Designing a Distributed Space Systems Simulation in Accordance with the Simulation Interoperability Standards Organization (SISO)

Benjamin Cowen
North Carolina State University, Raleigh NC 27607

Simulations are essential for engineering design. These virtual realities provide characteristic data to scientists and engineers in order to understand the details and complications of the desired mission. A standard development simulation package known as Trick is used in developing a source code to model a component (federate in HLA terms). The run-time executive is integrated into an HLA based distributed simulation. TrickHLA is used to extend a Trick simulation for a federation execution, develop a source code for communication between federates, as well as foster data input and output. The project incorporates international cooperation along with team collaboration. Interactions among federates occur throughout the simulation, thereby relying on simulation interoperability. Communication through the semester went on between participants to figure out how to create this data exchange. The NASA intern team is designing a Lunar Rover federate and a Lunar Shuttle federate. The Lunar Rover federate supports transportation across the lunar surface and is essential for fostering interactions with other federates on the lunar surface (Lunar Shuttle, Lunar Base Supply Depot and Mobile ISRU Plant) as well as transporting materials to the desired locations. The Lunar Shuttle federate transports materials to and from lunar orbit. Materials that it takes to the supply depot include fuel and cargo necessary to continue moon-base operations. This project analyzes modeling and simulation technologies as well as simulation interoperability. Each team from participating universities will work on and engineer their own federate(s) to participate in the SISO Spring 2011 Workshop SIW Smackdown in Boston, Massachusetts. This paper will focus on the Lunar Rover federate.

Nomenclature

DCM	= Direction Cosine Matrix
DSES	= Distributed Space Exploration Simulation
FOM	= Federation Object Model
HLA	= High Level Architecture
IEEE	= Institute of Electrical and Electronics Engineers
ISRU	= In-Situ Resource Utilization
MIP	= Mobile ISRU Plant
NASA	= National Aeronautics and Space Administration
OMT	= Object Model Template
RTI	= Run-Time Infrastructure
SISO	= Simulation Interoperability Standards Organization
SIW	= Simulation Interoperability Workshop
UAH	= University of Alabama at Huntsville
USRP	= Undergraduate Student Research Program
V	= velocity
VPN	= Virtual Private Network
ρ	= Radius of the moon
q_0	= Scalar element of quaternion
q_1	= 1st element of quaternion vector

q_2 = 2nd element of quaternion vector
q_3 = 3rd element of quaternion vector
φ = Longitude in degrees
θ = Latitude in degrees
x = X position in Moon Centric Fixed Reference Frame
y = Y position in Moon Centric Fixed Reference Frame
z = Z position in Moon Centric Fixed Reference Frame
$X_{a/b}$ = Position of a with respect to b
X_a = Position of a
X_b = Position of b

I. Introduction

This paper will show how important simulation tools are in developing effective missions. It is important for all of the NASA centers as well as industry to be involved in creating simulations so that the systems and subsystems of a simulation be spread out to the people who have technical skills in that area. A distributed simulation allows for these capabilities. This type of simulation is being used to increase simulation interoperability, the ability of different systems to work together. The SISO Simulation Smackdown is an effort to convey the importance of modeling and simulation to collegiate students by getting them involved in creating their own HLA based distributed space simulation. A distributed simulation allows people from all over the world in industry and academia to interact with each other in real time. The Lunar Rover is part of the DSES project, collaboration between participants to simulate complex space systems that support NASA's Space Exploration Initiative. This year, the SISO Smackdown was dedicated to Keio University who could not attend the conference due to the catastrophic events in Japan.

II. Equations (Paul Tipler, 2003)

Equations used to calculate position in the moon centric fixed reference frame given an initial latitude and longitude:

$$x = \rho \sin\theta * \cos\varphi$$
$$y = \rho \sin\theta * \sin\varphi$$
$$z = \rho \cos\theta$$

Instantaneous delta V is used to make the rover move. Runge Kutta 4 integration is used to propagate the state. Once the new position is calculated in the moon centric fixed reference frame, the equations to calculate the new latitude and longitude are:

$$\theta = \frac{\pi}{2} - \cos^{-1}\frac{x}{\rho}$$
$$\varphi = \tan^{-1}\frac{y}{z}$$

Relative position is used in order to calculate how far a certain destination might be:

$$X_{a/b} = X_a - X_b$$

In order to calculate the Lunar Shuttle position, a quaternion has to be used. This is because the Lunar Rover operates under the moon centric fixed reference frame and if it were to receive the state of the Lunar Shuttle without making any changes, it would appear the Lunar Shuttle would continue to move even after it had landed (this is due to the rotation of the moon). To overcome this, the quaternion below was used:

$$DCM_{Intertial\ to\ fixed} = \begin{bmatrix} (q_o^2 + q_1^2 - q_2^2 - q_3^2) & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & (q_o^2 - q_1^2 + q_2^2 - q_3^2) & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & (q_o^2 - q_1^2 - q_2^2 + q_3^2) \end{bmatrix}$$

III. Overview

A. Scenario

The Lunar Rover federate is part of a much bigger distributed space simulation that encompasses a cargo mission to the lunar surface. The location of the simulation is at Rima Hadley, next to the Apollo 15 landing site. The reason this site was chosen is because the change in elevation is very minimal, an excellent choice for vehicles that are limited to ground travel. It is not on the side of the trench in order to preserve historic missions.

