AIAA Modeling and Simulation Technologies Conference and Exhibit
16 - 19 August 2004, Providence, Rhode Island

AIAA 2004-5436

Distributed System Architecture in VAST-RT for Real-Time Airspace Simulation

R. D. Lehmer*
Northrop Grumman Technical Services Inc.
NASA Ames Research Center
Moffett Field, CA 94035

S. J. Malsom†
Aerospace Operations Modeling Office
Aviation Systems Division
NASA Ames Research Center
Moffett Field, CA 94035

Abstract

The Virtual Airspace Simulation Technology Real-Time (VAST-RT) project at NASA Ames Research Center has implemented a new airspace modeling and simulation capability that provides a gate-to-gate, real-time, human-in-the-loop simulation environment to evaluate revolutionary Air Traffic Management (ATM) concepts. The system architecture of the VAST-RT connects high-fidelity flight simulators and air traffic control (ATC) simulators with each other and with low to medium fidelity target generators, as well as provides an interface for various models, agents, and decision support tools to create a flexible, extensible, and reconfigurable environment. The VAST-RT system architecture is responsible for delivering simulation data, including audio and video streams, in real-time to each of the participating simulation systems or facilities, as well as providing simulation synchronization and timing information. The architecture is based primarily upon the DoD developed High Level Architecture (HLA), however, the VAST-RT project has developed a standard interface method to reduce the time and cost of integrating existing simulation systems into the distributed simulation environment. In addition, VAST-RT has adopted the concept of bridging between HLA federations to overcome incompatibilities between different HLA run-time implementations as well as differences in modeling of simulation data. The design and implementation of bridges between HLA federations, portals from non-HLA data networks to HLA federations, and real-time communications toolboxes that provide "best-practices" software interfaces for VAST-RT will be discussed. The development of solutions for distributed audio and video communications across the simulation environment will also be described.

Introduction

Research into the future of the air transportation system critically depends upon the ability to evaluate the effects of revolutionary system concepts across the entire airspace. In order to meet the expected increases in air passenger-miles and daily flights, as well as provide a smooth transition for the introduction of new technologies, several of these revolutionary concepts can be expected to be blended with the normal evolutionary improvements in the existing National Airspace System (NAS). The testing of these revolutionary concepts through human-in-the-loop (HITL) simulation can provide information on both improvements in NAS system capacity and human-factors related impacts on pilot and controller workloads.

VAST-RT is part of NASA's Virtual Aerospace Modeling and Simulation (VAMS) Project and the primary goals of VAST-RT are to develop a real-time simulation infrastructure and execute real-time ATM simulations that have a human-in-the-loop component. To meet these goals, the VAST-RT project has developed an extensible architecture to integrate new technologies with existing simulation systems to produce a real-time simulation environment for simulating the national Air Traffic Management (ATM) system.[1] The environment for these simulations includes high-fidelity flight simulators, high and medium-fidelity tower and air traffic control simulators,

* Air Traffic Management Group Leader, AIAA Member
† VAST-RT ATM Simulation Project Manager

