Proceedings of the 2005 Winter Simulation Conference
M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds.

ADDRESSING COMPLEXITY USING DISTRIBUTED SIMULATION: A CASE STUDY IN SPACEPORT MODELING

Jaebok Park
Reinaldo Moraga
Luis Rabelo
Jeffrey Dawson
Mario Marin
Jose Sepulveda

Center for NASA Simulation Research
Department of Industrial Engineering and Management Systems,
College of Engineering and Computer Science
University of Central Florida
4000 Central Florida Blvd.,
Orlando, FL 32816, USA.

ABSTRACT

As the size, complexity, and functionality of systems to model and simulate continue to increase, benefits such as interoperability and reusability enabled by distributed discrete-event simulation are of interest, especially for distributed manufacturing and enterprise engineering. The High Level Architecture (HLA), a standard distributed simulation environment, is one technology that enables the interconnection of distributed model components. Many applications in industry are developed by a variety of Commercial Off-The-Shelf (COTS) simulation tools, which require some form of gateway to integrate the models into the HLA component-based simulation. This paper summarizes a study conducted to integrate COTS simulation models using gateway tools and visualization of the system states running as part of other simulation models under HLA. The study focused on the prototype of a virtual engineering environment, called the Virtual Test Bed, designed to analyze operations of current and future space vehicles, spaceports, and ranges as a distributed simulation environment.

1 INTRODUCTION

Modeling and analysis efforts rise significantly when dealing with a complex system. In general, a complex system is a nonlinear system of systems whose interactions bring together interesting emergent properties that are difficult to visualize or study by using the traditional method of decomposition. While parallel and distributed discrete event simulation has been an active area of research for more than thirty years, researchers have until recently focused almost exclusively on fast execution of process- and event-oriented models of discrete-event simulations. The advances in this field suggest that distributed simulation may play an important role in modeling complex systems for the analysis of these emergent properties.

Space vehicle ground operations processing as well as ascent and decent phases are complex systems whose interactions give rise to the appearance of emergent properties. A Virtual Test Bed (VTB) has been designed for this case as a simulation architecture to facilitate the integrated execution of different simulation models with other supporting non-simulation applications; as such the architecture must deal with issues related to the coordination of different hardware platforms and components and different software components. The objective of the VTB is to provide a collaborative computing environment that supports the creation, execution, and reuse of simulations that are capable of integrating multidisciplinary models representing the elements of launch, ranges, and spaceport operations in order to assist with cost analysis, flow optimization, and other important decision making factors. The High Level Architecture (HLA) is used as a distributed simulation framework in the VTB architecture for integration of both current and future simulation models. In general, simulation languages/packages may have special areas of use, distinct advanced features, and require specific computing environments such as operating systems (OS), external application interfaces, and scripting languages. These characteristics of the modeling languages may impose difficulties when attempting to seamlessly integrate them with other simulation modeling languages/packages.

Since the HLA was developed for military simulation models, this has become its main area of use. Although the HLA does not mandate the use of any specific software – it is designed to incorporate new technologies as they develop over time – currently the only supporting interfaces available constrain the applicable program languages to C++, Java and Ada. This becomes a problem when interconnecting simulation models in the VTB, many of which are developed by COTS simulation packages such as Arena, AnyLogic, or Synchronous Parallel Environment for Emulation and Discrete-Event Simulation (SPEEDES). In addition, the future models may need to be developed using COTS simulation tools and non-simulation (supporting) tools, which provide advanced functionalities difficult to find elsewhere (e.g., Calpuff and ArcGis in VTB), and additional tools that are necessary in the process of model development such as input/output analyzers, process optimization applications, and visualization software. Also, many COTS simulation packages do not expose their internal data structure or time advance mechanism with an external interface, both of which are required to interoperate with the HLA-RTI, and the interface programming languages need to be among the RTI supporting languages such as C++, Java, or Ada.

This paper focuses on the VTB environment and, particularly, on its interfaces among participating simulation models. In order to overcome the restrictions imposed by the HLA-RTI interface for using COTS tools, several approaches and their implementations were researched. Some examples of such implementations that link COTS Simulation Packages to the HLA are Arena/ProModel (Charles and Frank 2000), AnyLogic (Borshchev, Karpov, and Kharitonov 2002), SLX (Strassburger 1999; Strassburger, Schulze, Klein, and Henriksen 1998), Matlab (Pawletta, Drewelow, and Pawletta 2000), and MODSIM III (Johnson 1999), in addition to the SPEEDES HLA gateway, the Distributed Manufacturing Simulation Adapter (DMS Adapter) with Arena, and the AnyLogic HLA support module. All these approaches provide a solution for the HLA interoperability, but sometimes they cause compatibility issues. For instance, the Manufacturing Simulation Adapter (DMS Adapter) provides a variant of the Federation Object Model (FOM) in Extensible Markup Language (XML) format, which is a format that enables the simulation model to have extended data types, flexibility in individual document structure and format, ease of creation, parsing, interpretation, display by COTS tools, and semantic validation of the file. However when it is required to integrate models written in simulation languages/packages which are using the DMS Adapter and other HLA interoperability tools, the object classes or the interaction classes that are referenced may not be compatible in format. Therefore the development of additional components to translate data formats, including data structure and semantics of attributes, is necessary to make the FOMs compatible.

