Research Article

Simulation Modeling of Space Missions Using the High Level Architecture

Luis Rabelo,[1] Serge Sala-Diakanda,[1] John Pastrana,[1] Mario Marin,[1] Sayli Bhide,[1] Oloruntomi Joledo,[1] and Jorge Bardina[2]

[1] *Department of Industrial Engineering and Management Systems, University of Central Florida, Orlando, FL 32816, USA*
[2] *Intelligent Systems Division, NASA Ames Research Center, Moffett Field, CA 94035, USA*

Correspondence should be addressed to Luis Rabelo; lrabelo@mail.ucf.edu

This paper discusses an environment being developed to model a mission of the Space Launch System (SLS) and the Multipurpose Crew Vehicle (MPCV) being launched from Kennedy Space Center (KSC) to the International Space Station (ISS). Several models representing different phases of the mission such as the ground operations processes, engineered systems, and range components such as failure tree, blast, gas dispersion, and debris modeling are explained. These models are built using different simulation paradigms such as continuous, system dynamics, discrete-event, and agent-based simulation modeling. The High Level Architecture (HLA) is the backbone of this distributed simulation. The different design decisions and the information fusion scheme of this unique environment are explained in detail for decision-making. This can also help in the development of exploration missions beyond the International Space Station.

1. Introduction

Distributed simulation plays an important role in modeling complex systems. Space vehicle ground operations processing as well as ascent and decent phases are complex processes whose interactions give rise to the appearance of emergent properties [1–3]. For these cases, a Virtual Test Bed (VTB) was designed as the architecture to facilitate the integrated execution of different simulation models with other supporting nonsimulation applications [4–9].

Our completed initial VTB development efforts (see Section 2) for modeling space shuttle missions and operations at NASA Kennedy Space Center (KSC) are based on the High Level Architecture (HLA) and the run-time infrastructure (RTI). The RTI, a software implementation of the HLA Interface Specification, defines the common interfaces for distributed simulation systems during the execution of the HLA simulation [10–13]. It is the architectural foundation that promotes portability and interoperability. All shared information exchanged during a federation (i.e., a set of simulation models) execution must be passed through the RTI.

The objective of the VTB developments 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 the 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. In general, simulation languages/packages may have special areas of use, distinct advanced features, and require specific computing environments such as operating systems (OSs), 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. As the application of HLA distributed simulation architectures widely spreads to different areas of application, the need for middleware development and/or adapters/controllers for communications becomes necessary [6, 12, 14]. The web can provide additional functionality to the HLA/RTI configurations.

A number of distributed simulation research work have been focused on global cooperation via the web and its architectures [13, 15–20]. Our enhanced VTB approach considers the capabilities and constraints of web-enabled HLA/RTI configurations. Traditionally, vendor specific HLA/RTI implementations and different RTI versions imposed a number of restrictions on distributed simulations interoperability characteristics and services for heterogeneous domains. However, developments of HLA-based web service tools have enabled the introduction of simulation functionalities to heterogeneous users in remote locations in distributed simulation systems architectures [16]. For example, 3D visualization simulation capabilities can be introduced in a distributed simulation environment as a separate heterogeneous platform in a remote location. Tu et al. [18] proposed an HLA web-enabled type architecture to improve federate interoperability and agility within it distributed components. Their architecture developed a web-service bridge and server as an API to the Portico RTI implementation. In addition, other HLA-based web-service architecture implementations developed by researchers grant interoperation between heterogeneous simulation systems as discussed by Tang et al. [16]. According to Tang et al. [16], web-service architecture capability introduces the concept of Service-Oriented Architectures (SOA) which enables HLA-based architectures to deliver federate designs as applications with specific functionalities as a service to end users in remote locations. These developments support the concept of layered architectures.

Our new developments with the VTB are based on a layered approach. The enhanced VTB architecture design approach adopts the benefits of layered architectures and more flexible middleware solutions to achieve a desirable interoperability and scalability distributed simulation platform. Al-Zoubi and Wainer [15] explain that structural rules inherent in many distributed simulation middleware solutions, like HLA/RTI, impose constraints in scalability and interoperability capabilities. In their work the authors propose the RESTful Interoperability Simulation Environment (RISE) architecture for distributed simulation designs in open computing networks like the Web. Their distributed simulation environment allows for better decoupling through middleware HLA/RTI configurations for achieving enhanced scalability of distributed simulation designs. Further, Topcu and Oguztuzun [17] explain how the layered architecture approach to distributed simulation systems separates the user-interface, the simulation main control method, and the HLA-specific federate communication mechanisms for enhancing the system flexibility. Their idea is that designers can develop or implement the different federation components in their programming languages or platforms of choice and to capture the repetitive HLA interface implementation in one layer for design simplicity. The layered approach is taken into consideration in our enhanced VTB configuration for the implementation of the mission of the SLS and MPCV vehicles being launched from KSC to the International Space Station.

This paper expands on lessons learned from our initial developments carried out in order to start the modeling of the Space Launch System (SLS) and the Multipurpose Crew Vehicle (MPCV) being launched from KSC to the International Space Station (ISS). In addition, this paper discusses the different design decisions and the information fusion scheme of the VTB for decision-making that can help in the development of exploration missions beyond the ISS.

2. Initial Efforts and Lessons Learned

Since 2002, we have developed several configurations using the VTB. NASA and the Defense Industry have been our major supporters. The dominant architecture was a centralized one (i.e., centralized RTI Node [13]). Figure 1 illustrates the distributed simulation configuration of the NASA space shuttle launch. The basic federates in this configuration are described as follows.

(1) The Shuttle Model Federate (SMF). This federate is a simulation model written in Arena (http://www.arenasimulation.com/Arena_Home.aspx). SMF was developed by experts in discrete-event simulation and space shuttle operations. SMF simulates the flow of a space shuttle from landing at KSC through its normal processing assembly flow and its launch pad flow [21]. If the mission is a success, this federate displays the Shuttle flying around the earth and returning to KSC followed by landing and repeating the operational cycle. However, if the mission ends up in an accident then the display screen changes and shows the explosion of the shuttle, the date and time of the accident, coordinates where the explosion occurred, and amount of contaminants released into the atmosphere from the shuttle's unused propellants at that location. When the shuttle to be launched to the space reaches the launch pad, a signal is sent through the RTI indicating that the shuttle is ready for launch. This signal is received by the Control Room Model Federate (CRMF), and the shuttle waits for authorization for liftoff.

(2) Control Room Model Federate (CRMF). The Control Room Model Federate (CRMF) is an AnyLogic- (http://www.anylogic.com/) based model which simulates the behavior of the shuttle's hardware/software systems. This model is based on continuous simulation and electronics/Boolean logic. It is developed by experts in continuous simulation and electrical/mechanical and systems/software engineers with experience in the NASA shuttle hardware/software systems. When the shuttle arrives at the launch pad, a message is sent to the CRMF. On receiving the message that the shuttle is ready for launch, the CRMF is activated. CRMF checks for failure in four systems—electronic, electrical, communications, and propulsion systems. After verifying that no delays or scrubs occurred, CRMF waits for a go-ahead signal from the Weather Expert Model Federate (WEMF), and then it authorizes the launch by sending a message through the RTI that the systems in the CRMF are all green and the launch is a "GO."