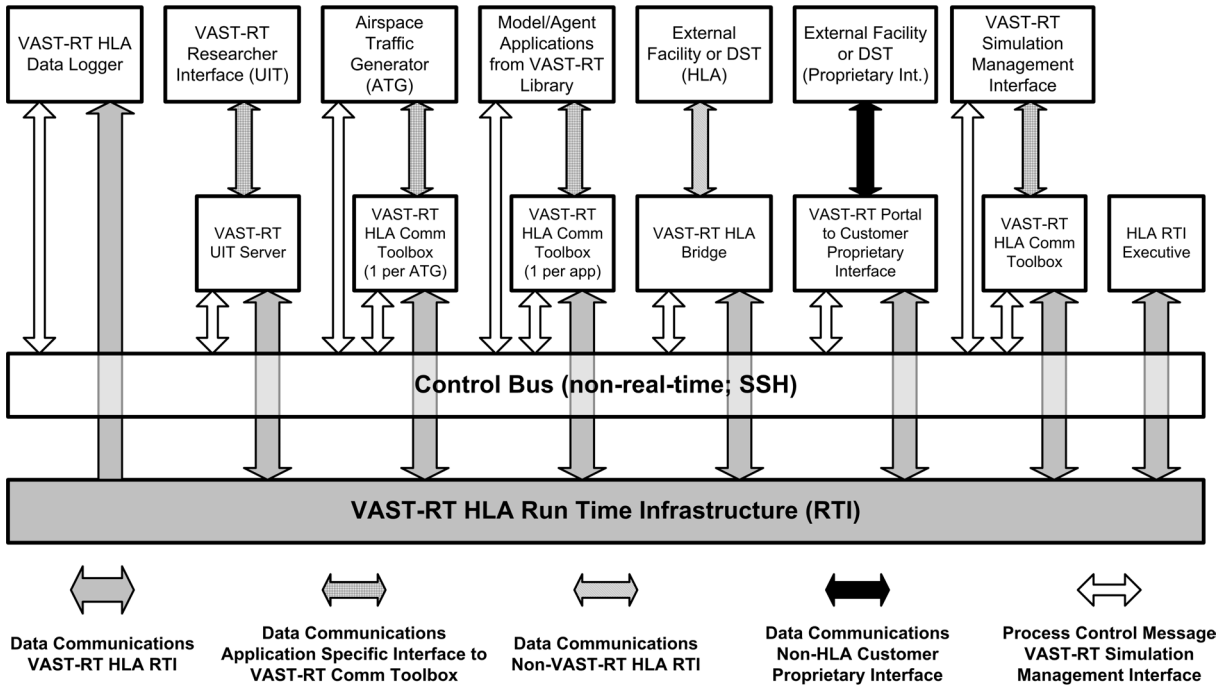

Figure 1 - Schematic diagram of the real-time data communications between individual simulators using the VAST-RT System Architecture and HLA. The VAST-RT HLA RTI handles the real-time data, while the Control Bus refers to the non-real-time scripting tools that are used to manage the VAST-RT software components.

and low-fidelity target generators. These simulators are seamlessly integrated together to generate realistic flows of traffic to support the evaluation of future operational concepts in a transformed National Airspace.

This paper is arranged as follows. The next section introduces the VAST-RT system architecture. Following that is a more detailed discussion of the software toolbox that makes up the VAST-RT interface. The VAST-RT project has also developed tools for bridging different simulation architectures based on the VAST-RT software toolbox. Other simulation components based on the VAST-RT software toolbox are also discussed. Finally, the experience of integrating audio and video in the VAST-RT architecture is presented.

System Architecture Design

The VAST-RT system architecture provides the protocols and infrastructure to design, execute, and analyze a distributed simulation of the Air Traffic Control (ATC) system on a NAS-wide scale. The design of the VAST-RT system architecture has focused on the deterministic real-time performance of the system infrastructure. In addition, the architecture has been designed to allow the flexibility to integrate a wide variety of simulation systems and analysis tools.

The VAST-RT system architecture is required to handle a variety of data types, including simulation state data such as aircraft position, flight plan and intent information, audio communications between participants in the simulations that simulate radio and land-line communications, and digital video recordings of participant actions for human-factors related research. In addition to handling these types of data in a deterministic, real-time fashion, the VAST-RT system has to provide the tools to access and manage the collected data.

After evaluating the types of messages and protocols that would be needed in a simulation environment containing flight simulation and ATC components, a decision was made early on to use the DoD High Level Architecture (HLA)[2] as the back-bone for the real-time communications between simulator systems. The selection of HLA as the distributed simulation architecture is based on the success that it has enjoyed in integrating both flight[3,4] and other distributed simulation environments.[5,6,7] Compared to other messaging protocols that have been used in flight simulation (SIMNET and DIS, as examples), HLA is

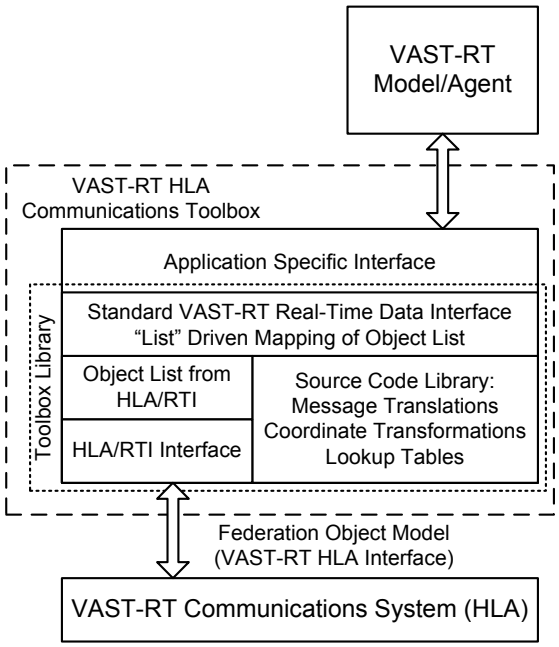

Figure 2 – The VAST-RT HLA Communications Toolbox connects a simulation model or agent to the underlying HLA infrastructure.

superior in its performance and flexibility, especially once the simulation environment expands beyond a single facility or network subnet.[8]

Figure 1 shows a generic application of the VAST-RT system architecture in schematic form. A variety of simulation components, shown in the top row of boxes, have been successfully networked together using the HLA Run Time Infrastructure (RTI). These components include air traffic generators, other models and agents such as flight deck simulators, and other networked applications, such as ATM DSTs. Between these simulation components and the real-time infrastructure lies a set of proxy applications. Each of these proxy applications is modeled on a collection of standard procedures known as the VAST-RT HLA Communications Toolbox. The Toolbox's prime objective is to convert data and messages to and from a simulator's native external interface and the VAST-RT HLA RTI. Each HLA Communications Toolbox is an independent entity in the VAST-RT HLA Federation and is known as a Federate in that Federation.

