A University Perspective on the 2011 NASA/SISO Smackdown Modeling and Simulation Outreach Event

Nana Essilfie-Conduah
Paul Grogan
Phillip M. Cunio
Ryan McLinko
Olivier de Weck
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139
617-335-3840
nconduah@mit.edu, ptgrogan@mit.edu, pmcunio@mit.edu, mclinkor@mit.edu, deweck@mit.edu

Keywords:
HLA-Evolved, Smackdown, Space Logistics, Hopper, In-situ Resource Utilization, University Outreach

ABSTRACT: *Modeling and simulation (M&S) functions as a method and set of tools to better realize and enhance the functionality of engineering systems. While M&S is extensively used as a required skill in many engineering professions, M&S itself lacks extensive academic programs at the university level. With the intent of alleviating this issue the NASA/SISO Smackdown competition aimed to introduce university students to the HLA-Evolved (IEEE 1516-2010) standard for federated simulation. Having participated in the competition with a focus on the logistics of a lunar fuel economy, the MIT team of graduates and undergraduates reflect in this paper on the standard itself and the competition as a whole. We make suggestions on what could be improved. The paper highlights the thought process with reference to the modeling approach used, how it shaped the final simulation, commentary on how future Smackdown events could be improved, and how teams could be rated. The paper also serves as a reference for future teams in terms of a suggested approach to tackling the HLA-Evolved standard and participating in the Smackdown competition events.*

1. Introduction

Modeling and simulation (M&S) plays a great role in today's science and engineering endeavors. Many systems and problems, especially within the space industry, are immensely complex and weighty in their implications. It is thus highly advisable to perform some form of calculated foresight, in order to avoid unforeseen problems that could result in a detrimental flaw in the project as a whole. M&S has the potential to provide this foresight and the prospect of virtual testing where complete and actual tests are too expensive to undergo, in terms of cost, time and other resources. With the increasing needs and thus complexity of engineering problems, it is likely that modeling and simulation knowledge and practices have and will continue to play an important role in the success of many companies, research projects, and underlying technologies.

It is thus advantageous that students be exposed to basic modeling and simulation practices for future applicability. By increasing the number of motivated students with basic modeling and simulation skills, the workforce of the future will be better prepared to tackle the demanding and challenging problems that await. The issue, however, is that it is currently difficult to find educational programs that offer a strong M&S background at the university level. Many courses do have certain components that encompass M&S; however the focus is usually not placed on structure or on highlighting its broader usefulness. By realizing the potential of the HLA Evolved (IEEE 1516-2010) standard [1] to unify simulation practices and bringing the challenge of exploring the breadth of its advantages in the Smackdown completion, SISO, NASA, and the other sponsors are playing their part to increase the number of students who are exposed to the complexity of modeling and simulation.

As aerospace engineers interested in the logistics of a space resource economy and hopper vehicles, the use of M&S techniques presents an interesting avenue to provide the needed sophisticated foresight. One of the largest detriments to many engineering program conceptions is the inability to properly gauge and estimate the need for varied resources and the extent to which a technology can be expected to work and interface with others. M&S is an approach to avoid this within science and engineering. The HLA-Evolved (IEEE 1516-2010) standard provides a

general framework for building interoperable and reusable simulations and further alleviating the process of developing independent models that are difficult to expand or combine in future work.

By using this platform in the context of the 2011 SISO Smackdown competition the MIT team aimed to explore the initial possibilities of a lunar mining economy that made use of advanced concept vehicles such as the MIT and Draper Lab's TALARIS (Terrestrial Artificial Lunar And Reduced GravIty Simulator [2]) hopper to search for resource deposits and an in-situ resource utilization (ISRU) plant to collect and process usable resources such as oxygen. Furthermore, this concept enables the idea of creating and sustaining a lunar depot that would receive mined resources and serve as a logistics hub on the moon. This has many implications in the space industry – if feasible, it could drastically reduce the need for heavy fuel expenditure needed by shuttles traveling from the Earth.

Thus, by taking part in the competition the MIT graduate and undergraduate student team gained experience and insight in the realm of M&S, specifically the HLA-Evolved standard. Moreover, the model allowed for many different mission-planning phases to be explored in reference to the use of the hopper. Given that the TALARIS project is associated with a team in the Next Giant Leaps team within the Google Lunar X-Prize competition, the interoperability of the standard would also illuminate the interaction between the hopper and other vehicles. Finally, the sponsor participation in the Smackdown event was a huge incentive for networking and opportunities to see the benefits and challenges of applied M&S technologies.

In the following sections the initial objectives set out by the MIT team in the competition will be discussed. The project management tasks performed by the team will also be covered, highlighting how they played a key role in the simplification of the task. Attention will also be given to the actual implementation of the standard with special emphasis on the MATLAB platform employed. A final conclusion will be made based on the findings in regards to the success of the effort and suggestions made as to what improvements could be made in the future by the competition organizers.

2. MIT Smackdown Objectives

The MIT team's initial assessment of the Smackdown objectives indicated that the levels of participation (see Section 3.4) mapped onto levels of interaction with the federate and with other teams. At the highest level of participation, a federate created by the MIT team should be able to interact with the Run Time Infrastructure (RTI) environment; other teams' federates, and with other MIT federates. Lower levels of participation would entail successively lower levels of interaction, moving from interaction with only other MIT federates to only interaction with the RTI environment to only observing the interactions of other federates. The MIT team elected to participate at the highest level, preserving the option to scale down participation as the availability of team members dictated. This was possible due to the way in which each higher level of participation built upon each preceding level.