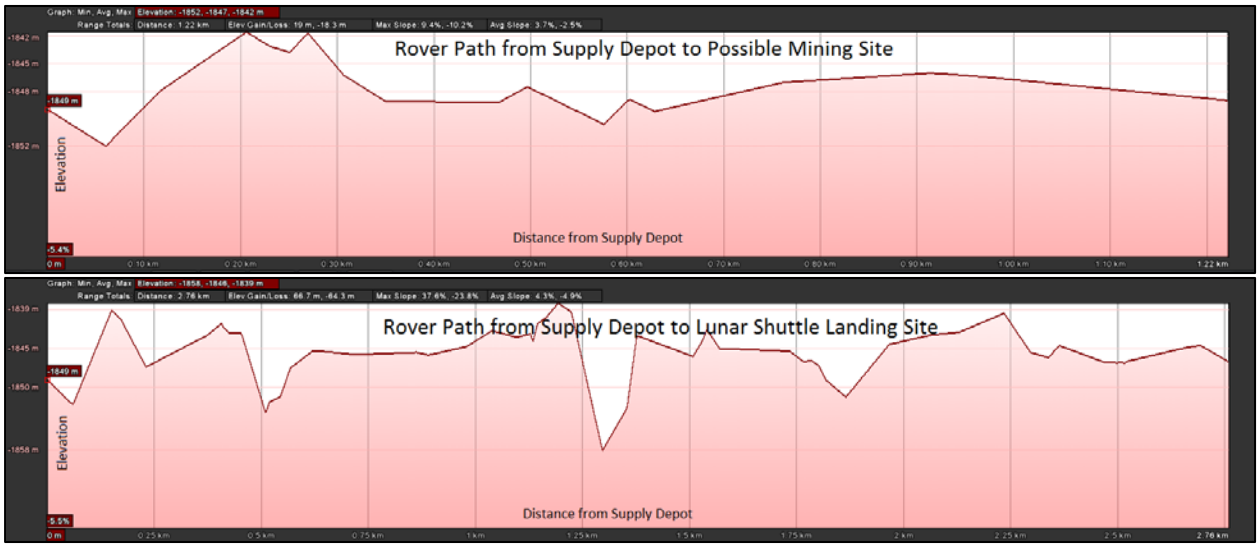

Figure 1. Elevation Profiles.

Analysis of change in elevation as Lunar Rover moves across the lunar surface. (From Google Moon)

The scenario this year involved a cargo mission to the lunar surface. The federates continually operating on the lunar surface include the Lunar Rover, Supply Depot (Combination of Earth Cargo Depot and Supply Cargo Depot), Mobile ISRU Plant, and Exploratory Hopper. The Exploratory Hopper, designed by the MIT team, was used to explore the site around the supply depot for deposits of valuable resources. Before the Lunar Shuttle arrives, the Lunar Rover only has one task and that is to transport Helium-3 from the Mobile ISRU Plant to the Supply Depot. There were also two satellites, designed by UAH for communication purposes.

Once the Lunar Shuttle arrives, the Lunar Rover defers its primary task from relieving the Mobile ISRU Plant of its cargo to completing the tasks that are related to the Lunar Shuttle. The Lunar Shuttle brings supplies from orbit to the lunar surface that is crucial for the continued operations on the moon. Once the Lunar Shuttle arrives on the lunar surface, the Lunar Rover receives an interaction telling it that the Lunar Shuttle has arrived. The Lunar Rover receives the position of the Lunar Shuttle and then begins moving to the lunar shuttle landing site. Once the rover arrives it receives cargo from the shuttle and then the rover makes its way back to the Supply Depot. After the cargo has been transferred to the Supply Depot, the rover then makes two more trips to the Lunar Shuttle. One trip is to refuel the shuttle and the other is to take Helium-3 to it. Once the Lunar Shuttle has been refueled and has received the Helium-3, it will begin its ascent back to lunar orbit to complete the mission scenario.

The Earth-Moon Transfer Vehicle starts in Earth orbit and goes to lunar orbit. However, in order to keep the simulation simple this year, the simulation simply started in lunar orbit. Next year, the simulation will be much more elaborate.

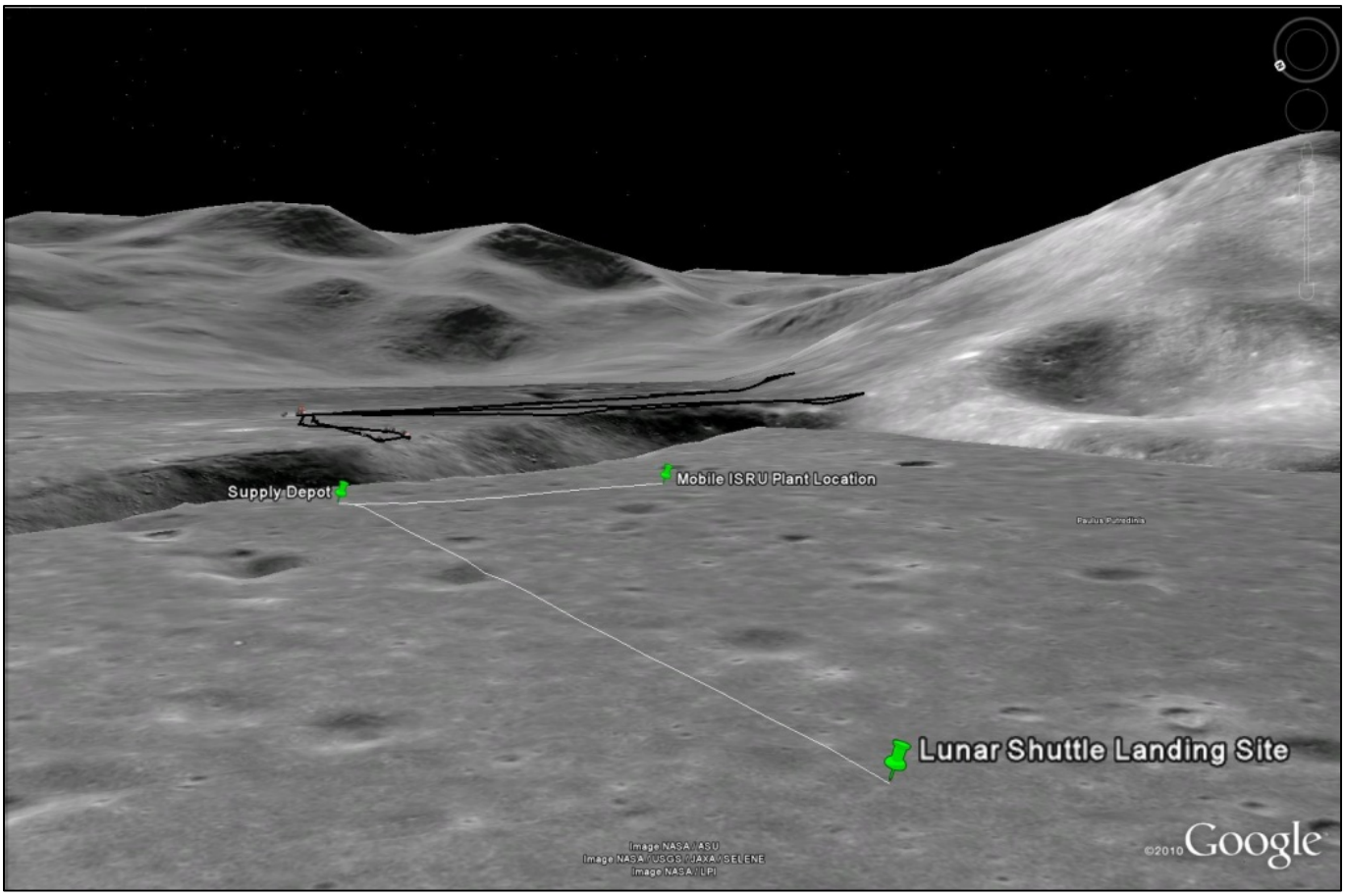

Figure 2.Visual of Federate Locations

This shows the location of the Lunar Shuttle Landing Site, Supply Depot, and one of the many locations of the Mobile ISRU Plant. The white lines indicate the path that the Lunar Rover would take. (From Google Moon)

B. Teams

There were two ways in which teams could learn about modeling and simulation concepts as the SISO Smackdown is worked on. The first way is to be an active participant. This meant that a modeler just dives right into the mix and builds a federate for the federation execution. Another way is to be an observer. This is useful for teams who do not yet feel comfortable or do not have time to build their own federate. The team can observe the first year to see exactly how the SISO Smackdown works and then next year, they can participate. These two options provide great insight into modeling and simulation techniques. The goal is to get more teams to enter the observing stage so they can begin to learn about what it means to design a federate. In a couple years the simulation should be constructed by an array of teams all over the world.

