.

Similar to the Protocol Manager, the Gateway Manager provides an interface between the internal application and the HLA RTI. The Gateway Manager receives incoming data from an RTI Interface (see next section) and performs data transformations. For interactions, there is practically no difference; an HLA RTI interaction is always a complete set of data. In contrast, HLA RTI object updates or reflections are mostly partial data sets containing only changed data, excluding the first update or instantiation where a complete update is expected. Upon receiving an object update from the RTI Interface, the Gateway Manager gets a current copy of the entity's dead reckoned model from the Distribution Manager or creates one if this is the first update. The partial updates are applied to this state information which is then forwarded back to the Distribution Manager and simultaneously passed to the Protocol Manager for distribution to the legacy simulation network. Interactions are similarly distributed to both of these Managers.

The Gateway Manager accepts direct function calls to send application data to the HLA RTI via the RTI Interface, first performing any appropriate data transformations. A state diagram depicting the communication between the Protocol Manager, Distribution Manager, and Gateway Manager is given in Figure 2. The solid lines are messages or incoming data and the dotted lines are direct function calls.

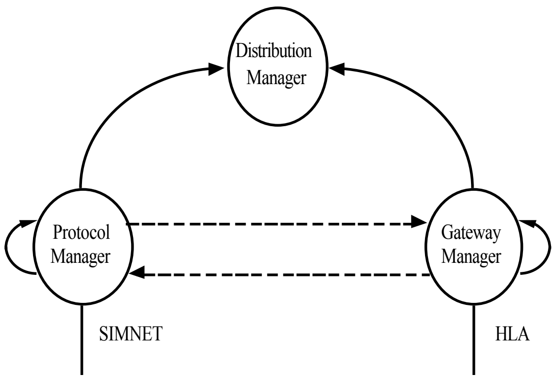

Figure 2. Gateway Message Flow.

The majority of the Gateway's modifications were largely related to the exchange of data between the legacy simulation protocol and the HLA RTI. However, the Gateway must provide additional functionality to meet the HLA Interface Specification requirements as implemented by the HLA RTI. The HLA RTI requires an application or federate to join a federation and to publish and subscribe to object and interaction classes. The federate must also be able to respond to calls invoked by the HLA RTI on the federate (i.e., incoming HLA data). The implementation of these functions and the transformation of application data into HLA RTI calls is described below.

RTI Interface

The RTI Interface is implemented through the four simple API functions shown in Table 2.

API Service	Function
Start	Initialization Services
Stop	Termination Services
Send	Send Data to RTI
Receive	Process Incoming RTI data

Table 2. Gateway RTI Services

The *start* function initializes the RTI Interface components and using these components it invokes the RTI services to create a federation (optional), join a federation, and subscribe and publish to object and interaction classes. The RTI Interface components are parameterized via a configuration file that contains parameters for naming the federation, specifying appropriate host machines (executive and ambassador), selecting subscription and publication classes, and determining whether to create a federation or join an existing one. As was mentioned above, the RTI Interface was developed on top of the CSF. The CSF components that are created during initialization are given in Table 3.

CSF Manager	Functions
Exercise	federation creation / join / resign / destroy
FOM	maintain transient database of object classes and their published attributes and interactions
Interest	subscription, database buffering of object reflections and interaction generations, process object and interaction queries
Object	publication, create objects, update object attributes, send interaction
Process	abstraction for HLA network event processing

Table 3. CSF Managers

The RTI Interface defines a publisher and a subscriber component to support the use of the CSF managers. The publisher and subscriber components define what object and interactions classes are published and subscribed (i.e., classes of data to be output and received). They are also used to support the transformation of data from the application into calls on the CSF Managers and vice versa. These two components are initialized and used for publication and subscription when start is called.

The RTI Interface *send* function accepts the full description of an entity state or entity interaction and transforms the data into object attributes and interaction parameters for delivery to the CSF Object Manager (OM). Since interaction parameters are not named, all interaction attributes are required and their positioned order is fixed. In contrast, object attribute updates are named and should only be sent when a change has occurred. Because the CSF OM determines when an attribute has changed, the RTI Interface delivers the complete set of entity state information to the CSF OM. The CSF OM only sends out those attributes that have changed. The design of the CSF OM also supports filtering attribute updates for those attributes not activated by the RTI (i.e., has no subscribers); however, this RTI capability has not been implemented.

The RTI Interface *receive* function first invokes the CSF Process Manager (PM) to process the RTI event queue (e.g., object reflection and interaction generation) and fill the CSF Interest Manager's (IM) object attribute and interaction database. Queries are then invoked on the CSF IM's database to extract the updated attributes and interactions. A message is constructed for each set of object attributes and for each interaction. The messages are sent to the Gateway Manager for data transformation and delivery to the Protocol Manager and the Distribution Manager. The database is then cleared for the next processing cycle.

The RTI Interface *stop* function invokes the CSF Exercise manger to resign from the federation and optionally destroy it.

Figure 3 shows the communication between the Gateway Manager and the RTI Interface.