The character of the participation was influenced by the character of the event, the character of the simulated mission, and the character of the uses to which the software had been put in the past. Because HLA was originally developed for use in military simulations, and because the competition was called a "Smackdown," there was initially a perception that the interactions that could occur between a federate and the RTI or between federates would have to be competitive. However, given that the simulation was to be centered around space exploration of the Moon, and given the inherently cooperative nature of joint space exploration to date, as well as the fact that exploration of harsh environments like deep space lends itself better to cooperation than competition, there emerged primarily an element of collaboration between teams and federates as our discussions progressed.

In the end, the MIT team's contribution was based on a logical evolution of the likely state of the Smackdown competition. It was anticipated that succeeding years would include more advanced modeling efforts and complex interactions, although it was also expected that the potentially dual competitive and cooperative character of interactions could be retained. The obvious end state of this situation is a complete economy, wherein agents compete for resources but cooperate in trading them. Expecting this lunar economy to come into being in future competitions (and further expecting that it may eventually come into being in the real world), the MIT team tailored its participation in the competition to reflect this sort of expected future. The logical basis of such an economy would be fuel derived from natural lunar resources, and accordingly the MIT federates were tailored to this expectation.

The MIT contribution was split into two active federates plus an enhanced environment:

The first active federate was the mobile ISRU production plant, which generated usable fuel after gathering and processing lunar resources. Although there are multiple types of resources on the Moon, including but not limited to water ice, ilmenite, and other minerals, the MIT team elected to simplify the resource model to two in-situ resources and one propellant-type resource which could be produced via processing. Due to the timescale of the Smackdown event (minutes of simulation), placeholder processing models were used in lieu of realistic models. It is expected that future Smackdown executions will require more detailed models in terms of resources and fuels.

The second active MIT federate was an exploratory hopper, intended to scout the terrain of the Moon for resources to be used by the ISRU plant to generate fuel. The hopping technology and assumed performance characteristics were based on work currently being done by the TALARIS team, which proposes to use hoppers as high-mobility vehicles to explore planetary bodies with low gravity and little or no atmosphere [3].

The third federate, which was not completely implemented but rather integrated with the hopper federate, was an enhanced environment model of a resource distribution to overlay the Moon's terrain. The concept is that the resource distribution would be partially observable to other federates through an analysis interaction, with some level of error or uncertainty. For example, the exploratory hopper may land at a location and perform an in-situ analysis of the resource concentrations by sending an interaction to the enhanced environment, which would then return an interaction providing a "sensed" resource concentration that would be sampled from its "known" distribution with some component of random noise added.

The three federates represented various levels of interactivity, as indicated in Table 2.1. Note, as implemented, the resource analysis interactions (italicized) were modeled within the hopper federate rather than in a separate environment federate.

The main interactions involving the MIT Smackdown federates included resource transfer between the ISRU Plant and the Lunar Rover, resource analysis of the environment performed by the hopper, and scouting reports sent by the hopper to the ISRU plant. This set of intra-team and inter-team interactions was deemed sufficient to explore the breadth of possible interactions, and formed the basis for the MIT team's contribution to the Smackdown competition.

3. Project Management

Several aspects under the categorization of project management contributed to achieving the MIT team Smackdown objectives. First, it was apparent that it would be difficult for each team member to have mastery of both the HLA-Evolved standard as well as the individual federates under development; thus the team was organized according to specialty. Second, in order to mitigate the risk of integrating successive levels of modeling challenges, incremental levels of participation were identified to serve as "fallback" positions. Finally, several key project milestones were identified to carefully track progress en-route to the Smackdown event execution.

3.1. Team Organization

The MIT team divided responsibilities between an HLA interface team and a federate development team. The goal of this separation of tasks was to allow the model-builders to focus on developing system federates (and their internal behaviors) while maintaining a subset of the team with deep knowledge of the HLA-Evolved standard to help with integration. This general structure is recommended as the tasks of fully understanding the standard and developing a detailed model may be challenging for an individual team member on a university team.

Table 2.1 – Federate Interaction Table

	Other Smackdown Federates	**Mobile ISRU Plant**	**Exploratory Hopper**	**Enhanced Environment**
Other Smackdown Federates				
Mobile ISRU Plant	Resource Transfer (ISRU Plant → Lunar Rover)			
Exploratory Hopper		Scouting Report (Hopper →ISRU Plant)		
Enhanced Environment			*Resource Analysis (Hopper → Environment)*	

3.2. HLA Interface Team

HLA interface team members focused on developing an interface to the HLA function calls suitable for use in both federates. The interface included common code modules for use across both federates and federate-specific code modules spanning the start-up, simulation, and shut-down sequences of an execution. The HLA interface team also developed the modular Federation Object Model (FOM) which provided object and interaction class definitions as well as new data types to the federation.

In addition to supporting the hopper and ISRU plant federates, a testing manager federate was created to enable local testing. The primary functions of the manager federate included creating and destroying the simulation execution and providing a clocked time-regulation sequence such that the simulation progressed at wall clock time. The test manager played an important role in local debugging using a comparable execution environment.

Developing the HLA interface was an iterative process that strongly built on previous examples and tutorials. The team approached the problem by digesting as many examples as possible, line-by-line, while prototyping sample federates, slowly building up complexity and features. Often documentation or comments were insufficient and many examples included verbose exception handling which, while important, distracted from early learning. Above all, early iterative development including a "demo or die" mentality contributed to success in this area, though a tutorial-based example to build a simple federate from scratch would be a major improvement to the learning process.

3.3. Federate Development Team

The federate development team members focused on creating the simulation models and user interfaces for the ISRU plant and hopper federates and coordinating the HLA interactions with the HLA interface team members. Due to the differences between the two federates, the workflow between the hopper and ISRU plant federates differed substantially.

The ISRU plant federate team focused on developing a tele-operated model of an ISRU plant. A graphical user interface served as the inputs to command the federate during execution. Due to technical challenges related to the implementation platform (MATLAB) discussed in the following section, this aspect required significant development, testing, and iteration.