In addition to the several issues exposed previously, the visualization of system components' interactions and remote systems states in distributed simulation is an important issue, especially in geographically distributed simulation systems like the Virtual Test Bed (VTB). Visualization as a part of a simulation system provides certain insights into the complex dynamics of the system that cannot be obtained using other analysis techniques. Visualization helps the modeler, the decision-maker, and non-technical users to gain better understanding of the modeled systems. In the HLA-based distributed simulation, however, it is difficult, if not impossible, to provide the same level of insight to the user as the COTS visualization tools currently available. This is mainly because the COTS visualization tools are integrated into their simulation engine or are deigned to support a stand-alone simulation execution instead of a distributed simulation. In a distributed simulation environment, although geographically dispersed simulation models may have their own visualization environments, it becomes difficult to provide a comprehensive, global presentation of a distributed simulation system. In order to support an effective decision-making process, informative visualization coupled with distributed simulation models are essential tools when dealing with a large and complex distributed simulation such as space shuttle processing operation models, supply chain simulations, or enterprise engineering models. Therefore, there is a need to have a visual representation of distributed simulations in a single visual display in order to provide comprehensive insight of the whole distributed simulation.

2 VIRTUAL TEST BED: REVIEW OF RELATED TECHNOLOGIES

According to Barth (2002), "Spaceport technologies must employ a lifecycle 'system of systems' concept in which major spaceport systems – launch vehicle processing systems, payload processing systems, landing and recovery systems, and range systems – are designed concurrently with the flight vehicle systems and flight crew systems." The goal is to develop a VTB that can host models representing the systems and elements of a spaceport. These models will work together on the VTB in an integrated fashion, synthesizing into a holistic view and becoming a Virtual Spaceport. A Virtual Spaceport will allow for an intelligent visualization of the entire spaceport concept and the implementation of knowledge management strategies. Details of initial stages of the VTB development, related to concepts and the architecture, can be found in Compton et al. (2003), Sepulveda et al. (2004b); and Sepulveda et al. (2004).

In order to accomplish the goals of the VTB framework, several important aspects are required: (1) *Real-Time Visualization*, - which allows distributed users to collaborate using VTB; (2) *Knowledge and Information Repository* - a repository for storing data, software, object models and lessons learned, so that new exercises or scenarios or tests can be readily constructed; (3) *Integration Environment* - a suite of tools for integrating models, visualizing, planning, executing, collecting data, analyzing, and reviewing scenarios; and (4) *Flexible and Evolving Architecture* – the VTB needs the ability to flexibly reconfigure resources to meet new requirements (see Figure 1).