In addition to the VAST-RT HLA Communications Toolbox, the VAST-RT system architecture has to communicate with other distributed simulation environments, such as other HLA RTI implementations. Traditional HLA Federation development methods would normally drive all participants to build a common HLA Federation Object Model (FOM) and agree to use the same version of the HLA RTI.[9] However, in circumstances where the cost of modifying an existing simulator's HLA interface was prohibitive, VAST-RT has employed novel techniques to interconnect simulation components. The VAST-RT project has developed an HLA Bridge application that not only converts between different HLA FOMs but also between different HLA RTIs.

In addition to HLA, the VAST-RT architecture also needs to interface and interoperate with other distributed simulation systems, such as the Airspace Operations Laboratory (AOL), another Air Traffic Control simulator at NASA Ames. The AOL uses a custom interface referred to as the Aeronautical Datalink and Radar Simulator (ADRS).[10] The ADRS architecture has a different methodology for exchanging data between participants than what is found in HLA. VAST-RT has developed an application, known as a Portal, to interface between the ADRS and HLA data transfer methodologies. Portals and Bridges in VAST-RT can be easily built-up from the common components of the VAST-RT HLA Communications Toolbox.

While HLA provides the architectural framework to transfer data between simulation components during the real-time operation of the simulation, a non-real-time control scheme is required to bring the system on-line. VAST-RT has developed scripting to invoke each of the VAST-RT components and dynamically control certain HLA Federation parameters (the Federation end-point, for example). Access to this control scripting is limited to a UNIX secure shell (SSH) to provide basic connection security.

The VAST-RT HLA Communications Toolbox

The VAST-RT HLA Communications Toolbox is a set of C++ classes and methods that provides a common interface to the VAST-RT simulation environment. The Toolbox consists of two major components (see Figure 2). One component is a library of common code that supports a standard interface to the VAST-RT HLA RTI. The second part is the application specific portion of the Toolbox that connects the standardized VAST-RT interface to the external simulator interface. The Toolbox library standardizes the methods for marshalling the data to and from the HLA RTI interface. These methods access the distributed simulation data from an ordered list of objects that are accessible to the application specific portion of the Toolbox. The Toolbox also provides the interface for the HLA Interaction behavior, which is a system for

sending non-persistent messages and is used for event based data and other simulation messaging.

The application specific portion of the Toolbox also has access to standard methods for instantiating and deleting objects in the simulation and publishing attribute updates for those objects. By standardizing the low-level object management, attribute update management, and data marshalling, the Toolbox concept reduces the time and complexity of integrating new model components into the VAST-RT simulation environment while minimizing the amount of HLA RTI specific programming knowledge required.

Another method used in the Toolbox architecture to reduce the integration effort is to automatically synchronize the Toolbox data structures used in the object list with the FOM. Thus, changes to the FOM become immediately available to the developer simply by recompiling the Toolbox. In order to automate this process, the VAST-RT HLA Communications Toolbox development process relies on the Object Model Development Tool[11] (OMDT) that is distributed as part of the DMSO (Defense Modeling and Simulation Office) HLA RTI. The OMDT is used to manage data describing the HLA Federation, such as the names and types of attributes and interactions, as well as publishing, subscribing, and other information used in the HLA Federation. The data contained in the OMDT serves two purposes in the VAST-RT development process. First, the OMDT is used to generate the Federation Execution Details (FED) file that defines the FOM in HLA. Second, a VAST-RT developed parser extracts from the OMDT data file the type and cardinality of each attribute and writes out a FOM-specific class header file. This header file is referenced in the application specific portion of the Toolbox when data from the list of HLA objects is accessed. Using the OMDT and the Toolbox parser, changes to a FOM can be made consistently between the Toolbox and the HLA Federation configuration and rapidly integrated into the Toolbox, which is an important feature in a research and development environment where interface requirements may change significantly between experiments.

The architecture of the VAST-RT HLA Toolbox enforces the concept of best practices in VAST-RT interface software development by minimizing the amount of application specific software that is required to interface each model or agent. Likewise, the HLA Communications Toolbox also allows VAST-RT to impose standards for data types and units by providing libraries of globally agreed upon constants, standard conversions, coordinate transformations, and look-up tables.

Each HLA Communications Toolbox operates asynchronously in the real-time environment. Each Toolbox has a local timer that is typically set between 5Hz and 30Hz depending upon the specific data requirements of the connected simulation component. This timer triggers a process that checks for updates from other HLA Federates and updates the attribute values in the object list. The Toolbox also processes data queued up from the application specific interface and prepares it for the RTI to send it to the other Federates. Since data is moving asynchronously between Federates, each HLA Communications Toolbox caches the last valid update of an object attribute. Caching greatly reduces the data transmission requirements within the Federation since only updates need to be transmitted. Caching also increases system performance by allowing the Toolbox to handle requests for data from other HLA Federates without interrupting the simulation components attached to them.

The VAST-RT HLA Communications Toolbox has been specifically built using the DMSO version 1.3NGv4 of the HLA RTI, though other versions of the DMSO HLA RTI could be easily adapted to the Communications Toolbox framework. Adapting the Toolbox to another HLA RTI implementation outside the DMSO framework, while not impossible, would not be trivial as the HLA RTI interface layer has been specifically coded for the DMSO HLA RTI to maximize performance of the Toolbox.