1) Active Participants

- NASA – Environment Federate and Earth-Moon Transfer Vehicle Federate
- NASA USRP Interns – Lunar Rover Federate & Lunar Shuttle Federate
- Massachusetts Institute Technology – MIP & Exploratory Hopper
- University of Bordeaux – Earth Cargo Depot and Supply Cargo Depot
- University of Genoa – Earth Cargo Depot and Supply Cargo Depot
- University of Alabama Huntsville – Two communication satellites
- Keio University (Had to drop out due to catastrophe in Japan)
- ForwardSim – 3-dimensonal viewer

2) Observers (plan to participate next year)

- Louisiana State University

IV. Simulation Software

A. **Trick** (S. Lippman, 2005)

Trick is a simulation toolkit that makes modeling much easier on the programmer. This run-time executive is designed to support real-time and non-real-time simulations and include human-in-the-loop and hardware-in-the-loop applications. Because of its ability to generate source code for the run-time input files, the programmer only has to focus on developing the code for the model. Trick has the ability to integrate its executive with that of the programmer's model.

Trick is data driven, meaning the data controls how the program flows. Trick also utilizes object-oriented programming. This programming paradigm uses data structures in accordance with interactions in order to design programs.

Features that Trick supports include:

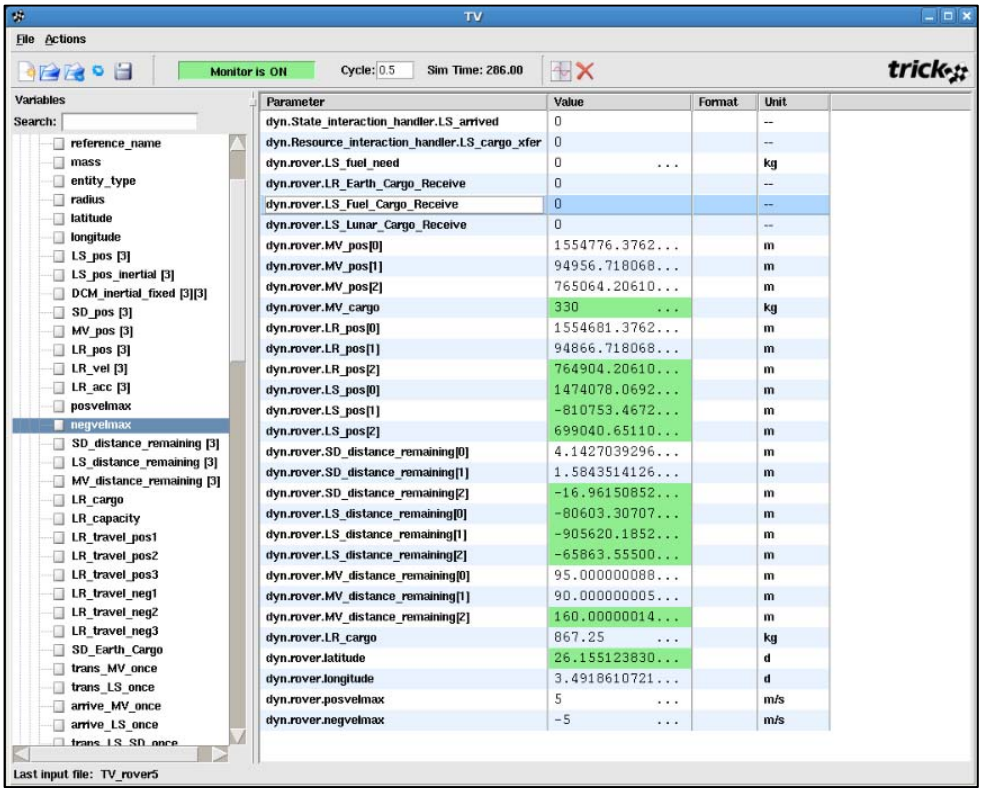

Figure 3.Trick View

Trick View is used to view variables and also allows for human-in-the-loop interaction. Highlighted in green are the variables that were changing at the second of that specific execution. Highlighted in blue is simply the variable that was being observed at the time the picture was taken.

- C/C++ programming
- Unit conversion
- Data logging
- Real-time and non-real time running capabilities
- Powerful math utilities
- Graphics
- Time based and event based scheduling
- Human-in-the-loop and hardware-in-the-loop applications

B. **TrickHLA** (Dexter, 2009)

TrickHLA is a bridge between Trick and HLA. It allows the programmer to have minimal knowledge of HLA but still be able to utilize the features that it has to offer. Because HLA takes an extended period of time to master, TrickHLA was developed. The effort of this package was to let the developer spend more time focusing on building their model rather than understanding the technical details of HLA.

Features that the NASA intern team used that TrickHLA supports include:

- C++ programming
- Send and receive interactions
- Publish and subscribe attributes
- Pack and unpack fixed record data
- Encode and decode data from RTI automatically
- Time advancement handling

- Multiphase initialization
- Opaque data
- Late joining federates

Interactions are single events that only occur one time. The interactions can be sent out to any other federate in the federation that subscribes to them. Interactions are imperative in expanding the capabilities of simulation interoperability due to the high degree of communication going on in the federation. Attributes are different than interactions in that they can be updated every second. If a late joining federate comes in, it can subscribe to an attribute and get updates just as easily as a federate that had been part of the simulation from the beginning.

Other features of TrickHLA that the NASA intern team did not use include:

- Lag compensation
- Ownership transfer

All of these features provide useful tools to allow simulations to interoperate. TrickHLA is data driven and allows Trick simulations to use the IEEE-1516 HLA without having to fully understand all of the technical details of HLA. Only Trick versions 07.7.0 or newer are compatible with TrickHLA.