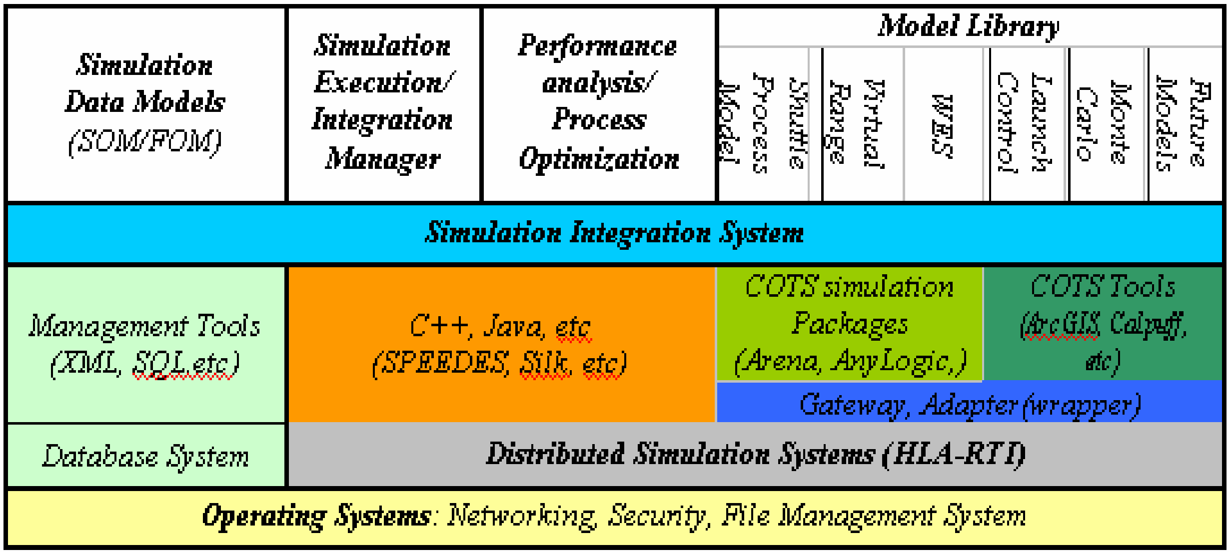


Figure 1. VTB System Architecture

3 HIGH LEVEL ARCHITECTURE

The HLA was introduced by the Defense Modeling and Simulation Office (DMSO) of the Department of Defense (DoD) in 1996 and was accepted as an IEEE standard for distributed simulation – IEEE 1516 – in September 2000. The HLA defines terms used in the context of distributed simulation as follows. A *federate* is a member of a federation; a federate refers to an actual simulation, and the role in a distributed simulation is defined in its Simulation Object Model (SOM). A *federation* is a set of simulations (federates) interconnected through the RTI; a Federation Object Model (FOM) and its supporting infrastructure are used to form a large model to achieve certain objectives. Interoperability between federates is achieved by three major components: *HLA rules,* which describe federation and federate responsibilities; the *Run Time Infrastructure,* which coordinates the local simulation time managed by each federate with the global simulation time in a federation and controls the data transfer; and the *Object Model Template* (OMT) which defines data structure, the format of the federates (SOM), and the common information in federation (FOM). These three components are described in detail in Judith et al. (1998).

The Run Time Infrastructure (RTI) is a software implementation of the HLA Interface Specification, which defines the common interfaces for distributed simulation systems during the execution of an HLA simulation. The RTI is comprised of the following three components: the RTI Executive process (RtiExec), the Federation Executive process (FedExec), and the libRTI library. The FedExec manages the process of joining federates and resigning the federation and facilitates data exchange between participating federates. A FedExec process is created by the RTI when the first federate successfully joins the federation and is eventually destroyed by the RTI when the last federate resigns from the federation. The RtiExec manages the creation and destruction of multiple federation executions within a network. The RtiExec ensures each FedExec has a unique identification and directs the joining of federates to the appropriate federation. Although more than one federation can be running under the RtiExec, communication between federations is not possible. The libRTI library extends RTI services to the federate developer. It enables the federate to access RTI services specified in the Interface Specification by the RTIambassador and FederateAmbassador. Data exchange between federates in a federation occurs only through the RTI by the HLA rules and is accomplished by means of the RTIambassador and FederateAmbassador. The libRTI includes both the RTIambassador and the FederateAmbassador class. Passing information from a federate to the RTI is accomplished by calling services in the RTIambassador.

4 DISCRETE-EVENT SIMULATION LANGUAGES AND PACKAGES

A discrete-event simulation can be built using either COTS simulation packages or a general-purpose simulation language. For this application, Arena, AnyLogic and SPEEDES were selected based on availability, the number of existing models already written in the modeling tool to be integrated into the VTB, and the requirements of the VTB simulation system. SPEEDES is a general-purpose, discrete-event, distributed simulation engine and modeling framework for building complex and interoperable parallel/distributed simulations in C++. It was developed at the Jet Propulsion Laboratory by Dr. Jeff Steinman (Bailey, McGraw, Steinman, and Wong 2001, Metron 2005, Steinman and Wong 2003). SPEEDES provides a parallel processing framework that enables the integration of objects distributed across multiple processors to increase simulation speed. It supports multiple time management algorithms such as the sequential algorithm, time-driven algorithm, Time Warp algorithm, and Breathing Time Warp algorithm, which is a combination of the Time Warp and Time Bucket algorithms. In optimistic time management, an event can be processed even if it may not be the next event to be processed in ascending time order while maintaining repeatability and causality by using "rollback" techniques, and in conservative time management, an event will not be processed until it is known that there is no possibility of an event arriving in the past relative to the simulation time. Rollbacks restore state variables and retract events scheduled during the simulation time period that needs to be rolled-back.

The SPEEDES modeling framework is comprised of four fundamental components that provide the basic functionalities needed for event-based simulation modeling: (1) object manager, (2) simulation object, (3) events, and (4) messages (Bailey et al. 2001; Fullford 1999). When the simulation is initialized, one simulation object manager for each simulation object type is created on each node. The object manager controls the creation, initialization and destruction of simulation objects, and the decomposition of the objects, which assigns simulation objects to nodes. Decomposition of objects can be done by automatic methods (block, scatter) or a user-defined manner. Block decomposition distributes the simulation objects to nodes evenly. Scatter decomposition distributes the simulation objects such that simulation objects with consecutive IDs are located on different consecutive nodes. Simulation objects are the fundamental concept behind the SPEEDES modeling framework, which represent entities in the simulation system. The modeling framework consists of a set of attributes which maintain the state of the object and the methods that define the activities of the object. The type of attributes may be primitive base types from C++ or rollback types if the attribute is state sensitive. The simulation object class in SPEEDES provides the primitive functions to schedule events, process event handlers, and responses to interactions. Simulation object events are a part of a simulation object and are used to change the values of the state variables in simulation objects. They are defined as public methods, the most accessible level, so that any simulation object in the simulation may schedule the simulation object events. SPEEDES provides a set of macros that turn methods on simulation objects into events, plugs these events into the SPEEDES framework, and generates functions for scheduling these events. To make scheduling events convenient, the macros automatically build a global function for each event defined, which users can use to invoke each event. When an event schedules a new event, a message is created by SPEEDES with header information that defines the type of event, simulation time, type of simulation object, and its local ID. The header information is used to create a corresponding event object at the destination node.