Bridging of HLA Federations

The VAST-RT system architecture was designed for the rapid integration of simulation components from various sources. However, it is not always practical to integrate the VAST-RT HLA Communications Toolbox into an existing simulation model or agent application. For example, a simulation model or agent may already have an HLA interface integrated, and it may be more efficient to use the existing interface rather than integrate the VAST-RT HLA Communications Toolbox into the simulation component. This is especially true for proprietary simulation systems with closed architectures or for systems supported by organizations that no longer have the tools or resources available to modify the internals of the simulator; then the cost of retrofitting a new interface becomes prohibitive even if it is technically possible. In order for the VAST-RT HLA implementation to interact with a simulator's existing HLA interface, two implementation problems must be overcome. First, most HLA RTIs are incompatible with one another, and, second, there is almost always a difference in the FOMs used in VAST-

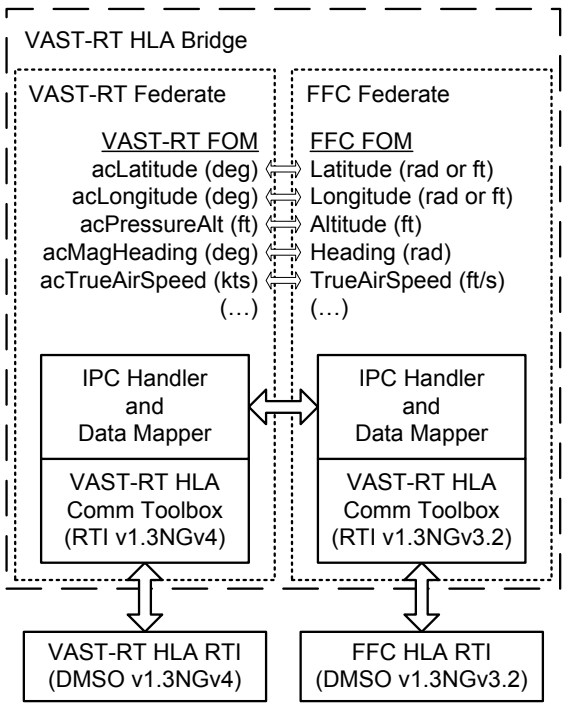

Figure 3 – Example of a HLA Bridge application in the VAST-RT Architecture. Bridges not only convert communications between incompatible HLA RTI implementations, but also handle coordinate transformations and unit conversions between different Federation Object Models.

RT and the existing simulator. VAST-RT has developed an HLA Bridge architecture that interfaces two Federates in separate Federations so that the Federations seamlessly share simulation data between each other.

The HLA Bridge architecture for VAST-RT is based on two independent executions of the VAST-RT HLA Communications Toolbox, with each one configured to support the HLA RTI and FOM that are defined by one of the two Federations being bridged. Customized code is required to map attributes from objects in one FOM to object attributes in the other FOM, as well as deal with differences in RTI execution. The attribute mapping also includes any necessary coordinate transformations and unit conversions.

The Bridge was designed to be as transparent as possible to the passage of data in the simulation. As objects are published in one Federation, they are discovered by the Bridge because it is set up to subscribe to all the object classes of interest. Attributes from objects are mapped to their corresponding attributes in the other Federation, and then published by the Bridge in that Federation. Attribute updates, like the object discovery and reflection, are handled between the two processes in the Bridge through a UNIX Inter-Process Communications (IPC) interface that links the two processes. Finally, the transport delay through the Bridge is reduced by cycling the Toolboxes as fast as possible so that attribute updates and other messages are processed with a minimum of delay.

Figure 3 shows an example of an HLA Bridge that was built for VAST-RT. The FutureFlight Central (FFC) facility is a HITL tower simulator at NASA Ames that has a vendor-provided HLA interface to the Adacel MaxSim® simulation software.[12] In order to integrate the VAST-RT architecture with the existing FFC interface, three major issues needed to be addressed.

First, the HLA RTI implemented in the FFC is based on version 1.3NGv3.2 of the DMSO HLA RTI, while VAST-RT uses the similar, but incompatible, version 1.3NGv4. The two HLA Toolboxes that make up the HLA Bridge in this example are optimized for their specific version of the RTI. In addition to spanning the different HLA RTIs, the VAST-RT component of the HLA Bridge has implemented certain HLA RTI callbacks that are required for the proper operation of the VAST-RT HLA Federation that are not utilized in the FFC HLA interface. These additional callbacks make the FFC interface to the VAST-RT simulation environment more flexible and robust, thus increasing overall reliability of the full distributed simulation environment.

Second, the FOMs in VAST-RT and FFC are different, with examples of the differences in FOM attribute names shown in Figure 3. These are examples of attributes that map between the two federations. There are also a number of attributes and classes that do not map between the VAST-RT and FFC HLA federations, especially in the area of flight plan and flight plan intent data. The HLA Bridge's role is to make sure that the available data from the FFC's HLA interface is mapped into the VAST-RT FOM and vice versa, but not preclude another HLA Federate from publishing additional classes and attributes that would fully populate the model of an FFC generated target in the VAST-RT environment.