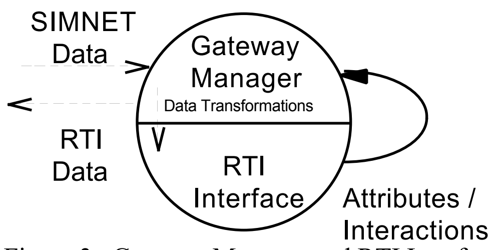

Figure 3. Gateway Manager and RTI Interface Communication.

It is significant to note that the specification of class and attribute names and attribute and parameter data types in the FOM is ingrained in the Gateway source code. Changes to the FOM would require recoding and recompiling the Gateway. However, these changes were mitigated by restricting affected code to the RTI Interface. To a small extent the design of the RTI Interface attempted to provide the flexibility to adapt to changing FOMs. The attributes and parameters used for exchanging data between the RTI Interface and the CSF (or RTI) are defined as "any" types or streams. These streams can store any type of data and are tagged with the data type they contain. The RTI Interface extends these types by adding two additional tags. One tag to identify the type expected internally and another to identify the type expected by the FOM. These additional tags can be used to perform data transformations when extracting data out of a stream or putting data into a stream. This technique allows a LOCATION attribute to be defined as three floats internally and three doubles in the FOM. Additional steps could be made to allow class and attribute names to be changed and initialized at startup via configuration files.

Federation Object Model

The HLA Specification defines the Federation Object Model (FOM) to prescribe the specific types and format of data exchanged between simulations (Department of Defense, 1996a). A FOM is tailored to the specific capabilities of the participating federates, including the objects, attributes, and interactions that comprise the federation. Information in the FOM is used to prepare a file of RTI Initialization Data (RID), which is used to initialize the RTI at the time a Federation is created. In SIMNET, information is exchanged through the transmission of PDUs. The SIMNET standard prescribes a set of PDUs, which are the basic unit of information exchange, the conditions under which PDUs are issued, and the range of values that each field in the PDU may be assigned. In addition to this high level application data, the gateway must also translate between different protocols used to provide lower level communication services.

Communication Services

There are significant differences between the communication services used by SIMNET and the RTI. SIMNET defines an association protocol (AP) to provide communication services underlying the simulation and data collection protocols (Pope, 1991). For HLA applications, communication services are provided entirely by the RTI and its use of the IP protocol suite. Each protocol is used to provide both reliable and unreliable services, although in many cases the gateway's translation requires a compromise between incompatible requirements. For example, SIMNET PDUs may be issued via a transaction service of the association protocol. The RTI, however, may be operating in a best effort mode using an unreliable broadcast or multicast datagram service in which no acknowledgment is possible. A similar incompatibility will occur if the RTI is providing reliable service for communication utilizing datagram transmission in SIMNET. Resolutions to issues such as these are frequently deficient in some regard. A conceptual diagram of the Gateway is shown in Figure 3.

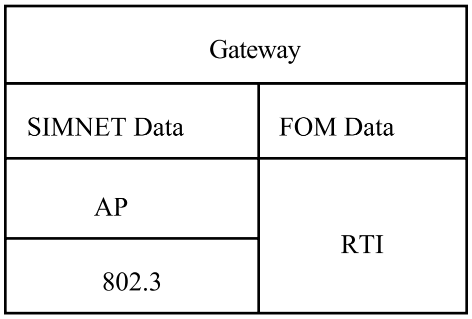

Figure 3. Gateway Conceptual Diagram.

Dead Reckoning and Periodic Updates

Dead Reckoning (DR) and Periodic Updates are two factors that significantly influenced gateway design. Dead Reckoning is a technique that reduces the frequency at which information must be transmitted via the underlying network. In addition to high fidelity dynamics information, each simulator maintains a dead reckoning model of itself and of all remote entities.

The DR model is used to extrapolate Time, Space, and Position Information (TSPI) to depict remote entities in the interval between updates. A simulator compares its high fidelity information with its DR model, and updates are issued only if some threshold value has been exceeded. Periodic updates, in contrast, often result in the transmission of redundant information. Regular updates provide several positive benefits, however, such as mitigating the impact of data loss in an unreliable network, allowing entities to quickly capture the state of an exercise regardless of the time of entry, and in regulating the amount of error that can be introduced by Dead Reckoning. The RTI is expected to provide minimum rate and state consistent categories of service to support these considerations.

SIMNET entities issue a Vehicle Appearance PDU based on one of three conditions: a change in discrete appearance attributes, a change in TSPI that exceeds some DR threshold, or upon the expiration of a vehicleAppearanceTime timer. Similarly, remote entities are considered to have timed out and are removed if vehicleDisappearanceTime has elapsed without receipt of a Vehicle Appearance PDU. In most categories of service, the RTI transmits only changes in state information, and does not transmit periodic updates. Accordingly, HLA entities do not time out, and must be explicitly removed. The gateway must ensure that periodic updates are provided to the SIMNET network. Two approaches are possible:

1. The gateway uses RTI services to query state information on a regular basis.
2. The gateway maintains state information on each HLA entity.