(3) Weather Expert Model Federate (WEMF). Weather Expert Model Federate (WEMF) is a sophisticated Java-based model. The main functionality of this federate is to show a summary of the weather forecast (updated at specific intervals of time (e.g., minutes, hours)) [22]. The information is collected by
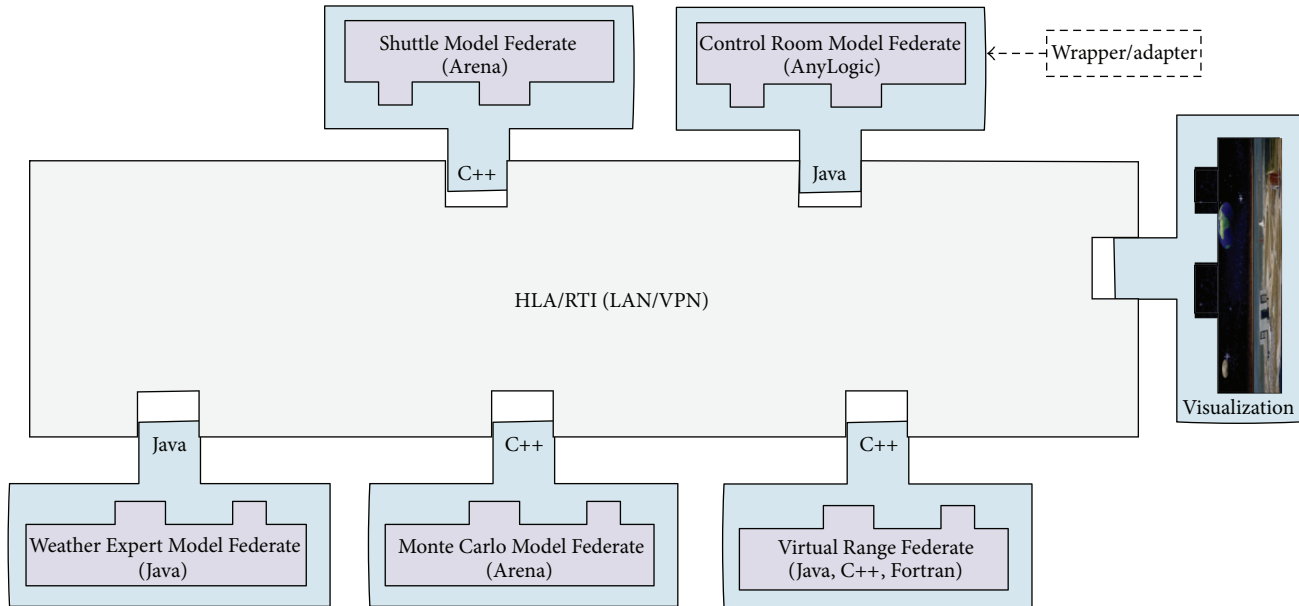
FIGURE 1: Space shuttle distributed simulation configuration using HLA/RTI (local area network (LAN)/virtual private network (VPN)).

Java servlets from different web sites, like the temperature and wind speed from http://weather.noaa.gov/weather (some of these websites are of open access and others are military/classified). The WEMF has several processing systems based on neural networks and wavelets that perform the required analysis of the weather conditions and the images. A set of Java servlets get the data or images from various sources across the US. Information about a 7-day weather forecast is combined with images from the GOES satellite (http://www.goes.noaa.gov/) with daily weather forecasts. Apart from these images, specific weather details like humidity, wind speed, barometric pressure, heat index, and dew point are updated at very short intervals. The US cloud classification is provided by the Naval Postgraduate School at Monterey, California. The lightning data is provided by the National Lightning Detection Network (NLDN) and at every 30 minute interval; details about lightning strikes across the US are updated. The surface temperature contour and surface wind speed are provided by the National Weather Service. The sea state analysis is provided to understand the booster rocket recovery operations. Weather criteria for an emergency landing at the Transoceanic Abort Landing Sites (TALS) are monitored by the National Oceanic and Atmospheric Administration (NOAA) in Spain and North Africa (http://www.srh.noaa.gov/smg/smgwx.htm). The data is downloaded from various agencies in the format of images and numerical data. Once the data is downloaded, the images are processed and specific values are derived for Florida/KSC. The data is used as inputs to an expert system that suggests the shuttle launch decision by GO or NO-GO. The WEMF receives a message from the CRMF that all the systems are a GO. Then WEMF checks if the weather conditions are also a GO and sends a message through RTI for authorizing the launch. Weather information can be accessed by querying WEMF through the RTI at any time.

(4) Monte Carlo Model Federate (MCMF). The Monte Carlo Model Federate (MCMF) is a discrete model developed in Arena (http://www.arenasimulation.com/Arena_Home.aspx). MCMF is notified through the RTI when the simulated shuttle lifts off. It also receives a message from the CRMF that a launch took place. The MCMF then determines if the launch will result in a success or if a disaster will occur by generating random numbers as is done in all MCMF (this is based on historical data and very sophisticated failure models at the subcomponent level developed for the NASA Shuttle; see Figure 2). As per the outcome of the simulation, an appropriate message indicating the success or the accident result is sent through the RTI.

(5) Virtual Range Federate (VRF). The Virtual Range Federate (VRF) is composed of several simulation models and software systems. The simulation models are continuous and model the shuttle trajectory, gas dispersions, and clouds of the huttle fuel systems. The Shuttle trajectory model was provided by the NASA experts in rocket trajectories using MATLAB/Simulink (http://www.mathworks.com/products/matlab/). The gas dispersion and clouds simulation model are a modification of CALPUFF (http://gcmd.nasa.gov/records/CALPUFF.html). CALPUFF is an advanced nonsteady-state meteorological and air quality modeling system developed by Atmospheric Studies Group (ASG—http://www.src.com/calpuff/calpuff1.htm). The model has been adopted by the US Environmental Protection Agency (US EPA) [24–26]. In addition, VRF has an incorporated geographical information system (ArcGIS—http://www.esri.com/software/arcgis) with several databases. This federate displays a counter of the number of launches and a summary of the current weather if the message from the MCMF indicates successful launch. In case an accident results from the MCMF, the VRF activates and
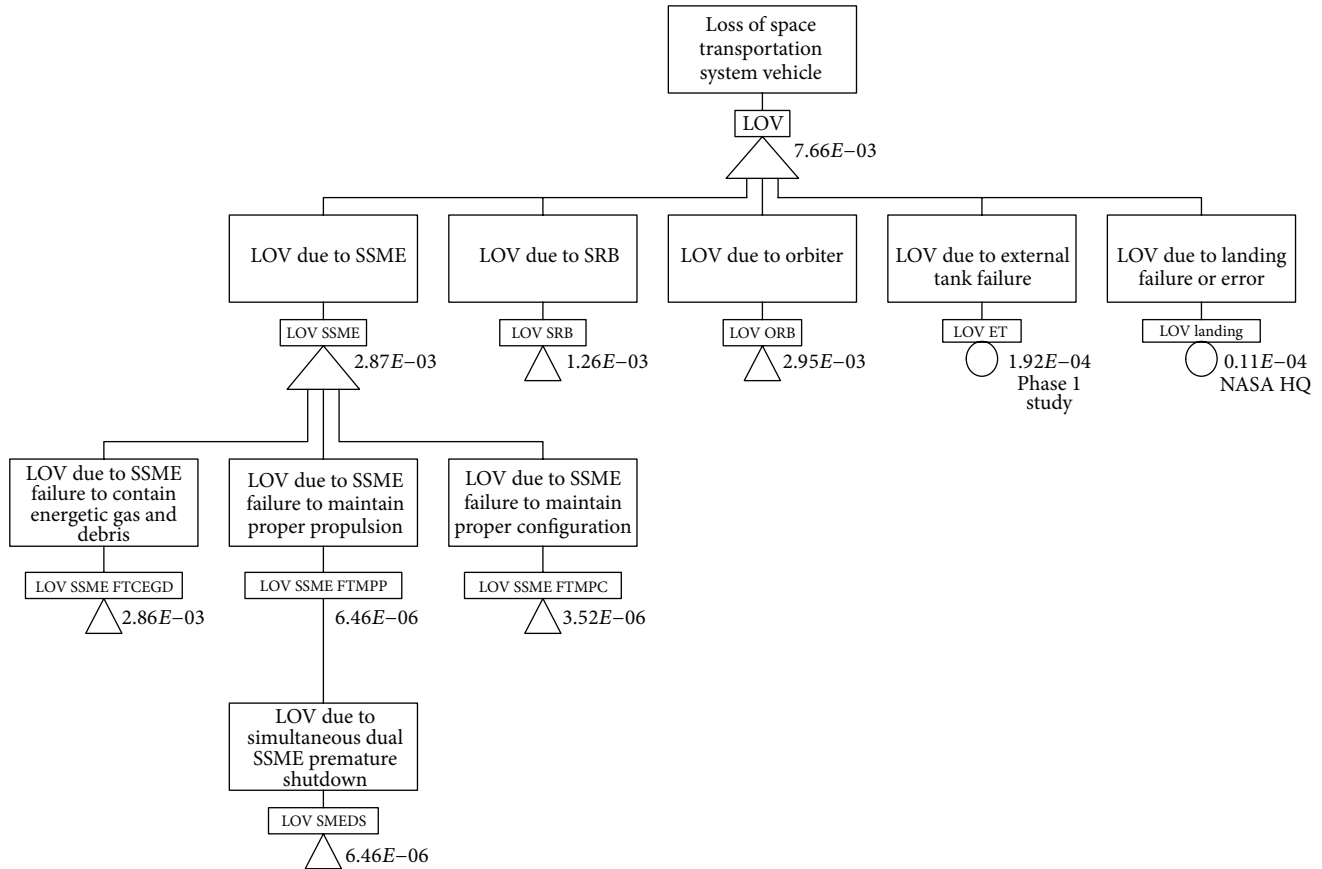
FIGURE 2: Top of the failure tree adapted from the reliability study of Fragola and Maggio [23], where LOV is loss of vehicle, SSME is Space shuttle Main Engines, SRB is Solid Rocket Boosters, ORB is Orbiter, ET is External Tank, FTCEGD is Failure to Contain Energetic Gas and Debris, FTMPP is Failure to Maintain Proper Propulsion, FTMPC is Failure to Maintain Proper Configuration, and SMEDS is Simultaneous Dual SSME Premature Shutdown. Probabilities of failure are also indicated.