Lastly, the HLA Bridge application is required to handle unit conversions and coordinate transformations of the attributes mapped between the two FOMs. Again, looking at the examples in Figure 3, the Bridge needs to handle the conversion of heading attributes between degrees and radians, and speed attributes

- Aircraft State Object Class
  - Identity (Callsign, Type, Tail Number, etc.)
  - Position (Lat., Long., and Altitude)
  - Heading (True and Magnetic)
  - Speed (Air and Ground)
  - Attitude (Pitch, Roll, and Yaw)
  - Articulation (Lights, Flaps, etc.)
  - ADS-B Surveillance Data
- Flight Plan Object Class
  - Flight Plan
  - Runway and Gate Assignments
  - Center and Sector Assignments
  - Meter and Outer Fix
  - Taxi Information
- Aircraft Object Class
  - Tail Number
- Ground Vehicle Object Class
  - Identity
  - Position (Latitude and Longitude)
  - Heading
  - Speed
- Simulation Status Object Class
  - Simulation State (Run, Pause, Stop, etc.)
  - Simulation Standard Time
- Federate Status Object Class
  - Federate Status (Ready, Running, Not Connected, etc.)

Table 1 – Overview of types of data attributes associated with the six major HLA Classes in the baseline VAST-RT FOM as it is currently implemented.

between knots and feet per second. In addition, the FFC HLA interface has a unique dual use scheme for its position data attributes, whereby the position data is in longitude and latitude when an aircraft is in the air, but in x and y coordinates relative to a local origin when the aircraft is taxiing on an airport surface. The Bridge has to cope with this unique coordinate conversion scheme and publish the position of the aircraft in latitude and longitude coordinates in the VAST-RT Federation.

VAST-RT Federation Object Model

As described earlier, the VAST-RT real-time interfaces are defined by the VAST-RT HLA FOM and are provided to various model and agent processes via a proxy method known as the VAST-RT HLA Communications Toolbox. Based on the requirements of the simulations that the VAST-RT architecture is to support, key objects have been defined as the baseline of the VAST-RT HLA FOM. The Toolbox enforces the use of the VAST-RT HLA FOM, and also provides the tools to rapidly modify the FOM so as to allow users of the VAST-RT system architecture to add additional attributes and classes as future simulation requirements dictate.

The baseline VAST-RT FOM, as outlined in Table 1, serves as the foundation data exchange model for the development of the VAST-RT system architecture. More than 100 attributes are grouped into the six classes shown in Table 1 and primarily focus on aircraft state and basic ATC data. For example, the model of a flight is made up of the association of an Aircraft State object and a Flight Plan object. The Aircraft State Object Class is a complete description of the vehicle's state data (identity, position, heading, speed, etc.), while the Flight Plan Object Class includes flight plan and other ATC data regarding schedule and intent of the flight.

In addition to flights, the VAST-RT FOM has two additional classes to manage vehicle objects in the simulation. First is the Ground Vehicle Object Class that is used to handle vehicles found at airports, such as baggage carts, fire trucks, and other service vehicles. The Aircraft Object Class is an object class that provides additional book-keeping capabilities by identifying objects based on tail-number rather than flight number. Having these separate classes for vehicles allows the HLA RTI to efficiently filter traffic between HLA Toolboxes through the use of the subscription controls in each HLA Federate. As an example, a Center traffic simulation does not require information about baggage cart position from an Airport simulation and, therefore, the Toolbox supporting the Center simulation would not subscribe to the Ground Vehicle Object Class.

Finally, there are two status objects defined in the VAST-RT FOM. The first, the Simulation Status Object Class, is used by the VAST-RT Simulation Management Interface (SMI) to communicate to all the HLA components. An additional discussion of the VAST-RT SMI is found in the following section. The second, the Federate Status Object Class, is designed for each Federate in the VAST-RT Federation to send specific status data for Federation status monitoring.

In addition to the HLA Object Classes, which refer to objects with persistent data, the VAST-RT HLA Communications Toolbox supports HLA Interactions, which model message behavior that does not have persistent data. In some cases, simulators will modify persistent data published in the simulation based on the contents of Interaction messages, but the Interaction messages themselves do not modify the data at the HLA RTI level. Some examples of HLA Interactions

include a variety of interaction messages having to do with simulation state behavior, such as messages to start and stop execution of the simulation or transfer ownership of objects from one simulator to another. Interaction messages can also be used to model actual messages in the simulation, such as flight plan amendments.

Additional System Component Descriptions

In addition to the VAST-RT HLA Communications Toolbox and the HLA Bridge application discussed in earlier sections, the VAST-RT Project has developed several pieces of software to meet infrastructure requirements for such a heterogeneous distributed simulation capability. Four examples of these software components are highlighted in this section.