C. RTI

RTI is computer software that allows software components from different simulations to connect. Programs running on different machines are allowed to interact. RTI is a reason why simulation interoperability capabilities have been expanding. The RTI's used for this project conform to the IEEE 1516 standard. The two RTI's that this project utilized were Pitch pRTI and Mäk RTI. Licenses have a limit on how many federates can join in. For the Smackdown, the license could support 20 federates even though the most that ever joined in at the same time was twelve. Without RTI, information exchange would not occur. In fact, HLA specifies that there must be an RTI with a standardized interface in order to operate. The RTI is also responsible for joining and resigning federates, sending information regarding attributes, interactions, and objects, and synchronizing time.

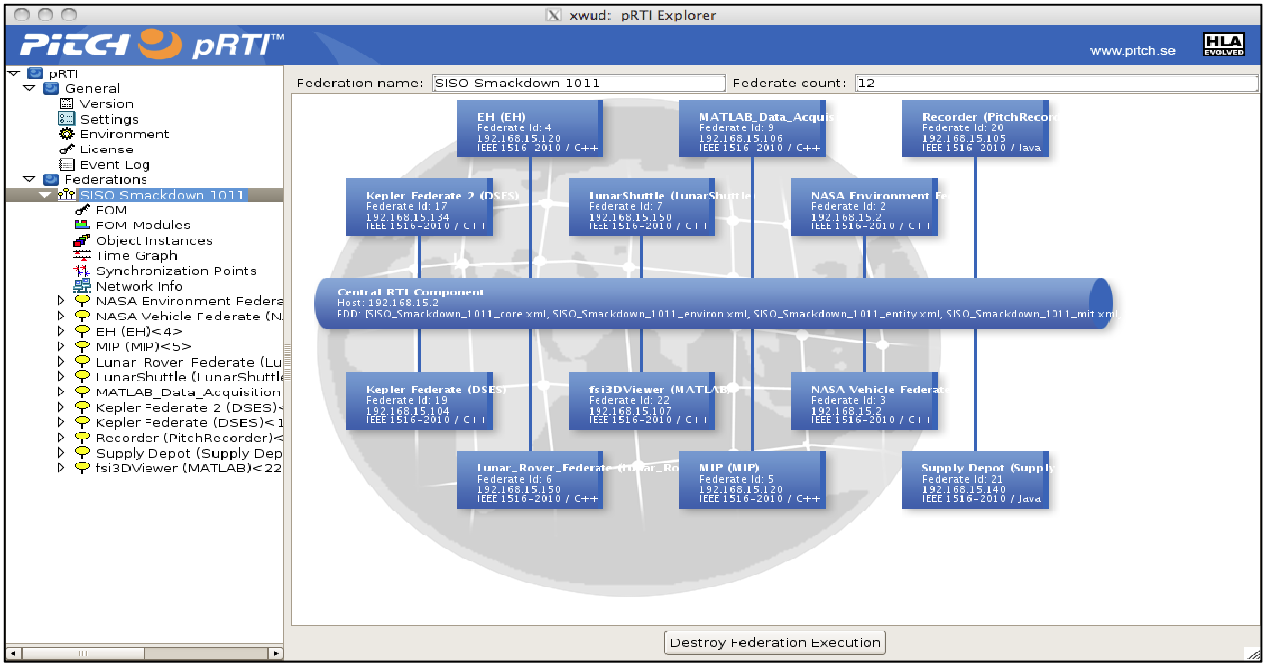

Figure 4.Pitch pRTI

This shows every federate connecting into a single federation.

D. FOM

The HLA standard consists of an OMT which is a convenient way to define object models for each federate. For this project, the graphical tool used is the Visual OMT. It allows a modeler to go in and describe what types of data will be published and subscribed and it allows the modeler to determine what type of encoding to use for each variable. This tool is used to create different object classes. For instance, the Lunar Shuttle inherits all of the properties from the Space Vehicle as shown in Figure 5.

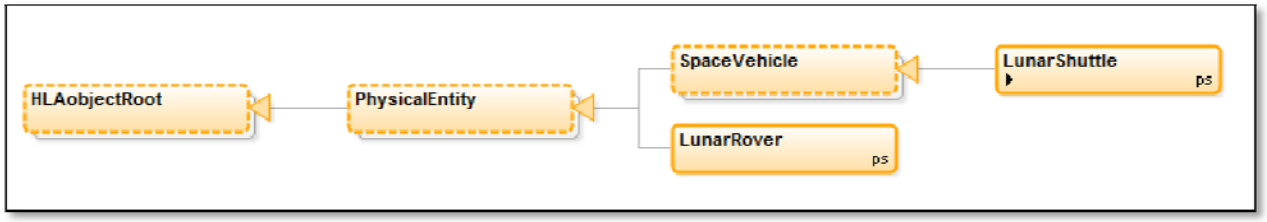

Figure 5.Object Classes

Shows the way in which each object class inherits properties from another.

FOMs are used to describe a shared object, interactions, and attributes for the federation. For the SISO Smackdown, the FOM had to be in accordance with the HLA OMT. The FOM also has an identification table in order to describe the purpose and author. Figure 6 shows the way in which FOMs are dependent on one another.

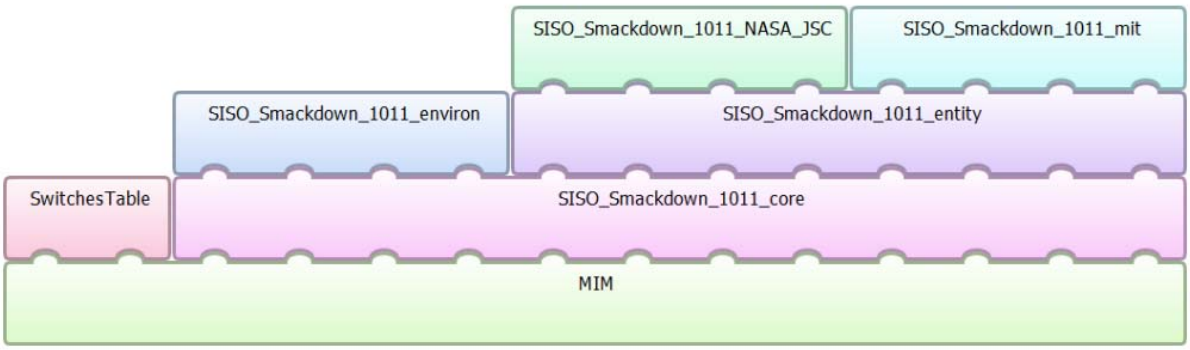

Figure 6.FOMs
This shows how FOMs in the SISO Smackdown depend on each other.