determines the location of an accident in the space (using the trajectory simulation model) and the amount of the contaminants released into the atmosphere (using the gas and cloud dispersion model). Similar to all other federates, VRF includes a clock displaying the date and time. Information from the VRF is transmitted through the RTI to SMF federate. The concentration of the contaminant in different locations around the accident site is determined by the VRF one hour after the accident by initiating the gas dispersion model and using the weather conditions for the day of the simulated launch (obtained from the WEMF). This information is provided to the geographical information system (ArcGIS) as an input, and the geographical points where the concentration of the pollutant exceeds the limits determined by the contaminant's Exposure Response Curves are then displayed over a map of Florida. The population exposed to the contaminated area is determined by some of the databases that are layered in ArcGIS. At the end, the number of people exposed to toxic levels released by toxic propellants is shown on the map of Florida by the VRF.

2.1. Data Communications. Since each component in the distributed environment is developed using a dedicated simulation modeling tool (e.g., Arena, AnyLogic), different

schemes are used for data transfer and conversion. It is important to emphasize that all information shared and exchanged by these federates during a federation execution must pass through the RTI. Each federate has a libRTI library, which includes the RTIambassador and the FederateAmbassador class. The libRTI library enables each federate to access RTI services specified in the Interface Specification [10]. Data transfer and exchange processes between federates occur by calling services in the RTIambassador. Transfer and exchange processes from the RTI to the federates are done by asynchronously invoking the FederateAmbassador callback functions that are implemented according to the function of the simulation.

The CRMF is an AnyLogic-based federate. It possesses a code generator which converts the model logic into Java code that supports HLA/RTI interoperability. This model integration is accomplished through the use of the HLA Support Module (HSM) provided by AnyLogic. The HSM enables AnyLogic to support a wide range of RTI services such as Federation Management, Declaration Management, Object Management, and Time Management. The HSM uses a StepHook [27] interface. This StepHook interface places specific methods on the engine that is performing the model's time steps. These methods enable models to exchange messages and

synchronize local simulation times to the global time of the federation.

The WEMF is a Java-based federate which is HLA compliant. Its data publications and subscriptions are queried through the RTI from the CRMF. Messages are sent and received in the GO or NO-GO form.

The SMF, VRF, and MCMF federates are Arena-based models. The integration of these federates is accomplished through the use of the Distributed Manufacturing Simulation (DMS) adapter, which is a component of the HLA Infrastructure for distributed simulation of enterprise facilities. This adapter minimizes the changes needed for simulations to participate in federations by providing time coordination mechanisms, message exchange, and object creation, update, storage, deletion, and transfer. The adapter maintains internal data for each federate: its federate number, federate list, time management data, local/remote object cache, incoming/outgoing message queue, adapter instance properties, and subscription and filtering data. Additionally, the adapter allows the user to set some of the simulation properties using XML, such as: Initialization of SimulationTime, SimulationStepSize, SimulationName, FederationName, and DebugMode. These XML documents are used to specify an "initialization file" and to describe objects and messages. The simulation object, its attributes, and the interactions or parameters of the simulations are stored in XML format which could be accessed through the XML Path Language (XPATH) and the Extensible Stylesheet Language (XSL).

2.2. Lessons Learned: Several RTI Platforms Are Available and You Have to Select an Appropriate One. One of the lessons learned during this initial effort was the selection of the RTI. The performance of the RTI is crucial to the optimization of the federation. For this reason, the evaluation and choice of an RTI were considered during the design phase. The implementation language of the RTI can have an impact on performance. For example, Java implementations may require more system resources, while the cross-platform nature of Java enables it to run without modification on any Java-enabled platform. Other independent variables that affect performance include number of federates, distribution of federates, Data Distribution Management, network transport mode, objects per federate, attributes per object, interactions per federate, parameters per interaction, attribute buffer size, interaction buffer size, and data bundling. The effects of these independent variables on measures of comparison such as latency and throughput should be evaluated before a choice of an open source or commercial RTI is made [11].

There are several commercial and noncommercial implementations of the run-time infrastructure. Some open source RTIs include PoRTIco (http://www.porticoproject. org/index.php?title=Main_Page), CERTI (http://www.nongnu.org/certi/), and EODiSP (http://www.pnp-software.com/eodisp/). PoRTIco is a fully supported open source RTI implementation which is supported by different platforms. It is licensed under the Common Developer and Distribution License (CDDL) and is funded by the Australian Defense Simulation Office (ADSO). PoRTIco is implemented mainly in Java and sometimes runs into compatibility issues with

real-time simulations. Interested developers can have access to the project's source code. CERTI was developed in C++ by the French Aerospace Laboratory (ONERA) to enable it to delve into research in the distributed discrete event simulation domain. The goal of the open source CERTI project is to spread the usage and knowledge of HLA and to foster collaboration with an international open source community. EODiSP was developed by the European Space Agency under the GNU General Public License to support development of end-to-end simulators for Earth observation satellite missions. Development of EODiSP stopped in 2006, and there is presently minimal support provided to developers when they run into difficulties.