The hopper federate team focused on reusing components from an existing model. Much of the development and testing could take place locally and at accelerated speeds as the hopper did not have strong dependencies on other federates (due to the as-implemented coupling of the hopper and enhanced environment models). To support the local simulation, equivalent non-HLA function calls were developed. Finally, because of the automation, the hopper federate could implement a simpler visualization rather than a full user interface that would require active callback functions.

Although not a component of the final models, there was also significant effort put into the enhanced environment federate. The goal was to provide surface topology and resource concentration distribution information to the federation, however due to complexities with data storage and transmission, efforts were abandoned in favor of integrated function calls with planar surface topology and randomized resource distributions within the hopper federate.

3.4. Technical Risk Management

Introductory Smackdown materials identified five levels of participation of increasing difficulty to help guide team development. At each level, several key steps are identified to achieve the objectives. Though the MIT team strived to reach the highest level of participation from the start, critical milestones were established at each level to mitigate the risk of transitioning to the next level of participation. Fortunately, no irresolvable problems were encountered during the ascent to full participation.

Level 1: Passive Observer. The role of a Level 1 participant is to attend the Smackdown execution and understand the events but in a passive sense. The steps required to achieve this level of participation include: a) understanding of the Smackdown event and objectives, b) understanding of the HLA-Evolved standard, c) installation of software components, and d) execution of bundled tutorials.

The major challenges to achieving Level 1 participation were centered on the installation and configuration of software. The main areas included: installation dependencies, cross-platform inconsistencies, and network/RTI configuration. Due to the complexity of multiple layers of software, installation dependencies can quickly become a challenge for new users. To mitigate this issue, a quick-start tutorial was developed to give a step-by-step installation for all software packages. Differences between platforms (Windows 7 versus

Windows XP, 32-bit versus 64-bit), however, contributed to some errors that were difficult and time-consuming to debug. Finally, there were a few network and RTI configuration changes (e.g. turning off firewall, license server address, response interval overrides, etc.) which required additional work to accommodate.

To help teams achieve Level 1 participation, the MIT team now suggests developing a Smackdown-wide tutorial with step-by-step installation instructions for all required software components through the successful test of a bundled example for all RTIs, including execution across several computers on a local network. The guide should be updated for each software release and managed for platform-dependencies where appropriate.

Level 2: Active Observer. The role of a Level 2 participant is to use a provided federate to actively join, observe, and interact with the simulation execution. The steps required to achieve this level of participation include: a) accessing the Smackdown Subversion (SVN) repository, b) compilation of "easy button" code, c) local testing and execution, and d) remote testing and execution over the virtual private network (VPN).

The major challenges to achieving Level 2 participation were related to local and remote testing and execution. A challenge encountered with local testing is that without active participation by other federates (most notably the reference frames), only limited testing can be completed. In addition to the configurations required in Level 1, the use of the VPN for remote testing requires additional configuration work. Also, the VPN used for the Smackdown testing also restricted network connections as a security requirement, though this affected the use of SVN, Skype, and other collaborative tools during testing sessions.

To help teams achieve Level 2 participation, the MIT team suggests focusing on one implementation of an "easy button" federate and providing extensive documentation and tutorials. Additionally, it would be helpful if there were an accessible Smackdown server continuously running an execution for remote connection and testing. Since university smackdown teams are globally distributed this server should run around the clock.

Level 3: Subsystems Development. The role of a Level 3 participant is to create additional functionality for an existing object through a new subsystem. The steps required to achieve this level of participation include: a) coordination with an external team to determine requirements, and b) compilation of altered code.

Although the MIT team did not develop a subsystem for an existing object, it would require coordination with the system developer to determine the required interfaces; however it would not necessarily include the additional HLA-related requirements of Level 4 participation.

Level 4: New Object. The role of a Level 4 participant is to create a new object instance to join in the execution of the federation. The steps required to achieve this level of participation include: a) development of the federate containing the new object instantiation, and b) integrated testing of attribute updates between federates.

The major challenges to achieving Level 4 participation center around modifications to the HLA function calls and functional changes within the provided federates. The main challenge with modifying the HLA function calls relates to the lack of input validation – it is easy to incorrectly specify data formats (encoding/decoding) or omit important functions that support the execution such as object registration and time regulation. Additionally, as the mission of the new object is defined, it will likely require programming or control from a combination of scripts or user interfaces, both of which require substantial development and debugging efforts.

To help teams achieve Level 4 participation, the MIT team suggests incrementally developing federates by slowly adding new attributes and features while maintaining both local and remote testing to verify correct data transfer between federates. Debug logs, outputs, or interfaces help to identify the origins of problems during an execution. Referencing previous years' federates may serve as a helpful advantage for defining programmatic or user control and reference to important HLA function calls.

Level 5: New Class. The role of a Level 5 participant is to create a new federation object model (FOM) module defining a new object class and instantiating a new object instance to join in the execution. The steps required to achieve this level of participation include: a) development of an extended FOM module, b) development of the federate containing the new class object instantiation, and c) integrated testing of attribute updates and interaction classes between federates.

The major challenges to achieving Level 5 participation relate to the development of new FOM modules and the associated changes to Level 4

federates. As a FOM module is used across an entire federation, care must be taken to structure the definitions in a flexible way. There must be discussion as to the structure of the FOM module, and it must be "frozen" with sufficient lead time to accommodate the dependent coding by other federates.

To help teams achieve Level 5 participation, the MIT team strongly suggests thoroughly understanding the HLA-Evolved standard, especially with respect to data structures and object model templates, and communicating between teams well in advance of implementation. Furthermore, knowledge of object-oriented programming methods is helpful for developing flexible FOM objects and interaction classes.