V. Lunar Rover Federate

A. Interactions

The Lunar Rover federate was a key component for the success of the simulation. It provides ground transportation of resources and interacts with three other federates. Because of all the interactions that the Lunar Rover relies on, the high degree of simulation interoperability was the main factor in getting the Lunar Rover working properly.

The Lunar Rover supports two different types of interactions. These include the state interactions and the resource transfer interactions. The state interaction is useful for determining where the Lunar Rover is or where it is going. For instance, once the Mobile ISRU Plant mines 500 kg of Helium-3, the Lunar Rover will depart for it to retrieve the resources. It is useful for other federates (especially the Mobile ISRU Plant) to know if the Lunar Rover is actually on its way, which is why the Lunar Rover sends out such interactions. Another interaction that the Lunar Rover receives is the arrival of the Lunar Shuttle at the landing site. Once the Lunar Rover receives this interaction, it will begin its travel to the landing site. All of this information is highly useful in determining if each federate is doing what it is supposed to do.

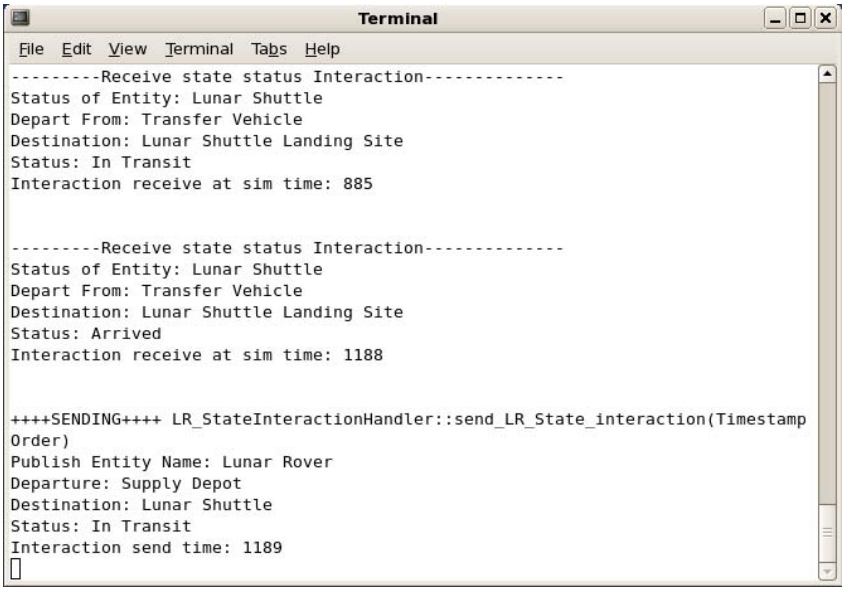

Figure 7. State Interaction
The state interaction can be seen in the figure above. Once the Lunar Shuttle sends the interaction out that it has arrived at the landing site, the Lunar Rover immediately sets course for it.

B. Attributes

The Lunar Rover also publishes and subscribes attributes. These attributes are updated every second to all federates that are subscribed to them. Attributes that the MIT team publishes and the Lunar Rover is

subscribed to include the amount of Helium-3 that the Mobile ISRU plant has on board and the status and position of the Mobile ISRU Plant. The Lunar Rover also subscribes to attributes that the Lunar Shuttle publishes. These attributes are the fuel amount that the Lunar Shuttle for its ascent and the position of the Lunar Shuttle. Because the position the Lunar Rover receives is in the lunar inertial reference frame, it must convert it to moon centric fixed so it is usable data. The Lunar Rover publishes a few of its own attributes. These include its position, mass, the reference frame it is operating in, and its name.