Commercial RTIs are more robust in operation than open source RTIs. Commonly used commercial HLA-compliant RTI implementations are the MÄK Real-time RTI, Pitch portable RTI (pRTI), and RTI Next Generation. One advantage of Pitch is its learning curve: Pitch is very visual and can be used to build a fast and complex federation structure. Table 1 gives more information on the aforementioned commercial HLA RTIs.

2.3. Lessons Learned: Advanced Visualization Is Important. Another important lesson learned was related to visualization. Visualization is a very important feature of modern simulation modeling environments. As our research of different visualization paradigms continues, we find that two types of visualizations are required in the context of the VTB distributed simulation [28]. First, a visualization of data and/or the specialized functions is an essential part of Commercial Off the Shelf (COTS) tools. In order to integrate the visualization tool into the VTB, a federate has to be created. This federate will interact with both the RTI and the visualization's external interface. A second type of visualization will have a simulation engine which includes a set of integrated animation facilities to display the state of the system being simulated, which may allow user-model interaction.

Our research has found that there are many visualization tools available. For space operations, among the most sophisticated tools are the Real-Time Advanced Graphics Engine (RAGE) from White Sands Missile Range [29], EDGE (http://active.boeing.com/mission_systems/products/index. cfm?content=products.cfm&pageid=m24121) from Boeing Autometric, and customized environments using JAVA 3D and the Virtual Reality Modeling Language (VRML) as depicted in Figure 3 and other extensions using the Extensible Markup Language (XML), such as X3D, Web3D, and Xj3D.

In addition, another system with distributed capabilities and one of the most popular and complete simulation and visualization COTS available is SIMbox from Simigon (http://www.simigon.com/overview.html), a Modeling, Simulation & Training solutions provider. It is a platform which provides the ability to create, modify, manage, and deploy any simulation-based content.

3. The Enhanced VTB and Demo

We are building an enhanced VTB using a distributed hierarchical simulation platform based on HLA and cloud

TABLE 1: MÄK Real-time RT, Pitch portable RTI (pRTI), and RTI Next Generation RTIs.

|	RTI Next Gen Pro	Pitch	MÄK Real-time RTI
Language implementation	C++	Java/C++	C++
Supported OS	Windows, Linux, Irix, and Solaris	Windows, Linux, Solaris, and so forth	Windows, Linux, Irix, and Solaris
Language bindings	C++, Java, Ada 95, CORBA	C++, JAVA	C++, Java
Developer	Raytheon Virtual Technology Corporation (VTC)	Pitch Corporation (Sweden)	MÄK technologies
Website	http://www.raytheon.com/capabilities/products/rti/	http://www.pitch.se/	http://www.mak.com/products/link-simulation-interoperability/hla-rti-run-time-infrastructure.html
Standard	HLA 1.3, IEEE 1516	HLA 1.3, IEEE 1516-2000, IEEE 1516-2010 (HLA evolved)	HLA 1.3, IEEE 1516-2000, IEEE 1516-2010 (HLA Evolved)
MATLAB toolkit	With adapters	With adapters	HLA/DIS Toolbox
sockets	Basic	Basic	TCP, UDP
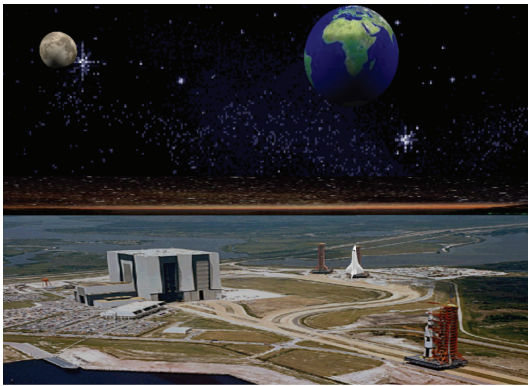
FIGURE 3: JAVA 3D space exploration environment created for the VTB and VRML/X3D object (the space shuttle).

computing with emphasis on the new NASA systems for exploration. These are very unique developments. These demos will be utilized to measure the flexibility of an approach for mission design, validation of strategies, and advancements in tackling complex problems where advanced engineered systems are used. The first demo is of the mission of the SLS and MPCV being launched from the KSC to the ISS.

3.1. Security and Cloud and Tablet Computing. A deficiency of the HLA is that it is not well suited for large-scale distributed simulation systems. Hence, a cloud-based simulation system can enhance the capability of the HLA. Cloud computing provides computing services remotely to users through the internet, thereby minimizing the burden related to managing computing resources and facilities [30]. The benefits that can be realized from cloud computing include but are not limited to on-demand simulation resources, shared and reuse of simulation resources, and load balancing capacity improvement

[30, 31]. Other advantages of cloud computing are cost reduction, resource sharing, and time saved for new service deployment.

HLA provides very few security features when used as a distributed simulation framework. It cannot guarantee integrity and confidentiality of the data exchanged between different federates connected through the web. There are possibilities of intrusion as illegal users can access network through web-enabled HLA/RTI, and any federate may connect and get access to data exchanged between federates [20]. It is also possible for intruders to tamper with the data in transmission networks. To deal with security problems involved in web-enabled HLA/RTI, cloud security features such as Hypertext Transfer Protocol Secure (HTTPS), Identity-based cryptography (IBC), and Public Key Infrastructure (PKI) can be adopted. The communication between federates and RTI needs security checks, and also requests for data require authentication. Users can be authenticated to prevent unauthorized users joining the federation, and sensitive data can be encrypted to maintain the confidentiality.

Tablets provide ease of operation over traditional desktop computers. Tablets can even provide simplicity over laptops to astronauts in order to perform various procedures and scientific experiments. Apple, Samsung, Amazon, Google, and Microsoft are some of the leading companies involved in the production of tablets. At present, the most widely used operating systems on tablets are iOS by Apple and Android by Google. Tablets are light in weight which makes them more portable. However, they provide less storage space as compared to desktops or laptops. To overcome local storage space and processing power drawbacks, tablets can work in conjunction with the cloud.

The application of tablet computing in the cloud can provide flexibility of operation in spacecraft systems. Tablets can be used by astronauts as mobile devices for monitoring and visualization of space. The tablet can work as a display interface, while all computing and processing is done via the cloud.
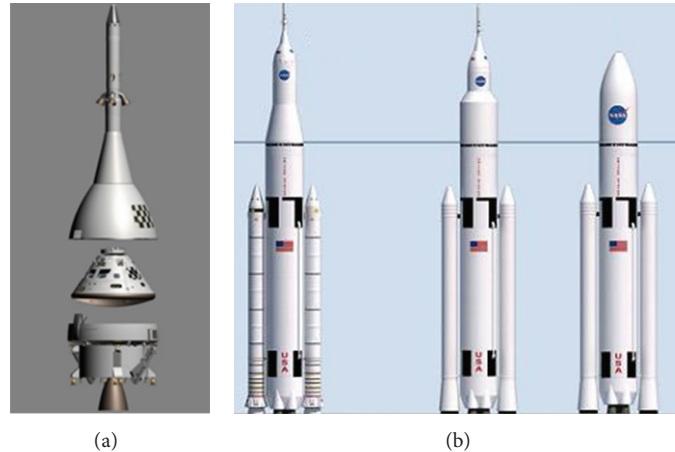
FIGURE 4: The MPCV and SLS [32, 33]. (a) Multipurpose Crew Vehicle (MPCV) [21]. (b) Evolution of the Space Launch System (SLS) [15].

Data processed on the tablet can also be saved into cloud. Astronauts can query the system, input their observations, and perform online data mining to spot trends through the use of tablets. With voice and gesture recognition, astronauts can connect with components to form "network ontology." Using the computing hierarchical/distributed infrastructure, astronauts can also study correlations and run simple simulation models of the current observed situations.