First, as mentioned earlier, VAST-RT has developed a Portal application that is an interface application between the HLA-based distributed data system of VAST-RT and the ADRS-based system used by the AOL. The differentiation between a Portal and a HLA Bridge is a subtle one, based primarily on the differences in the data marshalling and other software methods used by the attaching interface. A Bridge is characterized by a great deal of similarity in the software methods used in the two bridged HLA RTIs. A Portal, on the other hand, has the added complexity of compensating for differences in data availability, ownership, and transfer scheduling found in the external system. In the case of connecting to the ADRS, the VAST-RT Portal uses the ADRS Multi-Purpose Interface to pass state and flight plan data between the HLA Toolbox interface and the ADRS, as well as adapt the ADRS target control to the VAST-RT Ownership Handoff Manager interface.

One of the difficulties that the ADRS Portal resolves is the scheduling of data transfers. The VAST-RT HLA Communications Toolbox is an asynchronous data transfer system that locally buffers the last valid update of an attribute at the receive end of the system, while the ADRS interface is synchronously scheduled to receive and transmit data. Since the asynchronous update of the VAST-RT HLA Communications Toolbox, typically scheduled between 5Hz to 30Hz, is much faster than the 1Hz maximum rate of the ADRS interface, the transmission of attribute updates from VAST-RT to ADRS should appear smooth to the ADRS since it is only sampling data once a second. Likewise, the synchronous transfer of data from ADRS to VAST-RT will not significantly impact the VAST-RT simulation environment since all of the Toolbox components are already set up to manage data from multiple asynchronous sources.

Second, the VAST-RT simulation environment is made up of a number of flight deck and target generation simulators, and other simulation systems. These simulation components have high levels of fidelity within specific domains (en-route, arrivals, departures, taxiing, etc.) or areas of the simulation (such as around a particular airport, for example). Outside of its particular domain, a simulator's fidelity may be limited and another simulation component may have better fidelity. Thus, it is necessary to have a mechanism to control which simulation component is responsible for the publishing and updating of simulation data for a particular aircraft, or other object. Furthermore, the ownership of a particular aircraft may move from one simulator to another during a simulation as it transitions between the various flight domains.

Rather than have each simulator or simulation system determine whether it is best able to simulate a particular aircraft and, thus, have the ownership of the related HLA objects, an Ownership Handoff Manager (OHM) was developed to perform this function. Having the OHM as an automated decision tool for determining object ownership significantly reduces the complexity of Toolbox interfaces by eliminating the need for logic to negotiate ownership between individual components of the simulations. The OHM monitors the state of objects in the simulation and determines which simulator should have the responsibility to publish information about the aircraft based on the aircraft's physical location, flight mode, and intent. Provision has been made within the OHM to not allow the transfer of ownership of certain types of objects, such as aircraft simulated by piloted flight deck simulators.

A transfer of ownership request is signaled by the OHM using an HLA Interaction message.[13] The Interaction message is parameterized with the handle of the object that is to be transferred and the federate that is to acquire the object. In this manner, all transfers of ownership in the VAST-RT infrastructure are negotiated-pull transactions. The acquiring federate for the object negotiates with the divesting federate through the DMSO HLA RTI to gain ownership of the object attribute and then the acquiring federate is responsible for continuing to publish updates to the attribute. Up to this point, this transfer of ownership mechanism has been automated in the OHM; however, the HLA Interaction is sufficiently generic that, in the future, a particular simulator, using the HLA Communications Toolbox, could generate a transfer of ownership request based on the manual intervention of a test engineer or simulation participant.

It should be noted here that the transfer of ownership between simulators must be differentiated from the ATC Transfer of Control that occurs as a flight moves from one controller's authority to another. It may turn out that the boundaries for transfer of ownership and Transfer of Control coincide in some simulation configurations based on the airspace configuration or function that the participating simulators model, but the transfer of ownership mechanism does not model the same controller to controller and controller to pilot messaging that would be involved in negotiating an ATC Transfer of Control event.

The User Interface Toolbox (UIT) is the third example of a VAST-RT developed software tool. The UIT is the primary researcher interface for both real-time and post-simulation data access. The main component of the UIT consists of a dual-purpose server that accesses real-time data from the VAST-RT HLA Communications Toolbox during simulation execution and retrieves post-simulation data from a post-simulation data archive. In addition to the UIT Server, a client application with rudimentary data inspection tools and situational displays gives the researchers basic information on the status of the simulation and the state of the aircraft in the simulated airspace. The UIT Server and Client communicate with XML formatted messages to provide increased flexibility in the type and format of data that can be sent to the client without making major changes to the established client-server interface.

Finally, all of the distributed components that connect to the real-time HLA interface need to be managed to allow test engineers and researchers to control and monitor the simulation configuration and execution. The VAST-RT Simulation Management Interface (SMI) is a client-server tool that allows test engineers and researchers to control the execution of various VAST-RT simulation components (toolboxes, bridges, data loggers, target generators, other ATM components, etc.) and provide global information on simulation state (running, paused, or stopped) or other global information (the elapsed simulation time, for example). However, the control requirements of piloted HITL simulators are severely restricted due to human-occupancy safety requirements of those simulators. The control of HITL simulators is planned to be limited to the execution of the HLA Communications Toolbox process. Any information sent to a HITL simulator regarding VAST-RT simulation state will be taken by the HITL simulator as an advisory rather than a command. The SMI uses the existing VAST-RT non-real-time scripting discussed earlier in the paper to control the execution of the various VAST-RT components prior to the simulation being ready to operate.