3.5. Development Timeline

In conjunction with the levels of participation, the MIT team followed a timeline to accomplish each goal sequentially.

The official project kick-off took place on January 11, 2011 with an overview of the Smackdown event and high-level goals. During the first week of work an initial list of federates was developed. The second week was focused on installing and configuring software, including accessing and running the "easy-button" federates. By early February the MIT team had achieved Level 2 participation.

Final federate descriptions and a concept of operations were completed on February 11, followed by the development of the first federate prototypes the following week. The early prototypes were based on existing models with limited interactions, reaching Level 4 participation.

A draft FOM providing definitions for the hopper and ISRU plant object classes, resource transfer, scouting report, and text message interaction classes, and associated data types was completed on February 18 and discussed and approved in the following week.

Local testing, including primary development of the federate models took place in late February to early March, and integrated testing started March 11 initially between the lunar rover and ISRU plant. Over the following weeks leading up to the Smackdown event on April 6, 2011, many hours of integrated tests took place between several teams and using both the Pitch and MAK RTIs to secure the goal of Level 5 participation.

4. Technical Implementation

This section will describe the technical implementation of the lunar resource economy model as described in the previous sections. In order to develop this model, two federates were created to interact within the prescribed environment. This section will first describe what developments were necessary for each federate, followed by a description of the implementation details specific to each federate.

4.1. MIT Federation Object Model (FOM)

To enable customized behavior including object classes and attributes, interaction classes and parameters, and new data types, the MIT team developed a modular federation object model (FOM) to support its federates and basic logistics interactions. This section describes the MIT FOM components including motivations for their implementation.

Two object classes – MobileIsruPlant and ExploratoryHopper – inherit from the PhysicalEntity class from the Smackdown-provided Entity FOM module. The MobileIsruPlant has one attribute (isru) which defines the resources that have been produced and are available for transfer. ExploratoryHopper has one attribute (propellant) which defines the amount of available propellant to perform propulsive maneuvers.

The MIT FOM module also defines three interaction classes. TransferResources allows the instantaneous transfer of resources between two objects. It is parameterized by the delivering entity name, the receiving entity name, and the resources to transfer. ScoutingReport announces the detected resource concentrations at a specific position. It is parameterized by the sending federate name, the position at which the scouting occurred, and the detected resource concentration (mass fraction). Finally, the TextMessage interaction class enables the communication between federates as no internet connectivity will be available during simulation. It is parameterized by the sending federate name, the receiving federate name, and the message body.

The MIT FOM module also defines a few data types to help manage resource representation. The data types are based on previous work in the field of space logistics which focus on a functional class of supply taxonomy to characterize resource types in support of space exploration [4]. This representation of

resources has previously been implemented in the SpaceNet space logistics analysis tool [5-6].

The ResourceType enumeration defines the different types of allowable resources. For the time being, there are six available resources: isru_1, isru_2, propellant_1, propellant_2, other_1, and other_2. Specific names are not yet assigned to each resource type because of the uncertainty in interaction between federates and an unknown level of analysis fidelity. The other additional data type is Resource, which is a fixed record comprised of a ResourceType and Mass. This represents a realized amount of a specific type of resource and is used in MobileIsruPlant and ExploratoryHopper object classes as well as the TransferResource and ScoutingReport interaction classes.

4.2. MATLAB Platform

Based on team members' experience with the MATLAB computational environment for scientific computing, the MIT team leveraged the HLAToolbox as provided by a sponsoring company ForwardSim Inc to implement the Smackdown federates. Its use greatly accelerated the learning curve to get examples running by providing a familiar environment and code syntax.

The usage of MATLAB with the HLA architecture resulted in a number of challenges that needed to be overcome. The set of difficulties encountered and their solutions are as follows:

Handling of Callbacks. Using the HLAToolbox, a runtime environment may be implemented as a spin-while loop checking for callbacks. This poses some troubles for the implementation of other things, but the resolutions to those problems are described below.

Single Threaded Software. Unless specific toolboxes are used, MATLAB runs only with one thread. The only exception to this is a GUI, which will run in its own thread; however it is tightly coupled with the computational thread. As a result, in order to run the backend RTI interface and be able to accept input from the user, it is necessary to first initialize the RTI backend, and then initialize the GUI, while passing the GUI handles to the script that runs the RTI callback loop. During execution, the wait function was used to relinquish control from the computational thread (often in spin-while loops waiting for RTI function callbacks) to resolve user input.

Debugging. As MATLAB scripts use weakly-typed variables and are not compiled, errors must most often be debugged during test execution. Due to the nature of exception messages and stack traces, it can often be difficult to track down the root cause of errors during execution.

Latency Difficulties. To ensure that all federates ran at the same speed, the federation manager needs to regulate the time at which different federates step through the simulation. This system, however, is not robust in its current implementation, so it is necessary to check on the federate side to ensure that timings are being respected. This is done using the tic/toc MATLAB functions.

4.3. Reference Frames

The Smackdown-provided environment federate provides state information for several reference frames using three-dimensional Cartesian coordinates. For lunar surface federates the most relevant reference frame is Lunar Centric Fixed, which has an origin at the center of the Moon and rotates concurrently. Although this frame works well for airborne vehicles, it is difficult to work with for surface vehicles which must remain attached to the two-dimensional surface. As a result, the MIT team implemented two internal reference frames – Selenographic and Local Cartesian – with associated transformation functions, as shown in Figure 1.

Lunar Centric Fixed (LCF) is the global inertial coordinate system updated by the Smackdown-provided federates. The X-axis intersects at the prime meridian and equator, the Y-axis 90 degrees east, and the Z-axis at the Lunar North Pole.