3.2. Demo: Mission to the International Space Station (ISS). NASA has announced that the next manned spacecraft will be the MPCV, which is based on the Orion, the Apollo era crew capsule design (Figure 4(a)). The MPCV and SLS (Figure 4) are central to NASA's plan for the future of space exploration beyond Low Earth Orbit (LEO). The NASA Authorization Act of 2010 gives NASA until 2016 to field a heavy-lift rocket (now called the Space Launch System) and crew vehicle. This act authorizes approximately $10 billion in spending on the two projects over the next three years [34]. To meet the above goal, NASA plans to implement the MPCV and the Space Launch System (SLS) programs, including transition of relevant design and developmental activities of the previous programs. A major element of the transition involves shifting design and developmental efforts away from a closely coupled system to a more general launch vehicle (i.e., SLS based on the Heavy Lift Vehicle, Figure 4(b)) and crew vehicle (i.e., MPCV, Figure 4(a)).

Therefore, our first demo is the implementation of a mission of the SLS and MPCV being launched from KSC to the ISS. The mission is modeled at a very high level (in the hierarchy) using agent-based modeling. Several discrete models representing different parts of the mission such as the ground operations (e.g., transportation, assembly/stacking), the launching process, and reentry are being developed. Several of these models are built by consulting NASA experts and using as a baseline the processing times/features of the NASA shuttle and the current infrastructure such as the Vehicle Assembly Building (VAB; see Figure 5) that is going to be used in the future processes. Another model is a sophisticated decision level fusion approach based on Distribution

FIGURE 5: The Vehicle Assembly Building (VAB) was built for the Apollo program. The VAB was designed to house the assembly and final checkout of the 110-meter-tall Saturn V launch vehicle. One of the VAB's most notable features is its 139-meter-high doors. It was also used for the assembly/stacking of the NASA Space shuttle. The VAB will be used for the assembly/stacking of the SLS and the MPCV. Photo courtesy of NASA.

Envelope Determination. Several models are connected that implement fragmentation of debris, release of toxic gases, and propagation of blast waves, which are the three majors hazards to be produced by the SLS. Examples of some of the developed simulation federates are explained below.

3.2.1. Mission Process Agent Federate. The Mission Process Agent is the heart of the hierarchy. It describes the life cycle of a mission and owns different environments where the different decision-maker agents, resource agents, and other process agents can work together and collaborate [12]. However, the advantage of using the agent framework is the assignation of environments and features which allow other agents to use the environment and participate and collaborate with other subprocesses in the process. The following processes are required (see Figure 6).

(1) Supply Chains, Rollover and Vehicle Assembly Building (VAB; See Figure 5). This step in the life cycle of the mission
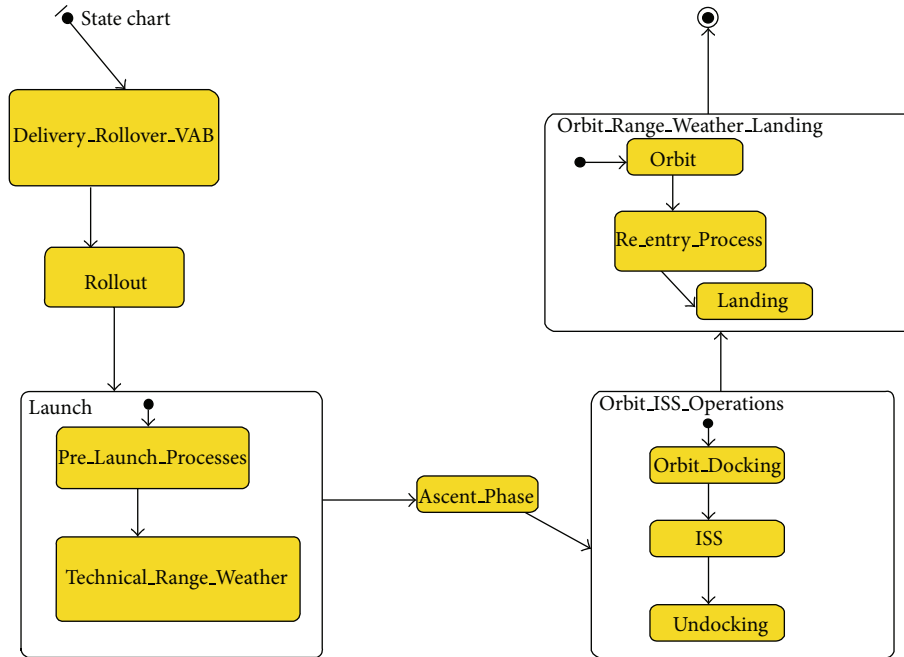
FIGURE 6: State chart of a mission to the ISS using AnyLogic (http://www.anylogic.com/) for the MPCV and SLS.

details the different resources and systems between NASA Centers and NASA Headquarters (HQ) for the mission and the external supply chain (i.e., the interactions between NASA and Major Contractors). In addition, the rollover of the major systems and the different processes occur inside the VAB.

(2) Rollout. This step is very short in time. The vehicle is transferred from the VAB to the launch pad.

(3) Launch Operations. This step includes prelaunch operations to be performed on the vehicle on the launch pad. There are many interactions among different agents. The decision-maker agents such as the launch director, range safety, weather officers, and the crew technician agents are heavily involved during this step. Scrubs are simulated, and the assignment of potential launch dates is also modeled. The weather and the range systems are executed accordingly. The final launch is modeled.

(4) Ascent Phase. This is a step with a short period of time. It simulates the Solid Rocket Boosters (SRBs) and the phases being released.

(5) Orbiting, Rendezvous, Docking, Orbit Operations, and Undocking. This step simulates the orbit, rendezvous, and docking of the vehicle (MPCV and the service module) with the ISS. Undocking and the planning of the reentry and landing (interactions of the different agent decision-makers such as the entry flight director, weather, and range safety officers, etc.) are simulated.

(6) Orbiting, Entry, and Landing/Recovery. This is the final step of the mission with the final orbiting, the release of

the service module, the entry and landing at a particular location (e.g., California Coast), and the logistics of the recovery.

The simulation platform selected is AnyLogic (http://www.anylogic.com/). An "Agent" in AnyLogic is a unit of model design that can have behavior, memory (history), timing, and contacts. Agents can represent people, companies, projects, assets, vehicles, cities, animals, ships, products, and so forth. AnyLogic has classes for developing agents as it has all necessary properties to define variables, events, statecharts, system dynamics stock, and flow diagrams.

3.2.2. Simulation Model of the Stacking in the VAB of the SLS Federate. This is a discrete-event simulation model. It was built by consulting NASA experts and uses the processing times/features of the NASA shuttle as a baseline. The SLS being developed consists of different modules as shown in Figure 7. These modules must be assembled in the VAB. The following sequences are required (see Figure 8) for an implementation using AnyLogic (http://www.anylogic.com/).

(1) Phases 1 and 2 Transfer to VAB. The first phase and second phases arrive at KSC. They are inspected and then off loaded and towed to the VAB transfer isle where they are stored until integrated with the SRB stack.