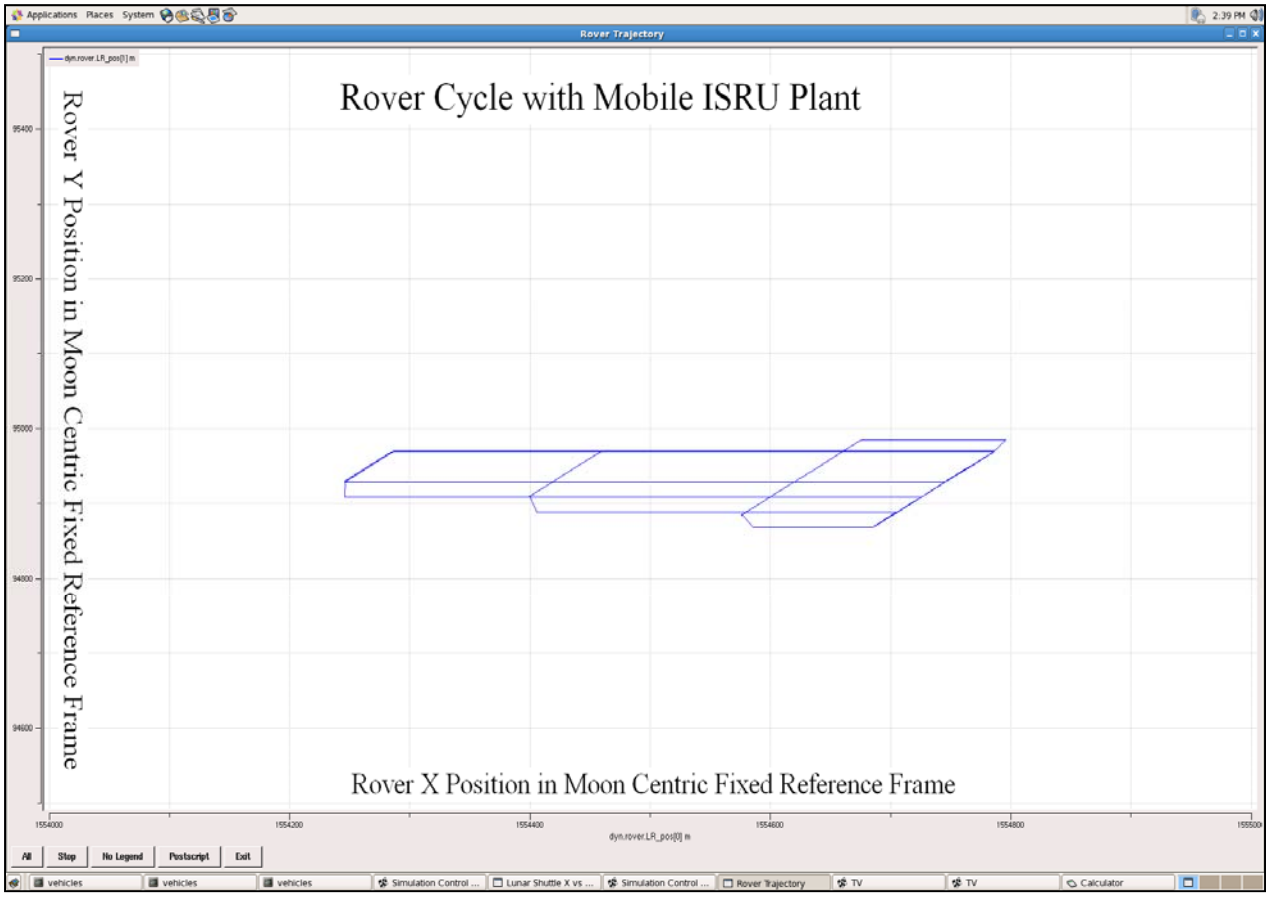

Figure 8. Lunar Rover Cycle with MIP

The Lunar Rover is subscribed to the attribute of the MIP that publishes how much Helium-3 it has. Once it is over 500 kg, the Lunar Rover will go to the MIP, receive the Helium-3 and go back to the Supply Depot. In the simulation this picture was taken, the MIP was finding a lot of Helium-3; therefore the Lunar Rover made numerous journeys back and forth.

C. Time Management

Time management is a critical component to building a successful simulation. There are two ways in which time is dealt with in the simulation. First of all, only two federates are time-regulating and those are the two NASA federates that provide the Earth-Moon transfer vehicle as well as the environment federate. Every other federate must not be time regulating. Time regulating federates can prevent other federates from advancing the logical time. Therefore, time regulating federates are the only federates that can advance time. Time constrained federates are constrained by the other federates. All federates in this simulation are time constrained.

Another important aspect of time management is determining at what interval time will be advancing. For the Smackdown, one second was chosen as the time interval. A second may seem like a long time in between updates. This means that interactions and attributes can only be sent and updated every second, and not in between. This is due to the fact that signals cannot travel faster than the speed of light. Because there are members of the Smackdown that live all over the world, testing was done through a VPN. If a signal has to travel from Texas to Massachusetts, France, Japan, Canada, and back, it will be very hard to run at anything less than one second. Updating every second allows the information to travel across the world as real-time updates. If the simulation ran at time intervals of 0.01 seconds, the system would get flooded with information, not to mention the fact that the teams in France may receive an update, but they are never up to speed with the people in Texas. If federates are all at different points in the simulation, then simulation interoperability begins to break down.

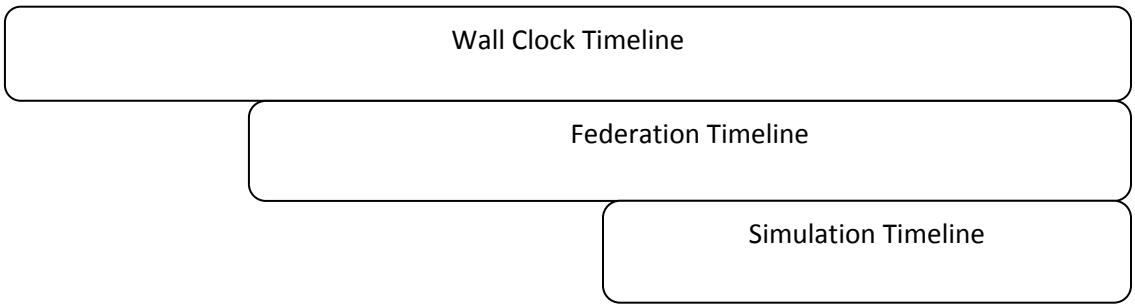

Figure 9. Timelines

The picture above shows how there are three different timelines that can be used as data for what time it is. The federation timeline is used to show when the simulation starts whereas, the simulation timeline shows when specifically the modeler's federate starts.

As shown in Figure 6, the Lunar Rover sends an interaction at simulation time 306 and the Lunar Shuttle receives this at simulation time 307. This keeps everything nice and orderly and does not flood the system with tons of information that it does not have time to process.

During a simulation there are three different timelines. There is the wall clock time which is different as you travel to different time zones. Also, there is the federation timeline which will give you information on how long the simulation has been running. The third timeline is the simulation timeline. This is defined as the time that a certain federate has been in the federation. All of these timelines are important with regards to time management. They allow for different time references that make analyzing data easier. For instance, if a federate joins into a simulation and sends an interaction as soon as it joins, it will show as one second into the simulation according to the simulation time. However, if the developer of the federate wants to know what time it is with respect to the beginning of the simulation, the developer can find that information out as well.