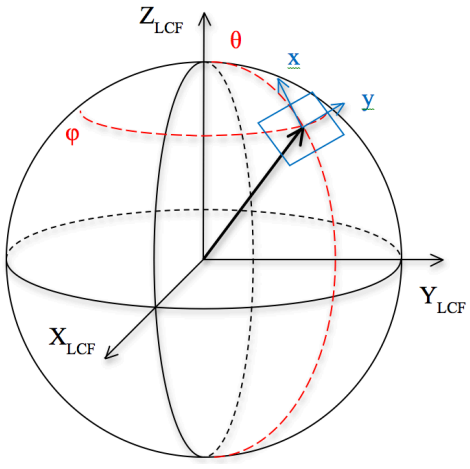

Figure 1: MIT Smackdown Coordinate Frames – Lunar Centric Fixed (black), Selenographic (red), Local Cartesian (blue)

The Selenographic coordinate system is a standard representation for surface locations on the moon, consisting of latitude, longitude, and altitude. The zero-degree latitude is the lunar equator and the zero-degree longitude is the lunar prime meridian.

A local Cartesian system is the simplest method for manipulating vehicles traversing the lunar surface using an observer frame. However, due to the curvature of the surface it is inaccurate over long distances (i.e. the points will not be attached to the surface). The directionality selected in this example is with the x-axis pointing north, the y-axis east, and the z-axis towards the center of the Moon. In the MIT federate implementation the origin of the Local Cartesian frame was the supply depot federate, which – unlike the ISRU plant – was assumed to be stationary.

The conversion between the Lunar Centric Fixed and Selenographic coordinate frame is a simple implementation of spherical coordinates with the first rotation about the Z-axis and the second rotation about the Y'-axis. The conversion between Lunar Centric Fixed and the Local Cartesian frames depends on a simple coordinate transformation. Both transformations (in both directions) were implemented using common functions in MATLAB.

Although the Local Cartesian frame provides an approximate plane of the lunar surface over short distances, it does not consider local features such as craters, rilles, or hills. Initial development efforts sought to develop an enhanced environment federate to provide altitude information, however the efforts were not completed due to difficulties of transferring information across federates. Future Smackdown executions should implement a terrain model federate to provide altitude and gradient information for regions on the lunar surface.

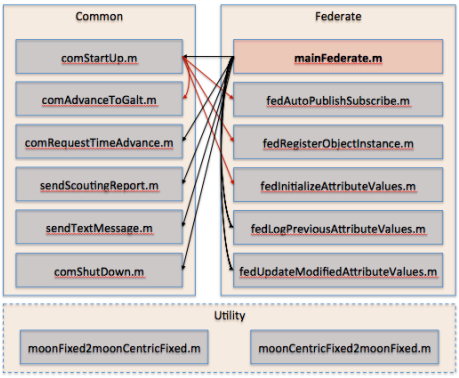

Figure 2: HLA m-file Functions – Main Script (red), and Supporting Files (blue)

4.4. HLA Interface

To help support the HLA interactions and provide a consistent interface across both MIT federates a set of common HLA functions were developed, shown in Figure 2 (bottom left). These included functions for start-up (connect to the RTI), request time advancement, and shut-down (disconnect from the RTI). The start-up and shut-down common functions also accessed federate-dependent functions to register objects, publish and subscribe to attributes and interactions, and initialize attribute values.

4.5. Hopper Federate

As described in the model section, the purpose of the hopper federate is to scout the area surrounding the lunar base for resources. In order to do this, it would traverse to a particular location (randomly chosen given lack of detailed geographic resource data), examine the resources in that location, and then broadcast its results to the federation. After this, it would move to a new location and repeat for as long as fuel is available onboard the hopper. In this implementation, the motion of the hopper is automated. This subsection will describe how the model was implemented, what the graphical user interface (GUI) looked like, challenges encountered, and future suggestions for the hopper federate.

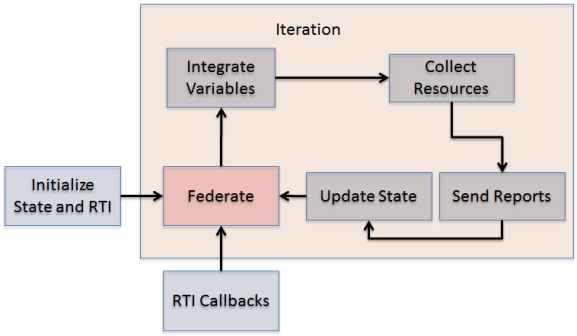

Figure 3: Hopper Flowchart Diagram

In a particular run, the hopper's state is initialized and then the federate is initialized, which is then passed to the RTI loop, where the state variables are integrated and resources are collected (if relevant for that time step). Then, scouting reports are sent and the state is updated. The federate is also responsive to RTI callbacks.

The user interface for the Hopper is relatively simple. It consists of shell output, a GUI, and visualizations including a map overlay and graphs of resource detection.

The GUI itself is as shown in Figure 4, which simply allowed the user to terminate the federate (the "Stop Simulation" button) and reset the hopper to its initial conditions (the "Reset Hopper" button). This GUI was needed as there was no end condition to signal an automated program when to stop simulation (other federations implement a simulation manager, SIMAN, for this purpose).

Figure 4: Hopper GUI

The shell output provided text-based updates at each time step of the current simulation time and a status report of the hopper, which can be any of:

- Hopper excavating, which is when it is checking the resources of a particular location
- Scouting report, which provides a report of resources at a particular location
- Hopper moving, which is when it is moving to the next location
- Distance to target, which is in meters
- Fuel percentage remaining

Finally, the hopper visualizations are shown in Figure 5. The top plot shows the trajectory of the hopper with the blue line and the current target position with the circle. The bottom plot shows the quantity of resources that were located at the last scouting site, with the break-even threshold indicating whether it is worth it for the ISRU plant to move to that location.