5 VISUALIZATION IN DISTRIBUTED SIMULATION

"Visualization is the process of transforming data, information, and knowledge into visual form making use of humans' natural visual capabilities. With effective visual interfaces we can interact with large volumes of data rapidly and effectively to discover hidden characteristics, patterns, and trends." (Nahum et al. 1998).

The use of visualization in simulation is justified because it provides an understanding of the complex dynamics of a system that would otherwise be impossible to obtain (Law and Kelton 2000; Rohrer 2000; Steven and Sisti 1994; Swider, Bauer and Schuppe 1994). It is a reliable method of presenting concepts of model dynamics to the end users who may not be aware of the technical details of the model. Visualization is also a tool to verify the correctness of a model (Steven D. Farr et al. 1994). Two types of visualizations are required in the context of VTB distributed simulation. First, a visualization of data and/or the specialized functions is an essential part of COTS tools, but the tools do not support any type of simulation concepts. In order to integrate the visualization tool into the VTB, a federate was created that interacts with both the RTI and the tool's external interface, which may be in such formats as the Component Object Model (COM) or Dynamic Link Library (DLL). Second, a simulation engine includes a set of integrated animation facilities to display the state of the system being simulated, which may allow the user to interact with the model. It does not, however, support any function for visualization of a remote federate in the federation. To address this problem, a COTS simulation package was used to include the state of a remote federate in a local federate. As an example, in order to effectively visualize a region and to accurately calculate population metrics associated with that region, ArcView with Spatial Analyst and the LandScan Global Population Database were used. In Figure 2, an example of VTB visualization is shown when a toxic envelope is covering a certain area of Central Florida. All the risk metrics are provided by the simulation, and the integration of different models working together is exemplified.

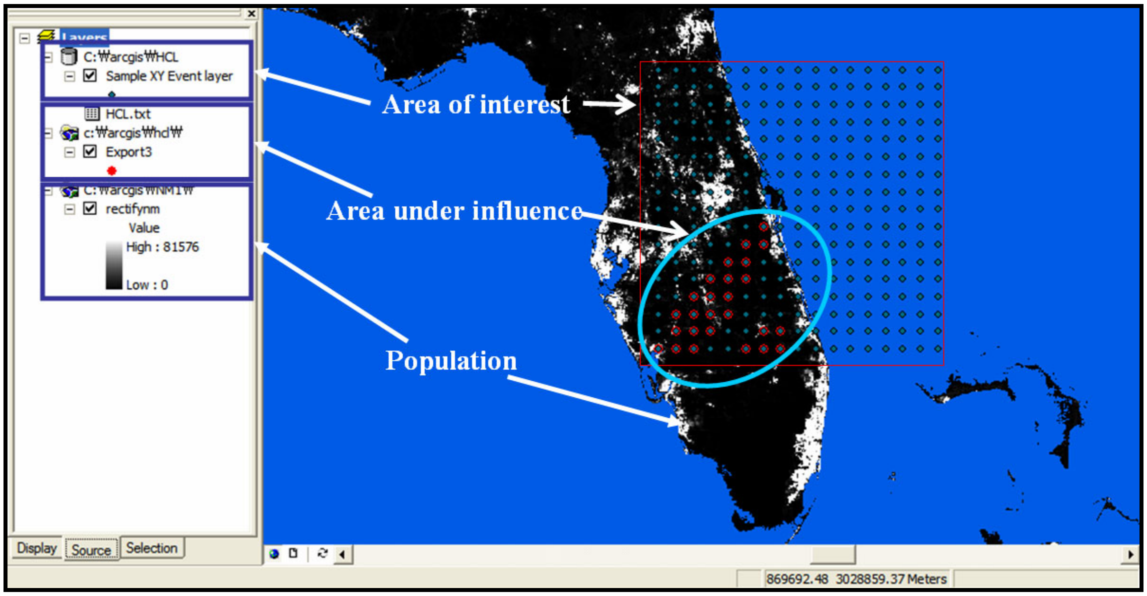


Figure 2. Calculation of Population at Risk

6 CASE STUDY

The VTB platform was tested by using several studies, but in this paper just one of them will be presented.

The Virtual Range toxicity model is an integrated set of software packages that exchange information in order to calculate the Expectation of Casualty as a result of gas dispersion when an accident ending in loss of vehicle affects the Space Shuttle within 120 seconds of liftoff. Among these software packages is a MonteCarlo simulation, a gas dispersion model (Calpuff), a population model (LandScan), a Geographical Information System (ArcView and ArcGIS Spatial Analyst), and access to weather data and flight path information. The VR toxicity model was divided into two simulation models (federates), the Monte Carlo federate and the VR federate (see Figure 3). The Monte Carlo federate simulates the time of accident, which is determined by the cumulative probability of an accident occurring in ten different stages during a launch. Each of these stages has a different duration and chance of accident. Once the stage is determined, the time of accident is fixed by equal chance within the stage. Based upon the time of accident, the Monte Carlo federate references coordinates for the path of orbiter and determines the volume of remaining pollutants from the existing model data file.