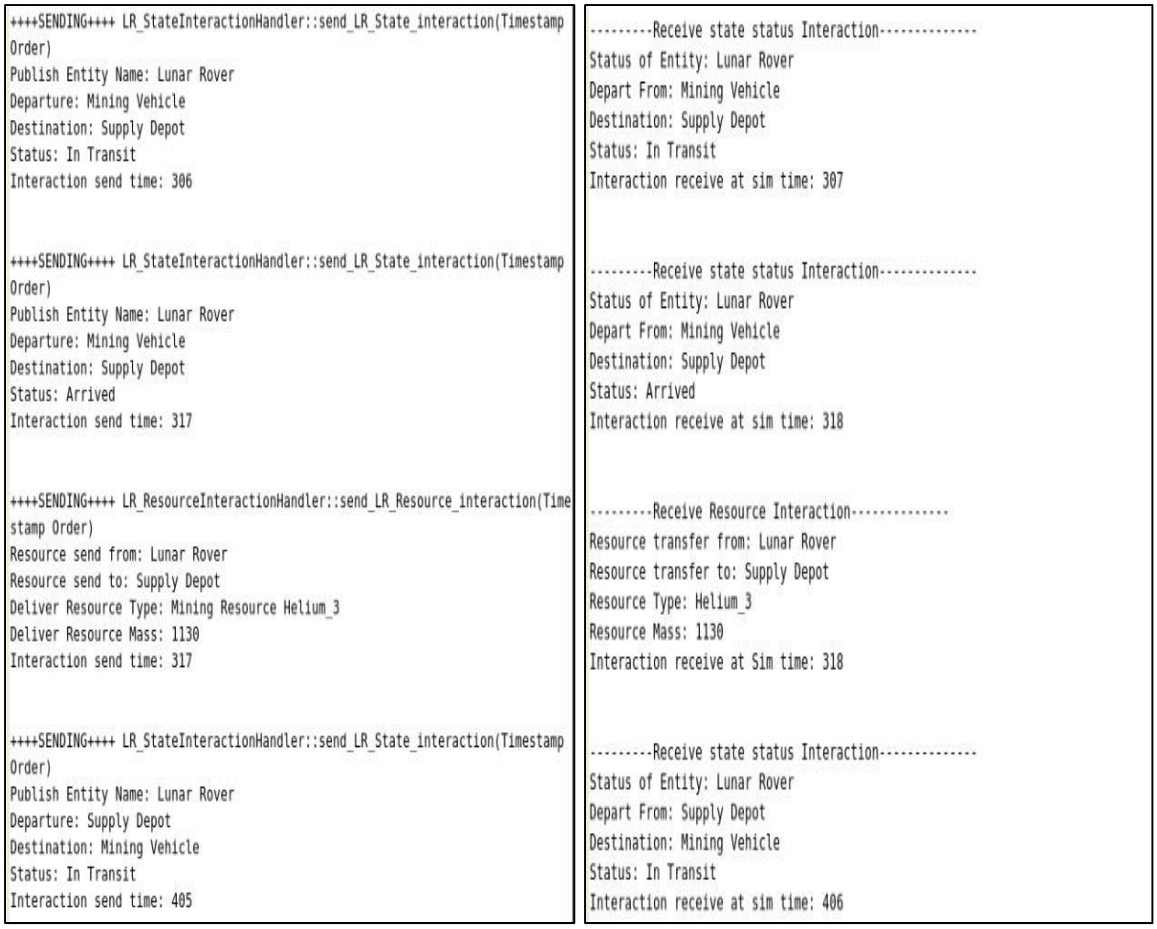

Figure 10. Resource Transfer Interaction

These two windows depict the interactions that the Lunar Rover (left) is sending out and the interactions that the Lunar Shuttle (right) is receiving. Notice that every time an interaction is send by the Lunar Rover, the Lunar Shuttle receives it one second later.

D. Specifications

The rover has a resting spot if it does not have a job to do. That location is:

- Latitude: 26°09' 14.4" N
- Longitude: 3° 23' 28.35" E
- Elevation: -1848 m

Purposely, this is also the location of the Supply Depot. This is a strategic spot due to it being the only federate that is not mobile. The Lunar Shuttle is only briefly at the landing site while the Mobile ISRU Plant and the Exploratory Hopper all incorporate tasks, requiring them to move around.

During a majority of the testing leading up to the SISO Smackdown, the Lunar Rover had velocities of 10 m/s in both the x and y directions (Assuming non-spherical surface due to very limited change in elevation and the proximity of every federate to another). Therefore, the magnitude of velocity was 14.14 m/s. This

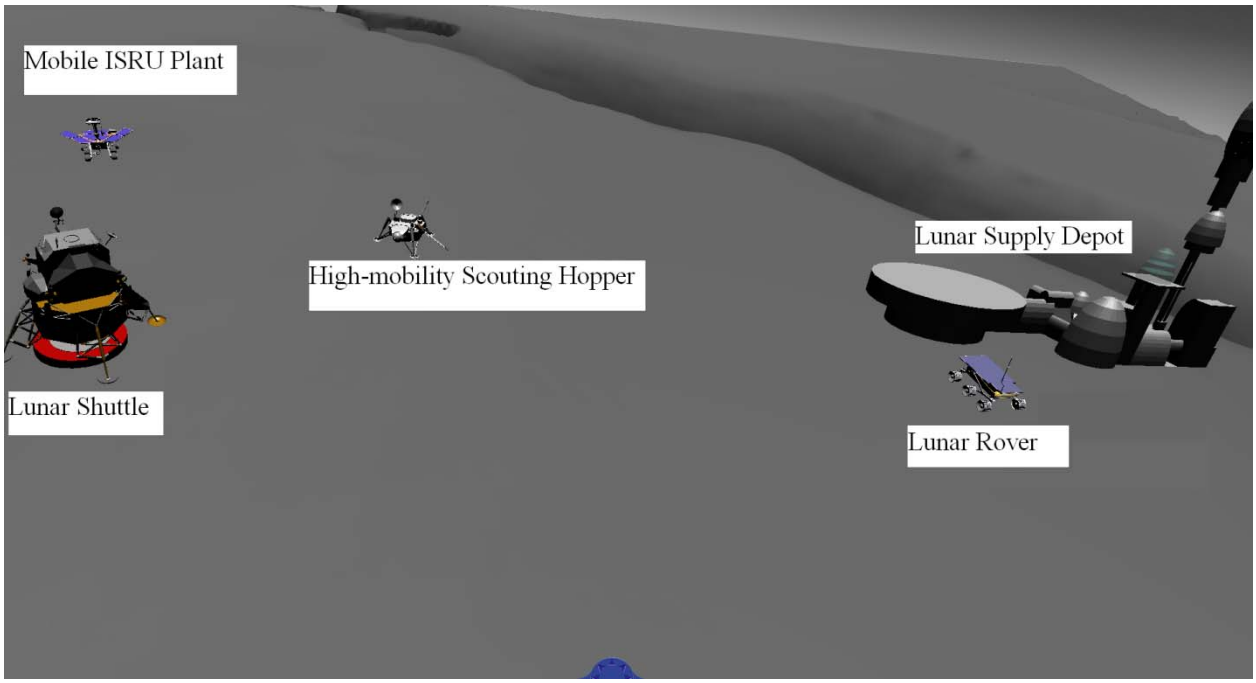

Figure 11. All Ground Federates
This shows five federates all operating on the lunar surface during the SISO Smackdown. (From Google Moon)

speed is already extremely fast for a Lunar Rover. More intriguing is the fact that the mission needed to be completed within an hour at the SISO Smackdown. Since the Lunar Shuttle approximately takes 22 minutes to land and about the same to ascend back into orbit, the Lunar Rover does not have much time to do all of the jobs it needs to do relating to the Lunar Shuttle. Before the Lunar Shuttle lands, the Lunar Rover is leisurely just traveling back and forth between the Mobile ISRU Plant and the Supply Depot. However, once the Lunar Shuttle is at the landing site, the Lunar Rover must almost immediately begin its cycles between the Lunar Shuttle and the Supply Depot. To compensate for lack of time, the Lunar Rover was given an increased speed of 20 m/s in both the x and y directions. This means the magnitude would reach up to 28.28 m/s which is unheard of for a Lunar Rover. The reason it must travel this fast is because the distance between the Supply Depot and the Lunar Shuttle is 2737.58 m. The primary reason for giving the Lunar Rover such a high speed was a means to create a distributed simulation that an audience could view without having to wait for hours. For a real scenario, the Lunar Rover could be drastically slowed down to improve the chances of mission success.