Although a hopping vehicle is a complex system itself, the simulation model used for the Smackdown competition was a simplified version. It functioned very well as a first-order demonstration of the logistics of hopping as a means of exploring the Moon and scouting for resources. Future versions of such a hopping model should include mode realistic and complex simulations of the hopper's motion. Additional upgrades to the hopper model could include an expanded GUI with more detailed control over the phases of the hopper's operation by users (although the flights themselves should remain automated), as well as more interaction between the hopper and the environment, or between the hopper and other vehicles. As a scout, the hopper could generate detailed terrain data and deliver that to the mobile ISRU plant, or even to other federates, which might be simulating science vehicles. Direct communication of the resources discovered by the hopper to a multitude of other federates (instead of just one mobile ISRU plant) could also help stimulate competition to retrieve the resources, which would further jumpstart the lunar resource economy. Furthermore, hopper refueling might also be implemented in the future. Ultimately, however, the hopper has to detect more resources than it consumes for itself.

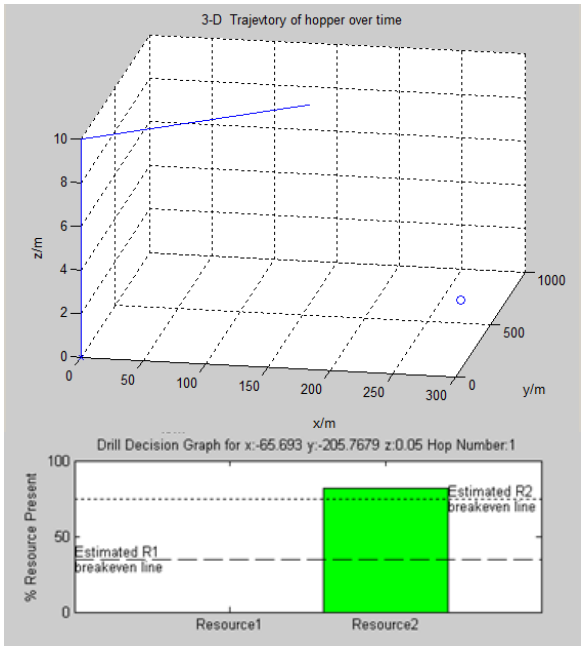

Figure 5: Hopper Visualizations

The hopper model made visible both the challenges and strengths of HLA. Challenges posed by the hopper were mainly as a result of the programming language limitations of MATLAB. In some cases this was managed by the use of many additional manager variables that allowed for the performance of logical checks. For instance, the hopper ran at different time resolutions than the rest of the simulation to improve the accuracy of traversal calculations. As such, one of the initial difficulties was ensuring the hopper only changed states at specific time steps to agree with the main simulation. Additional management state variables ensured that the simulation ran smoothly and the model only changed states as described. These complications can also be related to the choice and strength of the RTI used. The hopper also showcased one of the greatest strengths of the HLA standard – modularity and the ability to independently develop federate models and later

insert HLA interfaces to create a fully functional simulation. The MIT team took advantage of this and used the hopper model to explore different scenarios of mission planning phases as well as different degrees of details in the fulfillment of hopper requirements in relation to the actual TALARIS project. This was done without much fear of breaking the holistic structure of interactions with other federates or the simulation as a whole. It would therefore be advisable for future teams to consider more closely the options of modularity and which federates need to be deeply infused with the HLA standard.

4.6. ISRU Plant Federate

As described in the model section, the purpose of the ISRU plant federate is to receive scouting reports from the hopper (or hoppers) described in Section 4.5 and to then decide at which locations it should commence resource extraction. The ISRU plant is mobile, but moves much slower than the hopper, so it is critical to minimize travel time so as to maximize resource collection time. In this implementation, the motion of the ISRU plant is user-controlled. This subsection will describe how the model was implemented, what the graphical user interface (GUI) looked like, challenges encountered, and future suggestions for the federate.

Since the ISRU plant received inputs from both the GUI and the RTI, the flow of information through the ISRU plant is as shown in Figure 6. The federate state and GUI are initialized and then a loop iterates to integrate the state variables, update the GUI, and finally update the federate itself at each time step. Furthermore, the federate is responsive to inputs from both the GUI and the RTI. The ISRU plant subscribes to the hopper scouting reports and the federate object publishes its state variables to the federation.

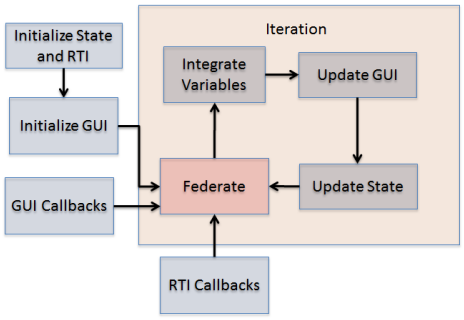

Figure 6: ISRU Plant Diagram

The purpose of the ISRU Plant GUI is twofold: to show the user the current state of the plant and to allow the user to manipulate the plant. The primary dashboard the GUI is as shown in Figure 7, with components as follows:

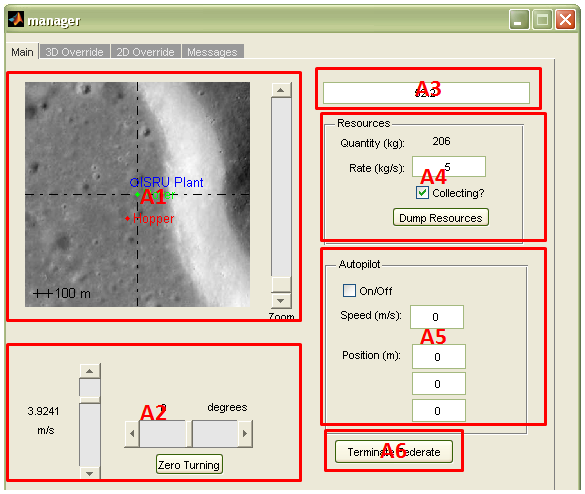

Figure 7: ISRU Plant Main Panel

The A1 box provides an overhead view of the position of the ISRU plant (and hopper) and its targeted location. The bar on the right side may be used to zoom to different view levels. The A2 box shows the manual driving controls: the vertical bar allows the user to adjust the throttle and the horizontal bar allows the user to steer the plant. The "Zero Tuning" box sets the vehicle to go straight again. The A3 box displays the current federation time. The A4 box shows the current resource quantity in storage and allows the user to manually tune the collection rate and if the plant should currently be collecting; in future revisions, the collection rate will no longer be user-defined. Furthermore, the user may choose to dump current resources. The A5 box provides the autopilot controls, which allows the user to specify a speed and waypoint. The A6 box highlights the "terminate federate" button, which closes the federate cleanly.

The secondary panes of the ISRU GUI can be seen in Figure 8 and Figure 9, which allow direct manipulation of the variables. Box B1 indicates the manual override, which allows these fields to be edited. Once B1 has been toggled, the fields inside the B2 and B3 boxes may be edited. The same is true for the 2D pane, where the C1 toggle allows the C2 fields to be edited. It should be noted that, as long as the toggles are active, the values in the fields on the panes will drive the plant's location and speed, so autopilot and steering functionality is disabled.

In addition to the generic difficulties with HLA and MATLAB, the creation of the ISRU model posed

difficulties with the need for a more advanced GUI, since it required more human interaction. With the framework now in place, more extensive manual controls and algorithms may be implemented, and more advanced autopilots, including waypoint capability, should be added. Furthermore, resource acquisition and release is currently entirely manual; automation must be incorporated and customized to meet the needs of a particular environment and set of other federates. For example, the model should ensure that collection can only happen at a valid rate at a valid site and that resources are transferred to other federates rather than simply being "dumped."

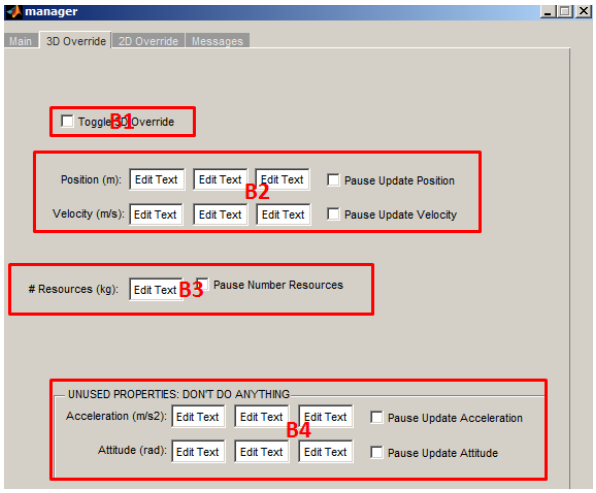

Figure 8: ISRU Plant Secondary Panels

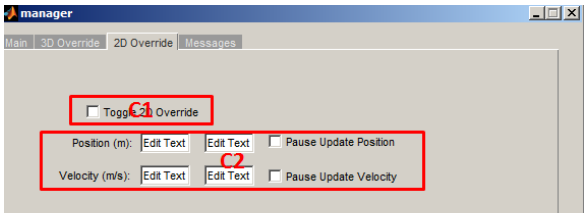

Figure 9: ISRU Plant Secondary Panels

5. Conclusions

The NASA/SISO 2011 Smackdown event was hugely successful with respect to university outreach. Many schools both national and international participated and gained early and useful exposure to professional M&S through the HLA standard. Students were challenged to apply themselves in a field where they possessed only limited knowledge with the aim of showcasing the difficulties and possibilities of modeling and simulation. Additionally, for the MIT team, work done earlier on the hopper federate had the cross-benefit of advancing research for detailed modeling of hopping vehicles and scoping of their missions. The need to model a hopper carefully within the confines of the Smackdown competition helped the hopper federate development team investigate ways to ground a hopping vehicle model in reality, which has applications in related work being undertaken by team members. Beyond university outreach, the Smackdown event harbors the potential to explore the boundaries of M&S as applied to real world engineering solutions. Thus it is beneficial to take a closer look and suggest ways to improve the event as a whole, including advice for future teams, by focusing on the structure of the Smackdown event, the use of the HLA standard itself, and the importance of project management.

The success of the event may also be attributed to the complexity, openness, and overall the collaborative approach many teams and the NASA/SISO organizers used. In fact, the Smackdown event presented itself more as a collaborative effort than a competition. This ties in with the HLA standard's attributes of interoperability. That is, based on the features that support interoperability of federates and interactions with the RTI, the MIT team found the standard to be more suited to collaborative use than competitive use. Competitive events, however, tend to motivate team participation through prizes. One suggestion for future Smackdown events would thus be to clarify the focus of the event as a competition or collaboration. In 2011 the event was announced as a competition but drifted to become a collaboration. A possible scenario based on the lunar fuel economy example mentioned in Section 2 of this paper could use HLA to allow teams to compete for resource acquisition on the lunar surface and then collaborate in its sharing and maintenance to develop the fuel economy.

The HLA standard played a critical role in the success of the competition. Competing teams and members were both privileged by and at the mercy of HLA's strengths and weaknesses. As mentioned, the greatest hurdle was the learning curve for the standard and the relative applicability of the standard for different team members. The standard is extremely capable and, as a result, is quite dense in its content, thus making it difficult for first-timers to begin to focus on their desired applications early in the process. However the complexity is countered by the fact that not every aspect of the simulation required an HLA interface during production, as such certain federates could be developed and tested without knowledge of the intricacies of the HLA standard.