For each request of a simulated launch from other federates, the Monte Carlo federate determines whether the launch is successful or whether a simulated disaster occurs. If there is a successful launch, the Monte Carlo federate sends an interaction indicating "successful launch" through the RTI; if there is an accident, it sends an interaction which includes the location (latitude, longitude, and altitude) and the concentration of toxicant released to the federation.

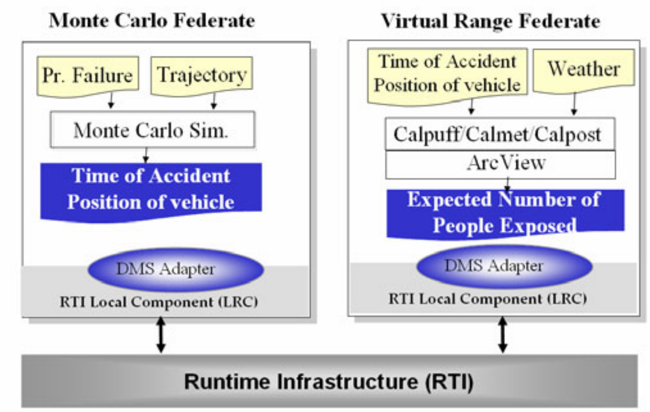


Figure 3. The Monte Carlo Federate and The VR Federate Architecture

All other components of the Virtual Range toxicity model are included in the VR federate and they work the same as before. Figure 3 shows the architectures of the Monte Carlo federate and the VR federate which are adapted to the HLA distributed simulation.

The advantages of this model are twofold. First, we can simulate many different shuttles and orbiters which may have different probabilities of failure without changing the Virtual Range toxicity model. Since the inputs to the Monte Carlo federate depend on the property of the shuttle, each Monte Carlo federate can be built in such a way that a Monte Carlo federate represents a shuttle as a component model. The component model can be integrated into the VTB as necessary. Second, the process of each federate can be initiated by other simulations such as the LaunchPad federate or the Mission Control federate, and the intermediate information can be utilized by other federates in the VTB. In addition, we are using the RTI synchronization to coordinate simulators and animations of the simulations as depicted in Figure 4.

The Web-based Weather Expert System (WES) is a critical module of the Virtual Test Bed development to support go/no-go decisions for Space Shuttle operations in the intelligent Launch and Range Operations (ILRO) program of NASA (Rajkumar and Bardina 2003). The weather rules characterize certain aspects of the environment related to the launch or landing site, the time of the day, the pad or runway conditions, the mission durations, the runway equipment, and the landing type. Expert system rules are derived from weather contingency rules, which were developed over several years by NASA. Backward chaining, a goal-directed inference method, is adopted to the system rules because a particular consequence or goal clause is evaluated first and then chained backward through the rules. Once a rule is satisfied or true, then that particular rule is fired and the decision is expressed. The expert system is continuously verifying the rules against the past one-hour weather conditions and the decisions are made. In the weather expert system, the user interface is automated in such a way that the inputs to the expert system are downloaded and fed to the system in a periodic manner. There is no need for human intervention in the expert system, and decisions for launch are automatically displayed as a web page. The weather expert system is based on Java technology and Web enabled. The real time weather data is obtained from different Federal weather monitoring agencies. Images and other types of data are downloaded and then processed, extracted and converted to suitable numerical values. The image processed data is stored in an image file and other numerical values are stored in a weather file. The above mentioned files constitute the inputs for the expert system.

We identified the weather expert system (WES) as an essential component in the Virtual Range. In order to integrate the WES into the VR, we decided to convert the Socket based the VR architecture (Sepulveda et al. 2004) to the HLA. The integration of the WES into the VR infrastructure was accomplished using the RTI APIs exposed through the Java Binding. The WES can pass information about the decision and the different weather information to the VR and its federates. The integration of the WES into an HLA federation includes developing a Federation Object Model (FOM) based on the information that needs to be exchanged, implementation of a FederateAmbassador, and adding some adaptation code into WES.

First, based on the objective of the federation, the shared information needed from WES consists of mainly the launch decision and the processed data which are collected from various weather sources and then incorporated into the decision algorithm. Second, the interface was implemented using the RTI Java Binding which is a thin layer of C++ code that exposes the native C++ API of the RTI to Java applications through the Java Native Interface (JNI) (DMSO 2001). In addition, since the original WES is running on a Tomcat Web Server and the users interact with the application by using a Web browser, it is difficult to make WES as a federate within the current VTB architecture. Therefore, most classes in the original WES were converted into Java Applications without altering the main algorithms, which is a simple change in the Java interface. In addition to the conversion, we have created a user interface to initiate the operations of acquiring source data from various weather sources, processing, invoking decision-making processes, and joining to the VTB federation as a WES federate (see Figure 5).

The WES federates publish (updates) near real-time weather information and go/no-go decisions requested by the federation. The weather information and go/no-go decisions will be used not only in the VR as weather factors for CALPUFF models but also in the Mission Control model to decide a launch weather decision.