Data Collection Requirements

In addition to generating simulation data, the VAST-RT system architecture is required to collect all of the data being published on the real-time HLA RTI interface. Because of the choice of HLA RTI implementation for VAST-RT, a COTS (Commercial Off the Shelf) HLA data logger was not available that met the VAST-RT data collection requirements. Using components of the VAST-RT HLA Communications Toolbox, a customized data logger application has been built. The design of the VAST-RT HLA Data Logger (HDL) Federate is significantly different than the VAST-RT HLA Communications Toolbox since the HDL does not have to manage data in the same way as any of the simulators. It does not use the list-driven mapping of data from the HLA RTI interface, but instead, the application is designed to receive updates from the HLA RTI directly and stores them as records in a data file that includes a locally generated time-stamp to document the time that the updates are received at the HDL.

The HDL data are converted post-simulation into time-series data sets for later analysis by researchers. In addition to the data collected by the HDL, more detailed engineering data is generally collected by each simulation component's native data collection system. The VAST-RT project has implemented several simulator specific data conversion tools that allow for the importation of locally collected simulator data sets into the post-simulation data repository. Also, the post-simulation repository is maintained on a system not involved in the real-time collection of simulation data so that the archive's use will not impact the performance of the real-time data collection.

Incremental Development Process

Not all of the components of the VAST-RT system architecture were developed simultaneously. Since the VAST-RT simulation environment can be considered a system of systems, techniques to decompose and manage large, complex software projects were employed.[14] One of the project management techniques adapted to the VAST-RT system architecture development is the use of incremental software releases. By focusing on a few key technical issues in the development phase of each incremental release, the highest priority risk areas can be addressed during each step in the development process. At the end of each incremental release, if the implemented solution does not fully address the risk or uncovers a new risk, or

additional complexity is added based on new requirements, the priorities for the next incremental development period are reevaluated. In that way, high risk or not well understood requirements get evaluated earlier in the project lifecycle and are not put off until the end of the project.

This project management process also leverages the development of prototypes to test concepts before significant software development resources are committed to integrating them into the overall system architecture.[15] The HLA Bridge and the OHM benefited particularly from this process. The idea of bridging different Federations and HLA RTIs was tested outside of the HLA Communications Toolbox framework and focused on simply transferring data from one or two attributes between two equally simple Federations. Likewise, the transfer of ownership mechanism was developed completely outside the VAST-RT environment. Once the concepts are tested and the outstanding architectural and technical questions have been answered by building and testing the prototype, the formalism of the production development environment is used to produce the software product that is ultimately delivered.

Audio and Video Communications

The selection of HLA as the solution for the real-time communications of simulation data was based on internal organizational experience and the published experience of other facilities with HLA, and the basic architecture was based on the results of early prototyping. On the other hand, the VAST-RT requirement to use HLA to transport the audio communications data in the simulation environment was a riskier proposition because there is much less published experience with HLA audio. There are a few demonstrated examples of the use of HLA to transport audio communications,[16] and these are principally HLA and DIS hybrid architectures. Since most of the existing simulation systems at NASA Ames have a DIS enabled audio communications system, the HLA-DIS hybrid architecture has been implemented for VAST-RT.

After a significant period of evaluation and prototyping, a COTS application[17] that bridges between HLA and DIS communications standards for audio was selected and integrated into the various simulation facilities that have been integrated into VAST-RT. The design uses HLA for inter-facility communications while the existing DIS interfaces that are used for intra-facility communications are not significantly modified.

Finally, VAST-RT video distribution requirements are substantially different from the real-time simulation data and audio communications requirements. The real-time data and audio communications are used to interface the simulation participants and the simulation components that they are interacting with, while video is primarily viewed and recorded for the benefit of the human-factors researcher. Such video recordings are used to evaluate how specific participants in the simulation respond to off-normal situations or man-machine interfaces. Since video does not have to be distributed in the same manner as the real-time data or audio, the transmission and collection of video is managed with an architecture that is separate from the other data streams.

The VAST-RT project currently relies on the existing video recording capabilities in each of the HITL facilities. However, the VAST-RT system architecture has provided time synchronization tools to ensure that the existing recording systems have access to a common time base. Time base synchronization is done in two ways. First is the synchronization of simulator system clocks by the use of the Network Time Protocol (NTP). In this case, the time-code generators of these simulators are linked to the simulator host computers and are thus synchronized to the common system time. The second method is to use a time base generator that provides both a master source for NTP but also IRIG-B time-code, which can be locally distributed within a facility via physical cabling or distributed through the audio communications system since its signal properties are consistent with audio distribution. At some point in the future, the VAST-RT project will evaluate the need for the digital recording and playback of video information. The digital recording and playback tools will be required to be compatible with the existing VAST-RT time-code synchronization tools so that the researcher can compare the video data with the other recorded data.