(2) Solid Rocket Boosters (SRBs) Stacking in the VAB High Bay. The Solid Rocket Booster (SRB) stacking consists of placing an SRB's aft skirt onto hold-down posts on the Mobile Launch (ML in one of the VAB High Bays (HB)). The SRBs are then stacked one segment at a time until all five segments are stacked. At this time the forward extension that houses the avionics and parachutes is added and the SRB stacking is complete. As explained by [25] "These boosters are derived
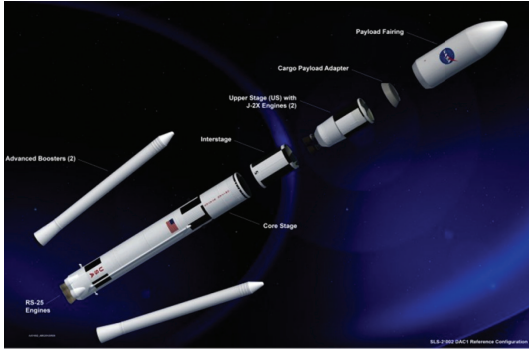
FIGURE 7: This picture (adapted and modified from [35]) describes one of the models of the SLS. The SLS will be the most powerful space launch vehicle ever flown. Additional boost from a pair of the largest solid rocket motors (SRBs) ever to be built will be required to lift the payload into orbit. The assembly/stacking of the different boosters, stages/phases, and the MPCV will occur at the VAB.

from the Space shuttle boosters, though they are larger and of an improved design. Whereas the shuttle boosters were made in four segments, the SLS boosters are made in five. These segments contain the fuel, which is composed of ammonium perchlorate, powdered aluminum, iron oxide, a polymer (such as Polybutadiene acrylonitrile (PBAN) or Hydroxyl-terminated polybutadiene (HTPB)) and an epoxy curing agent."

(3) Phases (i.e., Stages) 1 and 2 Are Assembled and Mated to the SRB Stack in the VAB. This is accomplished by raising the phases to a vertical position in the transfer isle, lifting it up and over into the HB and mating it to the stacked SRBs.

(4) MPCV with Service Module to VAB. The MPCV is towed to the VAB and placed in the VAB transfer isle. A strong back is attached to the MPCV and service module, and the vehicle is lifted up and moved, lowered and attached to the Phase 1/Phase 2/SRB stack.

3.2.3. Simulation Models of Range Safety Federate. This federate includes several models (mainly continuous) that abstract the potential destruction of the vehicle and its consequences such as gas dispersion, debris, and blasts from sound waves. The loss of two of the five Space shuttles during both the launch and the return phases of flights has raised public awareness on the safety issues related to space launches. Therefore, simulating mission failures which may result in the loss of life or property is a capability which was deemed important to integrate in the VTB. This federate considers the three main hazards, that is, debris dispersion, gas dispersion, and blast propagation. This subsection introduces briefly each of the models and discusses the information-fusion-based metric which was developed to estimate more appropriately the risk of operating a vehicle of a particular type, on a particular day, from a particular spaceport. A full discussion of this fusion-based methodology can be found in [24, 36].

(1) Debris Modeling. As its name indicates, the purpose of a debris model is to model the fragmentation and debris impact dispersion resulting from the breakup of a space vehicle in flight. For example, NASA uses the Common Real-Time Footprint (CRTF) in its decision to abort a launch. A debris dispersion simulation model was developed and validated with actual debris locations recovered from Space shuttle Columbia, the details of which can be found in [37]. Uncertainties accounted for when calculating the trajectories of debris include real-time state vector, fragment initial velocity, drag, lift, and wind. Figure 9 shows debris areas of three simulated break-up times of a vehicle launched from Kennedy Space Center (KSC). The outputs are overlaid on an ArcGIS map (http://www.esri.com/software/arcgis). The areas increase exponentially as the breakup occurs later in flight.

(2) Blast Modeling. An explosion is generally defined as a rapid release of energy into the atmosphere. This energy generates blast waves that can significantly damage the area surrounding the source of the explosion. In conventional launcher designs, the weight of the propellant carried by the vehicle can represent up to 90% of its total gross weight at launch. Therefore, it is important to understand the explosion potential of this propellant to reliably assess the level of risk to the public and the surrounding infrastructure (which may extend beyond the spaceport) associated with the use of a launch vehicle. A well-known software for blast modeling is the Distant Focusing Overpressure software (BlastDFO) developed by Acta Inc. This software incorporates real-time weather data in order to predict the potential for window breakage and casualties if an on-pad or early flight explosion occurs [24].

(3) Gas Dispersion and Toxicity Modeling. Given the amount and toxicity of fuels carried by launch vehicles, modeling the dispersion of gas released during an explosion is critical. A prominent example of systems developed to model such phenomenon is CALPUFF, an advanced nonsteady-state meteorological and air quality modeling system [8]. For the present effort, AERMOD, another model recommended by the Environmental Protection Agency (EPA), was used. AERMOD is a modeling system designed to calculate air pollutant concentration in all types of terrain, from flat surfaces to complex, mountainous terrains [24]. These capabilities are useful for modeling operations in different types of terrain, which could include both spaceport located on each coast (such as KSC in Florida and Vandenberg Air Force Base in California) and those that could be envisioned inland (such as the Oklahoma Spaceport).

(4) Estimating Launch Risk through Information Fusion. Estimating the risk incurred by the public as the result of operating a launch system is a complex task, and ensuring that the safety of the public is a significant cost driver of space launchesOne cannot be too cost conscious as this may result in operation that is unsafe for the public. On the other hand, being too conservative leads operations to be cost-prohibitive in many instances. As advocated by Sala-Diakanda [24–26, 36],
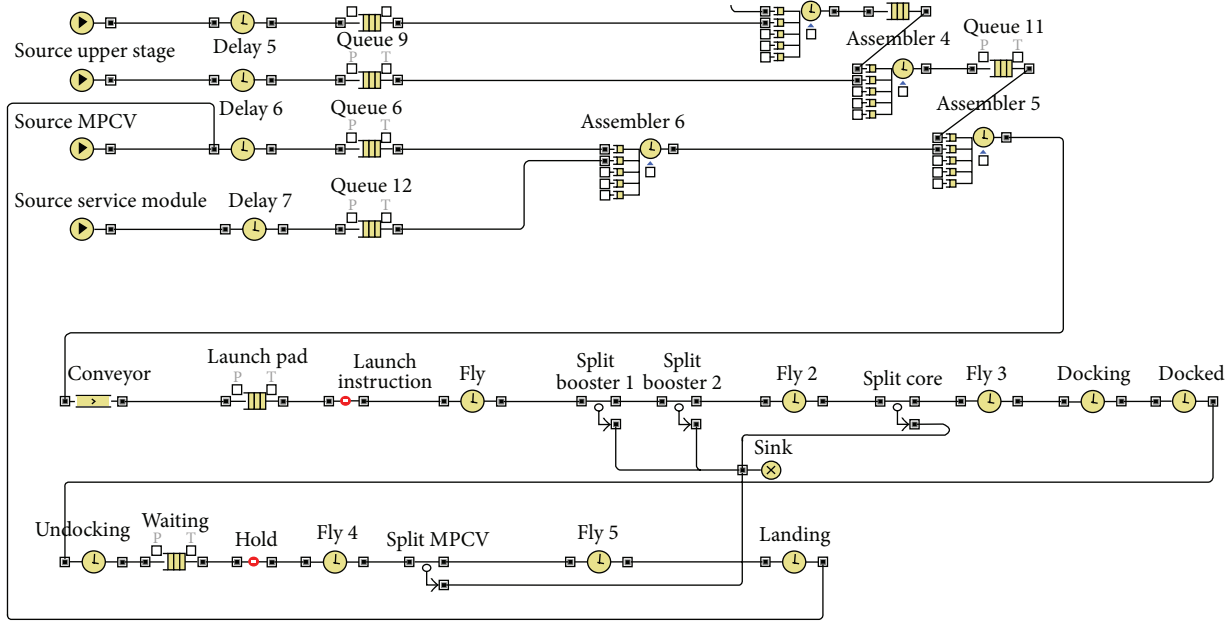
FIGURE 8: Partial depiction of the discrete-event model for the SLS and MPCV stacking operations inside the VAB using AnyLogic (http://www.anylogic.com/).
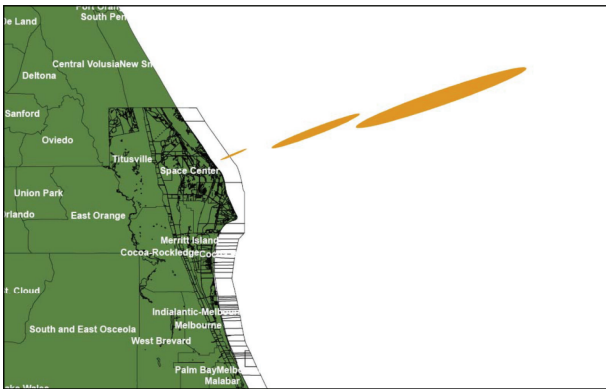