7 CONCLUSION

In this paper, the prototype implementation of the VTB presented successfully demonstrates that the VTB architecture can be used to analyze more complex, larger operations and provide associated solutions such as structural process or cost optimization. There are a number of challenges still remaining in this prototype: implementation regarding the integration of the models in the simulation model library using scripts and the design of a graphical user interface that supports the distributed modeling processes such as configuration and execution of a federation over a network, collecting and saving the outputs to a data.

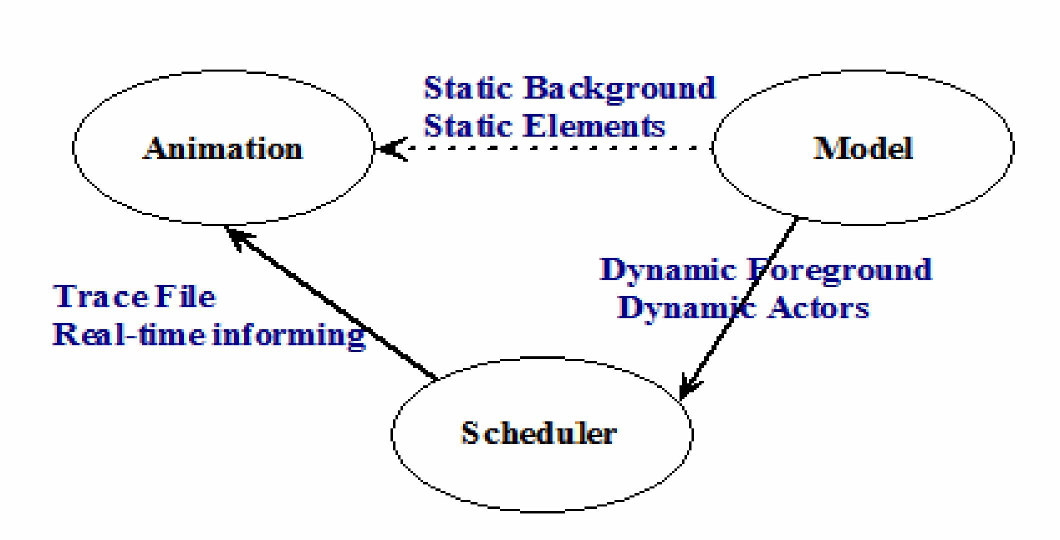
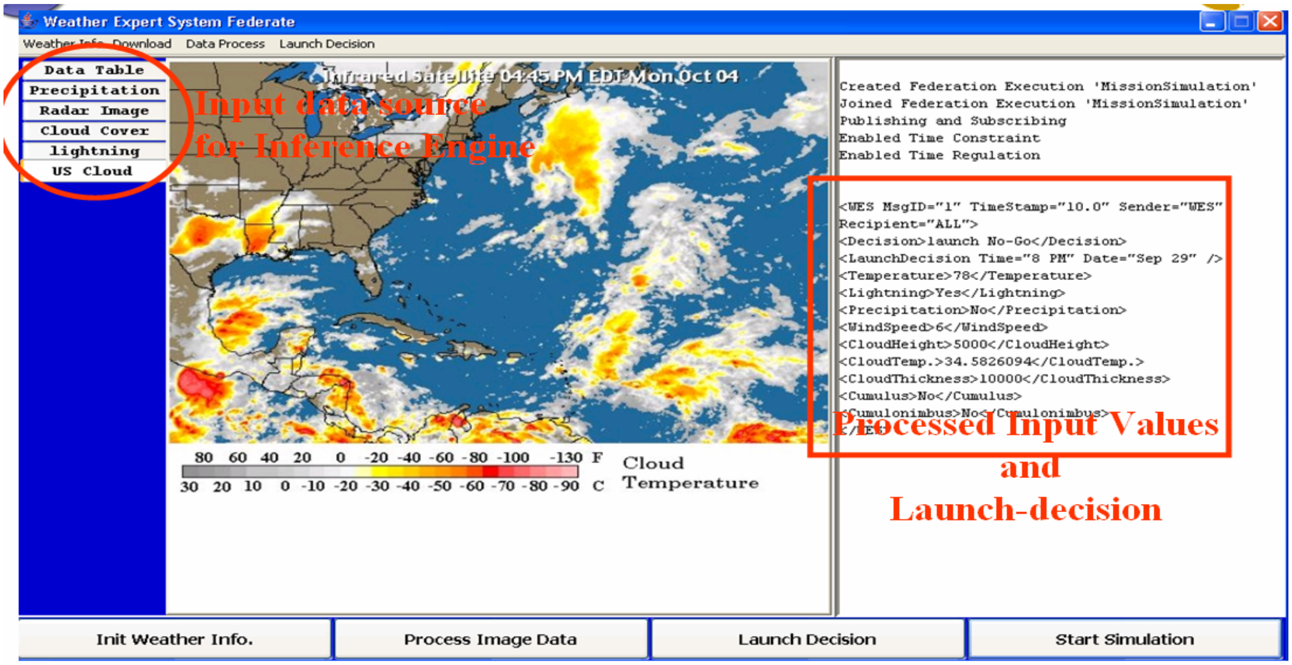


Figure 4. Model-Animator-Scheduler Paradigm

In order to present more realistic operations of space vehicles, spaceports, and ranges, the VTB needs to integrate simulation and non-simulation components such as cost analysis, human resources, planned flight manifests, optimization of mission-related operations, telemetry data collected from in-flight sensors or ground monitoring equipment, or input from live entities.