Conclusions

The VAST-RT system architecture was developed to integrate a number of human-in-the-loop flight simulation and ATM simulation systems into a multi-fidelity simulation environment that can test revolutionary concepts for increasing the capacity of the National Airspace. The design of the system architecture has focused on overcoming the technical barriers to transmitting data between simulation systems and allowing for the cost-effective, rapid integration of new simulation systems into the distributed simulation. Finally, the architecture has already implemented a minimum set of objects and attributes to represent flights in the National Airspace,

and it also has the flexibility and capacity to expand these data models to support future ATM concepts.

Acknowledgements

The authors would like to thank the entire VAST-RT Simulation Development Team of Northrop Grumman Technical Services Inc., including Carla Ingram, Jeff Hernandez, Ghislain Saillant, Dr. Srba Jovic, and T. Martin Pethtel for their hard work and the many insightful discussions on simulation technologies. The authors would also like to thank Bill Cleveland, Leighton Quon, and Debbi Ballinger of NASA Ames Research Center and Rod Ketchum of the FAA for their contributions to the VAST-RT project. Work supported by NASA contract number NAS2-98084.

Glossary of Acronyms

ADRS	Aeronautical Datalink and Radar Simulator
ATC	Air Traffic Control
ATG	VAST-RT Airspace Traffic Generator
ATM	Air Traffic Management
COTS	Commercial Off The Shelf
DIS	Distributed Interactive Simulation
DMSO	Defense Modeling and Simulation Office
DoD	Department of Defense
DST	Decision Support Tool
FAA	Federal Aviation Agency
FED	Federation Execution Details file
FFC	FutureFlight Central
FOM	Federation Object Model
HDL	HLA Data Logger
HITL	Human-in-the-Loop
HLA	High Level Architecture
IPC	Inter-Process Communications
IRIG-B	Inter-Range Instrumentation Group - B
NAS	National Airspace System
NASA	National Aeronautics and Space Administration
NTP	Network Time Protocol
OHM	Ownership Handoff Manager
OMDT	Object Model Development Tool
RTI	Run Time Infrastructure
SIMNET	Simulator Networking Project
SMI	Simulation Management Interface
SSH	Secure Shell
UIT	User Interface Toolbox
VAMS	Virtual Airspace Modeling and Simulation
VAST-RT	Virtual Airspace Simulation Technology Real-Time
XML	Extensible Markup Language

[1] B. T. Sullivan and S. J. Malsom, "Development of a Real-Time Virtual Airspace Simulation Capability for Air Traffic Management Research," AIAA paper 2002-4592.

[2] Department of Defense, Defense Modeling and Simulation Office, "HLA RTI-1.3 Next Generation Programmer's Guide," Chapter 1.

[3] W. E. Austin, Lt. Cdr. D. Buchy, R. Perry, M. Deschenes, C. Wray, "Real-Time Joint Modeling and Simulation System Prototype (RTJMASS Prototype) Use in Virtual Strike Warfare Environment (VSWE-7) Exercise," AIAA paper 2001-4368.

[4] W. Huiskamp, H. Jense, H. Janssen, "An HLA Based Flight Simulation Architecture," AIAA paper 2000-4401.

[5] J. M. Brennan and R. Simons, "The Army Experiment IV (AE4) Simulation/C4I Experiments," AIAA paper 98-4168.

[6] R. Bertin, "Lessons from a Theater-Wide, Distributed, High Fidelity, Visual Simulation Exercise," AIAA paper 2001-4128.

[7] D. N. Sweet, V. Manikonda, J. S. Aronson, K. Roth, M. Blake, "Fast-Time Simulation System for Analysis of Advanced Air Transportation Concepts," AIAA paper 2002-4593.

[8] F. Kuhl, R. Weatherly, J. Dahmann, Creating Computer Simulation Systems, Prentice Hall PTR (1999).

[9] IEEE Computer Society, "IEEE Recommended Practice for High Level Architecture (HLA) Federation Development and Execution Process (FEDEP)," IEEE Standard 1516.3-2003 (2003).

[10] T. Prevot, E. Palmer, N. Smith, and T. Callantine, "A Multi-Fidelity Simulation Environment for Human-in-the-Loop Studies of Distributed Air Ground Traffic Management," AIAA paper 2002-4679.

[11] Department of Defense, Defense Modeling and Simulation Office, "High Level Architecture (HLA) Object Model Development Tool (OMDT) User's Guide," Version 1.3, June 1998.

[12] N. Dorighi and B. Sullivan, "FutureFlight Central: A Revolutionary Air Traffic Control Tower Simulation Facility," FFC website: http://ffc.arc.nasa.gov.

[13] Department of Defense, Defense Modeling and Simulation Office, "HLA RTI-1.3 Next Generation Programmer's Guide," Chapter 8.

[14] E. Yourdon, Object-Oriented Systems Design, an Integrated Approach, Prentice Hall PTR (1994).

[15] J. Connell and L. Shafer, Object-Oriented Rapid Prototyping, Prentice Hall PTR (1995).

[16] Telestra User Guide, Rev. F, Advanced Simulation Technology Inc., Herndon, Virginia (2004)

[17] ModIOS Voice Communicator, available from General Dynamic Decision Systems.