FIGURE 9: Partial details of a debris model for a spacecraft [24].

the right course to adopt is to shift the practice from a risk avoidance philosophy to a risk management philosophy. Understandably, current approaches are too conservative because there are simply too many uncertainties associated with such operating launchers. These uncertainties are introduced by such factors as (1) the difficulty in capturing all the failure modes of a system and their probability of occurrence due the lack of historical data and the sheer complexity of those systems, (2) the difficulty in modeling population distribution and hazard-specific sheltering scheme, or (3) the complex interdependencies between the different hazards when it comes to estimating the potential number of casualties. Indeed, if one is considered a casualty from a debris dispersion perspective, it is perhaps too conservative to count such a person from gas dispersion perspective as well.

An information-fusion-based metric, based on Distribution Envelope Determination (DEnv), also known as

Interval-Based Dependency Bounds Analysis, was proposed by Sala-Diakanda [24, 36]. DEnv is a convolution-based method for determining dependency bounds of binary arithmetic operations on random variables. This metric addresses precisely the problems of the uncertainty surrounding the mean number of casualties (the current metric) and the prevailing assumption of independence between the effects of all hazards by generating minimum and maximum joint cumulative distribution functions of variables that are dependent, but whose dependencies are unknown or only partially known.

The proposed metric shifts the focus from a mean value whose uncertainty is too large to a confidence around the "probability of exceeding a predetermined safety threshold." Therefore, from a decision maker perspective, with this metric, the decision to be taken is shifted from being based upon a subjective assessment of the size of the uncertainty around the mean to being based upon a range of probabilities of exceeding a prespecified safety threshold. And with respect to the assumption of independence, Sala-Diakanda [24] suggested that a better assumption than independence is "no assumption at all." To illustrate, suppose that the threshold value for the expected number of casualties is 3 (i.e. $E_{\text{CTRESH}} = 3$), then the proposed metric may generate an estimate of the form

$$\text{Min}_{\text{RISK}} = \text{minprob}\left(E_{\text{CFUS}} \geq 3\right) = 25\%,$$
$$\text{Max}_{\text{RISK}} = \text{maxprob}\left(E_{\text{CFUS}} \geq 3\right) = 67\%. \tag{1}$$

Here, Min_{RISK} and Max_{RISK} are, respectively, the minimum and maximum probabilities of a fused expectation of casualties of exceeding the safety threshold. A detailed case-study illustrating the benefits of such information-based
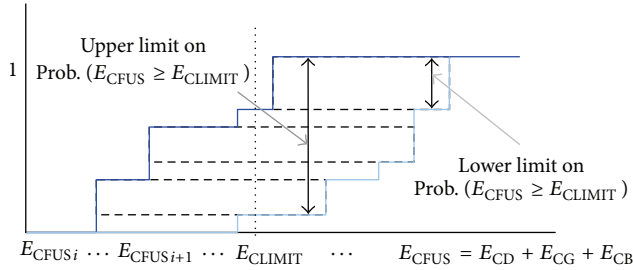
FIGURE 10: Curves bounding the true cumulative distribution function (cdf) of the fused output E_{CFUS}. E_{CB}, E_{CD}, and E_{CG} are expectation of casualties' random variables generated, respectively, by blast propagation, debris fragmentation, and gas dispersion. E_{CFUS} is defined as the fused expectation of casualties random variables generated by DEnv.

metric, how it can be used and interpreted, was discussed in [24, 36]. The concept is illustrated graphically in Figure 10.

4. Conclusions

Distributed simulation is very important to tame complexity. It is essential to emphasize the hybrid nature of distributed simulation models where discrete-event and continuous models are required due to the nature of the engineered systems [37, 38]. There are many sources of expertise required to build and model these engineered systems. Then, there is a need for different type of models to have the analysis capability to encompass their subsystems, processes, and life cycles.

The importance of simulation has been highlighted by the NASA Office of the Chief Technologist (OCT). NASA OCT explains that [39] "a digital twin is an integrated multi-physics, multi-scale, probabilistic simulation of a vehicle or system that uses the best available physical models, sensor updates, fleet history, and so forth, to mirror the life of its flying twin." Our approach can support the development of digital twins. In addition, components of one mission can be used in the planning for other types of missions.

This approach of Hierarchical/Distribution simulation modeling can be used for planning at different levels (i.e., strategic, operational, and tactical). It is very important to appreciate the level of integration to be achieved with other information systems and the real-time issues involved in particular for advanced digital twin concepts. Scripted visualization and simulation visualization are very different concepts. Simulation visualization is the one requested by the analysts.

A very important component of our current research focuses on the uncertainty aspects at the levels of data and operations. We are studying fuzzy logic and deep learning neural network approaches to model imprecision and ambiguity [40]. In addition, we are studying behavioral simulation in order to model team productivity effects of the human part of the systems [41].

This paper outlined some of our preliminary work that will evolve toward a more sophisticated and responsive simulation environment. We will report our progress in future papers.

Acknowledgments

References

[1] L. Rabelo, M. Marin, and L. Huddleston, "Data mining and complex problems: case study in composite materials," *SAE International Journal of Aerospace*, vol. 2, no. 1, pp. 165–170, 2010.

[2] L. Rabelo, P. Fishwick, Z. Ezzell, L. Lacy, and N. Yousef, "Ontology-centred integration for space operations," *Journal of Simulation*, vol. 6, pp. 112–124, 2012.

[3] L. Rabelo, M. Marin, P. Fishwick, and Z. Ezzell, "The semantic web and space operations," *SAE International Journal of Aerospace*, vol. 4, no. 2, pp. 652–660, 2011.

[4] M. Marin, L. Rabelo, and J. Sepulveda, "Spaceport simulation models integration," *Journal of Aerospace*, vol. 114, pp. 1264–1271, 2006.

[5] L. C. Rabelo, J. Sepulveda, J. Compton, and R. Turner, "Simulation of range safety for the NASA space shuttle," *Aircraft Engineering and Aerospace Technology*, vol. 78, no. 2, pp. 98–106, 2006.

[6] L. Rabelo, J. Sepulveda, M. Marin, and J. Park, "A distributed environment for spaceports," *Journal of Aerospace*, vol. 113, pp. 1507–1516, 2005.

[7] L. Rabelo, J. Sepulveda, J. Compton, R. Moraga, and R. Turner, "Disaster and prevention management for the NASA shuttle during lift-off," *Disaster Prevention and Management*, vol. 15, no. 2, pp. 262–274, 2006.

[8] L. Rabelo, J. Bardina, Y. Zhu, and J. Compton, "Ground and range operations for a heavy-lift vehicle: preliminary thoughts," *SAE International Journal of Aerospace*, vol. 4, no. 2, pp. 1064–1073, 2011.