E. Rover Logic

The logic of the rover becomes quite complex due to the amount of interactions that it receives and sends. Also, the Lunar Rover is constantly checking certain attributes of other federates that it is subscribed to in order to see if there are any updates on what it should be doing. A flow char is presented to show some of the processes that the Lunar Rover must “think” through.

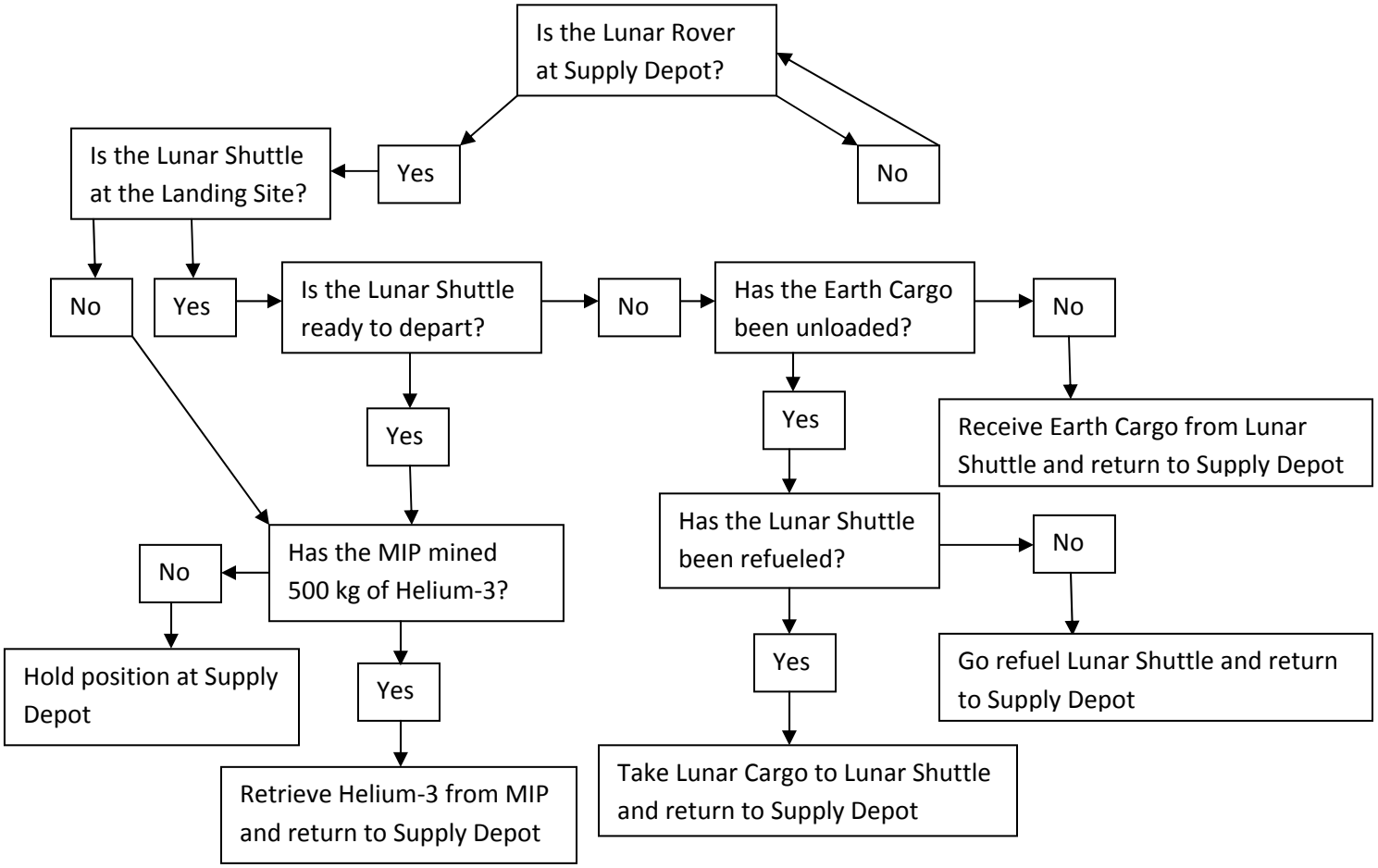

Figure 12.Rover Logic Flow Chart

This flow chart shows how the Lunar Rover processes the job that it needs to do. The cycle is updated every second during the simulation. Some interactions were left out such as the Lunar Rover receiving the fuel and cargo from the Supply Depot before it takes it to the Lunar Shuttle. This was done to reduce complexity.

Conclusion

The teams working on the Smackdown this year did an excellent job at getting their distributed space system up and running in such a timely fashion. The primary objective was to just get the simulation working, and each team excelled well beyond this goal. There are a lot of technical details with each federate that were not addressed due to the short time allotted to work on it. However, there will be another SISO Smackdown next year in which everyone is planning on improving each federate. The Lunar Rover will continue to be worked on, adding components that it lacked this year. New additions that the Lunar Rover should have include non-instantaneous delta V, non-instantaneous resource transfer, as well as six degrees of freedom. Another component that should be implemented into the Lunar Rover is a way in which to figure out the inclination of the surface of the moon at any given point. This will keep the Lunar Rover from driving off the surface of the moon and from driving down into it.

Modeling and simulation concepts are important tools and should be taught as a discipline in schools. They provide something that nothing else can – a chance to fix mistakes before an actual mission takes place. Simulations give characteristic data of real missions. Data such as this is invaluable. Reaching out to universities around the world is an important step over the course of the next year to help integrate more students into the world of modeling and simulation.

Acknowledgements

The author would like to extend his appreciation to his USRP mentor, Edwin Z. Crues, for his excellent guidance over the course of this project. He has a great deal of enthusiasm for getting university students involved in the modeling and simulation discipline, and it is people like him that inspire future generations of scientists and engineers. The author would also like to thank Dan Dexter for spending so much of time explaining the technical details of TrickHLA and other problems that the author encountered while developing his simulation. Two other people imperative to his successful internship are Mike Red and Zena Perryman. They helped the author to receive funding to attend the SISO Smackdown which was a crucial part for the completion of his project. Zuqun Li was very willing in helping the author to understand C and C++ since the author had never used it before. Lastly, the author would like to extend his regards to the education department at NASA for recruiting him as well as to everyone in his branch for making his time here at NASA one to remember.

References

S. Lippman, J. L. (2005). *C++ Primer.* Addison-Wesley.

Paul Tipler, R. L. (2003). *Modern Physics.* W. H. Freeman & Co Ltd.

Dexter, D. (2009). *TrickHLA v2.2.0.*