ACKNOWLEDGMENTS

We wish to thank Jeppie Compton from NASA Kennedy Space Center, Dr. Jorge Bardina from NASA Ames Reserach Center, and Frank Riddick from NIST for sharing their experience and knowledge with us.



```
Created Federation Execution 'MissionSimulation'
Joined Federation Execution 'MissionSimulation'
Publishing and Subscribing
Enabled Time Constraint
Enabled Time Regulation

<WES MsgID="1" TimeStamp="10.0" Sender="WES"
Recipient="ALL">
<Decision>launch No-Go</Decision>
<LaunchDecision Time="8 PM" Date="Sep 29" />
<Temperature>78</Temperature>
<Lightning>Yes</Lightning>
<Precipitation>No</Precipitation>
<WindSpeed>6</WindSpeed>
<CloudHeight>5000</CloudHeight>
<CloudTemp.>34.5826094</CloudTemp.>
<CloudThickness>10000</CloudThickness>
<Cumulus>No</Cumulus>
<Cumulonimbus>No</Cumulonimbus>
</WES>
```

Figure 5: An Implementation of the WES Federate

REFERENCES

Bailey, C. A., McGraw, R. M., Steinman, J. S., and Wong, J. 2001. *SPEEDES*: A brief overview. *Proceedings of the SPIE - The International Society for Optical Engineering, 4367,* 190-201.

Bardina, J. 2001. *Intelligent Launch and Range Operations. NASA ARC.*

Barth, T 2002. Found in space. *IIE Solutions, No. 4*.

Borshchev, A., Karpov, Y., and Kharitonov, V. 2002. Distributed simulation of hybrid systems with AnyLogic and HLA. *Future Generation Computer Systems, 18,* 829-839.

Cates, G. R., Steele, M. J., Mollaghasemi, M., and Rabadi, G. 2002. Modeling the space shuttle. In *Proceedings of the 2002 Winter Simulation Conference,* ed. E. Yücesan, C.-H. Chen, J.L. Snowdon, J.M. Charnes, 754-762.

Charles, M. and Frank, R. 2000. Simulation in the international IMS MISSION project: the IMS MISSION architecture for distributed manufacturing simulation. . *In Proceedings of the 2000 Winter Simulation Conference,* ed. *R. Barton. et al.,* 1539-1548.

Compton, J., Sepulveda, J., and Rabelo, L. C. 2003. *Integration of the Virtual Test Bed and Virtual Range Assessment, Technical Report*, NASA Kennedy Space Center.

DMSO. 1998. *High Level Architecture Run-Time Infrastructure Programmer's Guide 1.3 Version 5.* Defense Modeling and Simulation Office Available via: https://www.dmso.mil/public/[accessed January 4, 2005].

DMSO. 2001. The RTI1.3-NG Java Binding. DMSO Available via https://www.dmso.mil/public/[accessed January 4, 2005].

Fullford, D. 1999. A Federation Management Tool: Using the Management Object Model (MOM) to Manage, Monitor, and Control an HLA Federation. *Spring Simulation lnteroperability Workshop*.

Johnson, G. D. 1999. Networked simulation with HLA and MODSIM III. *In Proceedings of the 1999 Winter Simulation Conference, - A Bridge to the Future' (Cat.No.99CH37038), ed. P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 1065-1070.*

Joines, J. A. and Roberts, S. D. 1995. Design of object-oriented simulations in C++. *1995 Winter Simulation Conference Proceedings (Cat.No.95CB35865),* 82-89.

Judith, D., Richard, M. F., and Richard, M. W. (1998). The DoD high level architecture: an update. *In proceedings of the 1998 Winter Simulation Conference (Cat.No.98CH36274), ed., D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannan, 797-804.*

Law, A. M. and Kelton, W. D. 2000. *Simulation modeling and analysis*. (3rd ed ed.) Boston: McGraw-Hill.

Metron, I. 2005. *SPEEDES, Users Guide 2003* . Metron Corporation,San Diego,California.

Nahum, G., Stephen, G. E., and Stuart, C. 1998. Information visualization. *Interactions., Vol. 5,* pp. 9-15.

Pawletta, S., Drewelow, W., and Pawletta, T. 2000. HLA-based simulation within an interactive engineering environment. *Fourth IEEE International Workshop on Distributed Simulation and Real-Time Applications*, pp. 97-102.

Rajkumar, T. and Bardina, J. E. 2003. Web-based weather expert system (WES) for Space Shuttle launch. *SMC'03 Conference Proceedings.2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance,* 5040-5045.

Rohrer, M. W. 2000. Seeing is believing: the importance of visualization in manufacturing simulation. In *Proceedings* of the *2000 Winter Simulation Conference, ed. R. Barton. et al.,* 1211-1216.

Rossetti, M. D., Aylor, B., Jacoby, R., Prorock, A., and White, A. Simfone. 2000 . An object-oriented simulation framework. In *Proceedings* of the *2000 Winter Simulation Conference, ed. R. Barton. et al.,* 1855-1864.