It would thus be beneficial to provide teams with a tutorial-based example to build a simple federate from scratch, one that highlights the most common HLA interfaces that may be used and also provides step-by-step installation instructions for all required software components. This would allow for a greater level of modularity in the production of the model while reducing the time between learning and federate development. Details should also be provided to teams as to the levels of interaction needed by team members with the standard. The benefits and shortcomings of the development platforms used (Java, MATLAB, C++) should also be mentioned, to allow teams to assess which platforms would be most applicable to their objective. This is important, since the development and implementation of federates and the simulation as a whole is either limited or enhanced by the choice of platform.

The role of project management as applied to the development of the simulation could also be better presented initially to teams. There was no explicit mention of the significance and advantages of well-implemented project management practices that could be adopted by teams as they collaborate among federates and with other teams. The MIT team, however, found such efforts of great utility and was able to use them to improve the productivity of federate development. For instance, weekly testing sessions between the MIT and NASA teams greatly assisted the establishment of the FOM as well as reduced the effect and downtime incurred from code bugs during the federate development process.

It would thus be in the interest of the Smackdown event to prescribe some form of management guidelines that teams may follow. For instance, teams could internally produce a work breakdown structure to identify roles within their team and allow them to recognize their individual need for exposure to the HLA standard. For greater effect, the submission of a team report to document simulation development as an additional requirement, may also aid students in focusing on the steps taken as they develop their simulations for the event. This will also enforce a sense of risk mitigation, which will control the extent of coding features/capabilities that a team attempts to generate. A reporting system would also produce archival information for future years' participants and may yield materials suitable for conference presentations and scientific journal articles.

6. References

[1] *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Framework and Rules*, IEEE Standard 1516, 2010. doi: 10.1109/IEEESTD.2010.5553440

[2] Cunio, P.M., Nothnagel, S.L., Lanford, E., McLinko, R., Han, C.J., Olthoff, C.T., Hoffman, J.A., and Cohanim, B. E., "Further Development and Flight Testing of a Prototype Lunar and Planetary Surface Exploration Hopper: Update on the TALARIS Project," AIAA-2010-8635*, AIAA Space 2010 Conference and Exhibition.,* Anaheim California, Aug. 30-2, 2010.

[3] Cunio P. M., Alibay F., Meira P., Sheerin T., Lanford E., Krupczak E., Hoffman J. A., "Options in the Solar System for Planetary Surface Exploration via Hopping", IEEE Aerospace Conference 2011, Big Sky, MT.

[4] Shull S., Gralla E., de Weck O., Shishko R., "Future of Asset Management for Human Space Exploration: Supply Classification and an Integrated Database", AIAA-2006-7232, AIAA Space 2006 Conference and Exposition, San Jose, California, September 19-21, 2006.

[5] de Weck O.L., Simchi-Levi D., Shishko R., Ahn J., Gralla E., Klabjan D., Mellein J., Shull A., Siddiqi A., Bairstow B, Lee G., "SpaceNet v1.3 User's Guide", NASA/TP-2007-214725, January 2007.

[6] Grogan P.T., *A Flexible, Modular Approach to Integrated Space Exploration Campaign Logistics Modeling, Simulation, and Analysi*s, S.M. Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, September 2010.

Author Biographies

NANA ESSILFIE-CONDUAH is a Junior Programmer Analyst at Tessella Inc, Newton, MA. Nana graduated from MIT with a B.S. in Aerospace Engineering in June 2011. Nana has been a member of the MIT satellite team whose research saw the design and development of the CASTOR nanosatellite to test the Diverging Cusped Field Thruster (DCFT) technology. He was also a member of the TALARIS team and aided in the testing of the hopper vehicle. As a representative from the TALARIS team Nana was responsible for the coding and testing of the MIT Hopper federate in the SISO Smackdown competition.

PAUL GROGAN is a Ph.D. student in the Engineering Systems Division at MIT. His research

interests include modeling and simulation, space logistics, software engineering, and education. His doctoral work investigates the use of interactive simulations to improve learning, communication, and ultimately, design for complex systems.

PHILLIP M. CUNIO is a Ph.D. student in the Aeronautics and Astronautics Department at MIT, with a research background spanning space systems, system architecting, Mars exploration, and human decision support system development. He earned bachelor's degrees in Aerospace Engineering and Mechanical Engineering at the University of Florida, and a master's degree in Aeronautics and Astronautics at MIT. He is currently a senior graduate student on the TALARIS project, working with Draper Lab and MIT's Space Systems Lab to develop and operate prototype planetary surface hopping vehicle technology.

RYAN MCLINKO holds a B.S. in Aerospace Engineering from MIT. He is currently pursuing a Masters degree in the Aeronautics and Astronautics Department, after which he will join Sierra Nevada Corporation's Dream Chaser program. His primary area of interest is in structural design and analysis as well as systems integration, particularly as it relates to decreasing the cost of access to space.

OLIVIER DE WECK is an Associate Professor of Aeronautics and Astronautics and of Engineering Systems at the Massachusetts Institute of Technology (MIT) and an Associate Fellow of AIAA. He is an expert in strategic engineering of complex systems (http://strategic.mit.edu) and in multidisciplinary optimization. He has over 15 years of academic, professional and military experience in Aerospace Engineering. He obtained a graduate degree in industrial engineering from the Swiss Federal Institute of Technology (ETH Zurich) in 1993, and S.M. (1999) and Ph.D. degrees (2001) in Aerospace Systems from MIT. He is affiliated with the Space Systems Laboratory (SSL) at MIT and serves as an Associate Editor for the Journal of Spacecraft and Rockets.