[9] J. Sepulveda, L. Rabelo, and J. Compton, "A distributed environment for analysis of events related to range safety," *SAE Transactions, Journal of Aerospace*, vol. 113, pp. 1517–1521, 2005.

[10] J. S. Dahmann, R. M. Fujimoto, and R. M. Weatherly, "DoD high level architecture: an update," in *Proceedings of the Winter Simulation Conference (WSC '98)*, pp. 797–804, December 1998.

[11] L. Malinga and W. Le Roux, "HLA RTI performance evaluation," in *Proceedings of the SISO European Simulation Interoperability Workshop*, pp. 1–6, Simulation Interoperability Standards Organization, Istanbul, Turkey, 2009.

[12] G. Pedrielli, M. Sacco, W. Terkaj, and W. Toli, "An HLA-based distributed simulation for networked manufacturing systems analysis," *Journal of Simulation*, vol. 6, pp. 237–252, 2012.

[13] S. Tang, T. Xiao, and W. Fan, "A collaborative platform for complex product design with an extended HLA integration architecture," *Simulation Modelling Practice and Theory*, vol. 18, no. 8, pp. 1048–1068, 2010.

[14] C. McLean and F. Riddick, "The IMS mission architecture for distributed manufacturing simulation," NISTIR 6559, National Institute of Standards and Technology, Gaithersburg, Md, USA, 2000.

[15] K. Al-Zoubi and G. Wainer, "RISE: a general simulation interoperability middleware container," *Journal of Parallel and Distributed Computing*, vol. 73, pp. 580–594, 2013.

[16] S. Tang, T. Xiao, and W. Fan, "A collaborative platform for complex product design with an extended HLA integration architecture," *Simulation Modelling Practice and Theory*, vol. 18, no. 8, pp. 1048–1068, 2010.

[17] O. Topcu and H. Oguztuzun, "Layered simulation architecture: a practical approach," *Simulation Modeling Practice and Theory*, vol. 32, pp. 1–14, 2013.

[18] Z. Tu, G. Zacharewicz, and D. Chen, "Developing a web-enabled HLA federate based on poRTIco RTI," in *Proceedings of the Winter Simulation Conference (WSC '11)*, pp. 2289–2301, December 2011.

[19] G. A. Wainer, R. Madhoun, and K. Al-Zoubi, "Distributed simulation of DEVS and Cell-DEVS models in CD++ using webservices," *Simulation Modelling Practice and Theory*, vol. 16, no. 9, pp. 1266–1292, 2008.

[20] Z. H. Zhang, X. D. Chai, and B. C. Hou, "System security approach for web-enabled HLA/RTI in the cloud simulation environment," in *Proceedings of the 6th IEEE Conference on Industrial Electronics and Applications (ICIEA '11)*, pp. 245–248, June 2011.

[21] G. R. Cates, M. J. Steele, M. Mollaghasemi, and G. Rabadi, "Modeling the space shuttle," in *Proceedings of the Winter Simulation Conference*, pp. 754–762, December 2002.

[22] T. Rajkumar and J. E. Bardina, "Web-based weather expert system (WES) for space shuttle launch," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '03)*, System Security and Assurance, pp. 5040–5045, October 2003.

[23] J. Fragola and G. Maggio, "Probabilistic risk assessment of the space shuttle. Phase 3: a study of potential of losing the vehicle during nominal operation, Vol. 2: integrated loss of vehicle model," Report, report submitted by Science Applications International Corporation (SAIC) to NASA, 1995.

[24] S. Sala-Diakanda, *A framework for the assessment and analysis of multi-hazards induced risk resulting from space vehicles operations [Ph.D. dissertation]*, University of Central Florida, 2007.

[25] S. Sala-Diakanda, L. Rabelo, and J. Sepulveda, "On the development of a comprehensive hazard modeling tool through distributed simulation: learning from the Columbia space shuttle accident," *Journal of Aerospace*, vol. 114, pp. 1186–1192, 2006.

[26] S. Sala-Diakanda, L. Rabelo, and J. Sepulveda, "Case for a multidisciplinary modeling platform for space launch risk analysis," *Journal of Aerospace*, vol. 116, pp. 946–950, 2008.

[27] A. Borshchev, Y. Karpov, and V. Kharitonov, "Distributed simulation of hybrid systems with AnyLogic and HLA," *Future Generation Computer Systems*, vol. 18, no. 6, pp. 829–839, 2002.

[28] J. Dawson, *A holistic usability framework for distributed simulation systems [Ph.D. dissertation]*, A Holistic Usability Framework for Distributed Simulation Systems, University of Central Florida, 2006.

[29] J. Northrup, "RAGE: Real-Time Advanced Graphics Engine," 2010, https://bop.peostri.army.mil/sites/bop/Lists/Sources%20 Sought/Attachments/251/RAGEOperationsManual.pdf.

[30] R. Fujimoto, A. Malik, and A. Park, "Parallel and distributed simulation in the cloud," *SCS M&S Magazine*, vol. 3, pp. 1–10, 2010.

[31] H. He, R. Li, X. Dong, X. Zhang, and H. Han, "An efficient and secure cloud-based distributed simulation system," *Applied Mathematics*, vol. 6, no. 3, pp. 729–736, 2012.

[32] C. Bergin, "Dynetics and PWR aiming to liquidize SLS booster competition with F-1 power," 2012, http://www.nasaspaceflight.com/2012/11/dynetics-pwr-liquidize-sls-booster-competition-f-1-power.

[33] M. Eddy, "A Closer Look at NASA's Multi-Purpose Crew Vehicle," 2011, http://www.geekosystem.com/nasa-mpcv-info-graphic/.

[34] Review of U.S. Human Spaceflight Plans Committee, "Seeking a Human Spaceflight Program Worthy of a Great Nation," NASA Publications, 2009, http://spacegrant.nmsu.edu/ispcs/2009/news/augustin_report.pdf.

[35] D. Szondy, "NASA building world's largest solid-fuel rocket," 2012, http://www.gizmag.com/sls-largest-solid-rocket/24408/.

[36] S. N. Sala-Diakanda, J. A. Sepulveda, and L. C. Rabelo, "A methodology for realistic space launch risk estimation using information-fusion-based metric," *Information Fusion*, vol. 11, no. 4, pp. 365–373, 2010.

[37] L. Rabelo, H. Eskandari, T. Shaalan, and M. Helal, "Value chain analysis using hybrid simulation and AHP," *International Journal of Production Economics*, vol. 105, no. 2, pp. 536–547, 2007.

[38] L. Rabelo, M. Helal, A. Jones, and H.-S. Min, "Enterprise simulation: a hybrid system approach," *International Journal of Computer Integrated Manufacturing*, vol. 18, no. 6, pp. 498–508, 2005.

[39] NASA Office of the Chief Technologist (OCT), "DRAFT Modeling, Simulation, Information Technology & Processing Roadmap Technology Area 11," 2010, http://sites.nationalacademies.org/DEPS/ASEB/DEPS_059552.

[40] E. Cortes and L. Rabelo, "An architecture for monitoring and anomaly detection for space systems," in *Proceedings of the SAE AeroTech Conference*, Montreal, Canada, 2013.

[41] A. Conner and L. Rabelo, "Utilizing team productivity models in the selection of space exploration teams," in *Proceedings of the SAE AeroTech Conference*, Montreal, Canada, 2013.

The Scientific World Journal

▸ Impact Factor **1.730**

▸ **28 Days** Fast Track Peer Review

▸ All Subject Areas of Science

▸ Submit at http://www.tswj.com

Hindawi