Sepulveda, J., Rabelo, L., Park, J., Gruber, F., and Martinez, O. 2004. Factors Affecting the Expectation of Casualties in the Virtual Range Toxicity Model. *In Proceedings of the 2004 Winter Simulation Conference, ed, N.M. Steiger, M. E. Kuhl, 685-692.*

Sepulveda, J., Rabelo, L., Park, J., Riddick, F., and Peaden, C. 2004. Implementing the High Level Architecture in the Virtual Test Bed. *In Proceedings of the 2004 Winter Simulation Conference, N.M. Steiger, M. E. Kuhl, 380-387.*

Steinman, J. S. and Wong, J. W. 2003. The SPEEDES persistence framework and the standard simulation architecture. *Proceedings Seventeenth Workshop on Parallel and Distributed Simulation,* 11-20.

Steven D. Farr and Alex F.Sisti 1994. Visualization of General Purpose Simulation Results. Available via http://www.rl.af.mil/tech/papers/ModSim/DU94.html [accessed February 10, 2005].

Strassburger, S. 1999. On the HLA-based coupling of simulation tools. *Modelling and Simulation: A Tool for the Next Millennium.13th European Simulation Multiconference 1999. ESM'99,* 45-51.

Strassburger, S., Schulze, T., Klein, U., and Henriksen, J. O. 1998. Internet-based simulation using off-the-shelf simulation tools and HLA. *In proceedings of the 1998 Winter Simulation Conference, ed., D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannan,,* 1669-1676.

Swider, C. L., Bauer, K. W., Jr., and Schuppe, T. F. 1994. The effective use of animation in simulation model validation. *In proceedings of the 1994 Winter Simulation Conference, ed., J.D. Tew and M.S. Manivannan,* 633-640.

AUTHOR BIOGRAPHIES

JAEBOK PARK has a Ph.D. from the Institute for Simulation & Training at the University of Central Florida. He received a B.S. in Electrical Engineering from Korea Naval Academy in 1988, and a M.S. in Computer Science from Western Illinois University in 1994. His research interests include Distributed Simulation, High Level Architecture and Visualization. His e-mail address is `<ja969179@pegasus.cc.ucf.edu>`.

REINALDO MORAGA is currently a Visiting Assistant Professor in the Industrial Engineering and Management Systems at the University of Central Florida (UCF). He received two B.Sc. degrees in Wood Engineering in 1988 and Industrial Engineering in 1993 from Universidad del Bio Bio, in Chile, and has degrees of M.Sc. and Ph.D. from UCF. Dr. Moraga's dissertation was recognized by IIE with the 2003 Pristker Dissertation Award, third place. His research interests are Simulation, Operations Research, and Metaheuristics. His e-mail address is `<rmoraga@mail.ucf.edu>`.

LUIS RABELO, Ph.D., is an Associate Professor in Industrial Engineering and Management Systems (IEMS) at the University of Central Florida. He received dual degrees in Electrical and Mechanical Engineering from the Technological University of Panama and Master's degrees from the Florida Institute of Technology (Electrical Engineering, 1987) and the University of Missouri-Rolla (Engineering Management, 1988). He received a Ph.D. in Engineering Management from the University of Missouri-Rolla in 1990 where he also did Post-Doctoral work in Nuclear Engineering. He also holds dual MS degrees in Aerospace Systems Engineering & Management from the Massachusetts Institute of Technology (2001). His experience includes Ohio University, BF Goodrich Aerospace, Honeywell Laboratories, NIST, NASA, and MIT. His e-mail address is `<lrabelo@mail.ucf.edu>`.

JEFFREY W. DAWSON is currently a doctoral candidate in the University of Central Florida Department of Industrial Engineering and Management Systems. He has a bachelor of science degree in industrial engineering from the University of Tennessee, and a Master of Business Administration and Master of Science in industrial engineering from the University of Central Florida. He has worked in the nuclear and aerospace industries performing reliability, system safety, design, and cost analyses. His recent research areas include augmented reality, usability, ergonomics, and simulation. His e-mail address is `<Jeffrey@geomix.com>`.

MARIO MARIN received his Masters in Simulation Modeling and Analysis from the University of Central Florida. He is currently pursuing a Ph.D. in Industrial Engineering at the University of Central Florida. His research interests are mainly simulation integration of operations, HLA and XML technologies. His email is `<mmarin@mail.ucf.edu>`.

JOSÉ A. SEPÚLVEDA, Ph.D., P.E., is an Associate Professor in IEMS at the University of Central Florida. He received an Ingeniero Civil Químico degree from the Universidad Santa María, Valparaíso, Chile, and MSIE, MPH, and Ph.D. (Industrial Engineering) degrees from the University of Pittsburgh. Dr. Sepúlveda's major areas of research interest are object-oriented simulation, risk analysis, catastrophe response, measuring and modeling training effectiveness, and applications of industrial engineering and simulation in health care. He has written two books and numerous publications. His e-mail address is `<sepulved@mail.ucf.edu>`.