The first option was dismissed as requiring substantial overhead given the frequency at which the SIMNET network must be updated. The second option requires that the gateway maintain a dead reckoning model for each remote HLA entity to which it is subscribed. Given a DR model for each HLA entity, the gateway can determine with minimal overhead when a SIMNET PDU should be issued in the absence of an HLA update.

All attributes and interactions received from the RTI result in the transmission of a SIMNET PDU. Incoming SIMNET data, however, does not always require the invocation of an RTI service. Only changes in state information are required to be transmitted to the RTI. In the case of a quiescent SIMNET entity, no update is required. As previously stated, there are three conditions that require the issue of a SIMNET PDU. One of these includes a special case that led to considerable discussion. A moving SIMNET entity can conceivably be within its DR thresholds at the expiration of the vehicleAppearanceTime timer. At that time, it will issue an Appearance PDU. The location within the PDU will have changed. If this information is not forwarded to the RTI, then error will be introduced between the SIMNET and RTI DR representations of the issuing entity. This could possibly be addressed by maintaining two DR models within the gateway. However, an equally plausible resolution is to forward the new TSPI to the RTI, while still filtering out any unchanged data. This reduces the computational overhead within the gateway and should be a more scaleable solution.

Object ID versus Entity ID

SIMNET identified objects through a Site/Host/Entity ID, each component consisting of two octets. RTI objects are assigned an object identifier of four octets. The SIMNET ID has additional meaning beyond simple object identification, providing logical classifications based on geographic location (site) and host workstation, if applicable. If this additional information is not required, an RTI object ID can be used to generate a unique SIMNET ID. One approach is to use the high order octet as the SIMNET site, the next octet as the host, and the two low order octets as the entity ID. For example, object ID 0x01234567 would map to the unique SIMNET ID site 0x0001, host 0x0023, entity 0x4567. The HLA-PPF FOM included a Site/Host/Entity ID as a FOM attribute, in addition to the RTI Object ID.

Exercise ID versus Federation Name

An exercise ID and federation name are used in SIMNET and by the RTI, respectively, as a distinct exercise identifier. These parameters are configured prior to run time within the gateway.

Performance

The Gateway succeeded in connecting the BDS-D M1 Tank Simulator communicating via SIMNET PDUs into the HLA PPF experiment exercises communicating via the HLA RTI. The functional data flow between the two simulation network communication mechanisms proved to work in a straight forward manner. The SIMNET PDUs were readily transformed into HLA RTI function calls for object updates and interactions. Similarly, HLA object reflections and interaction generations were used to produce the appropriate SIMNET PDUs. The Gateway performed the necessary exercise setup required by HLA (e.g. join, publish, subscribe) without affecting the SIMNET side. Presently, the HLA Gateway does not handle HLA ownership transfer functions. Also, additional RTI services such as query, delete, and time management will be considered to increase HLA functionality.

Preliminary analysis of the data collected during PPF experiments have shown end to end latencies for RTI communication are significantly higher than similar communication via standard DIS implementations. For example, consider the results of two federates exchanging object updates using RTI best effort (broadcast) communication on a local Ethernet network under light load. Frequently these updates exhibited average latencies above the 100 to 300 milliseconds required by DIS simulation exercises. In contrast, DIS packets typically incur less than 10 milliseconds under similar conditions. The RTI latency does not include processing within the IST HLA Gateway, which required an additional 1 to 20 milliseconds to process each RTI message, depending on the number and type of attributes. It is anticipated that Gateway performance may be increased through optimizations in the Gateway, CSF, and RTI.

Detailed measurements of Gateway and RTI performance are still underway at the time of this paper. Final results of the analysis of PPF experiments are scheduled to be complete in early September 1996.

CONCLUSIONS

The gateway has been successful in providing a link between SIMNET applications and HLA-based simulations exchanging data through the Run Time Infrastructure. The use of a stand alone internetworking device appears to be a viable approach for cases in which re-engineering costs to achieve HLA interoperability may be prohibitive. We conclude that integrating legacy systems, in particular SIMNET and DIS simulations, into HLA federations with a Gateway is a feasible, convenient, and cost effective alternative.

References

Bachinsky, S. T., Hancock, J. P., Hooks, M. L., & Rybacki, R. M. (1996) Common Software Delivery 1: Status, Plan, Design, and API. TASC.

Harkrider, S., & Petty, M.D. (1996) "High Level Architecture and the Platform Protofederation." Submitted for consideration to the 18th I/ITSEC Conference.

Pope, A. R. The SIMNET Network and Protocols. (1991) Version 6.6.1. BBN.

Smith, S. H., Karr, C. R., Petty, M. D., Franceschini, R. W., & Watkins, J. E. (1992) "The IST Semi-Automated Forces Testbed." IST.

Department of Defense. (1996a) "High Level Architecture for Simulations, Object Model Template Draft v0.2." DMSO.

Department of Defense. (1996b) "High Level Architecture for Simulations, Interface Specification Draft v0.5" DMSO.