SISO Space Reference FOM - Tools and Testing

Björn Möller, Andreas Rydell
Pitch Technologies
Repslagaregatan 25
S-582 22 Linköping,
Sweden
bjorn.moller@pitch.se
andreas.rydell@pitch.se

Edwin Z. Crues, Dan Dexter
Simulation and Graphics
Branch (ER7)
Software, Robotics, and Simulation
Division (ER)
NASA Johnson Space Center
2101 NASA Road 1, Houston, TX
edwin.z.crues@nasa.gov
daniel.e.dexter@nasa.gov

Alberto Falcone, Alfredo Garro
University of Calabria
Department of Informatics, Modeling,
Electronics and Systems Engineering
(DIMES)
University of Calabria
Via P. Bucci 41C, 87036 Rende (CS),
Italy
alberto.falcone@dimes.unical.it
alfredo.garro@unical.it

Keywords:
Space, Interoperability, High Level Architecture, Testing, Federation Object Model, Reference FOM

ABSTRACT: *The Simulation Interoperability Standards Organization (SISO) Space Reference Federation Object Model (SpaceFOM) version 1.0 is nearing completion. Earlier papers have described the use of the High Level Architecture (HLA) in Space simulation as well as technical aspects of the SpaceFOM. This paper takes a look at different SpaceFOM tools and how they were used during the development and testing of the standard.*

The first organizations to develop SpaceFOM-compliant federates for SpaceFOM development and testing were NASA's Johnson Space Center (JSC), the University of Calabria (UNICAL), and Pitch Technologies.

JSC is one of NASA's lead centers for human space flight. Much of the core distributed simulation technology development, specifically associated with the SpaceFOM, is done by the NASA Exploration Systems Simulations (NExSyS) team. One of NASA's principal simulation development tools is the Trick Simulation Environment. NASA's NExSyS team has been modifying and using Trick and TrickHLA to help develop and test the SpaceFOM.

The System Modeling And Simulation Hub Laboratory (SMASH-Lab) at UNICAL has developed the Simulation Exploration Experience (SEE) HLA Starter kit, that has been used by most SEE teams involved in the distributed simulation of a Moon base. It is particularly useful for the development of federates that are compatible with the SpaceFOM. The HLA Starter Kit is a Java based tool that provides a well-structured framework to simplify the formulation, generation, and execution of SpaceFOM-compliant federates.

Pitch Technologies, a company specializing in distributed simulation, is utilizing a number of their existing HLA tools to support development and testing of the SpaceFOM. In addition to the existing tools, Pitch has developed a few SpaceFOM specific federates: Space Master for managing the initialization, execution and pacing of any SpaceFOM federation; EarthEnvironment, a simple Root Reference Publisher; and Space Monitor, a graphical tool for monitoring reference frames and physical entities.

Early testing of the SpaceFOM was carried out in the SEE university outreach program, initiated in SISO. Students were given a subset of the FOM, that was later extended. Sample federates were developed and frameworks were developed or adapted to the early FOM versions.

As drafts of the standard matured, testing was performed using federates from government, industry, and academia. By mixing federates developed by different teams the standard could be tested with respect to functional correctness, robustness and clarity.

These frameworks and federates have been useful when testing and verifying the design of the standard. In addition to this, they have since formed a starting point for developing SpaceFOM-compliant federations in several projects, for example for NASA, ESA as well as SEE.

1. Introduction

1.1 About the SpaceFOM

Spaceflight is difficult, dangerous and expensive; human spaceflight even more so. In order to mitigate some of the danger and expense, professionals in the space domain have relied, and continue to rely, on computer simulation. Simulation is used at every level including concept, design, analysis, construction, testing, training and ultimately flight. As space systems have grown more complex, new simulation technologies have been developed, adopted and applied. Distributed simulation is one of those technologies. It provides a base technology for segmenting these complex space systems into smaller, and usually simpler, component systems or subsystems.

Over the years, a number of distributed simulations have been built within the space domain. While many use the High Level Architecture (HLA) to provide the infrastructure for interoperability, HLA without a Federation Object Model (FOM) is insufficient by itself to ensure interoperability. As a result, the Simulation Interoperability Standards Organization (SISO) is developing a Space Reference FOM (SpaceFOM). The purpose of the SpaceFOM is to facilitate a priori interoperability and reuse for space simulations. It makes it possible to compose federations consisting of simulations from one or several different organizations. It will become an open, international standard, freely available from SISO.

The SpaceFOM defines: (i) a flexible positioning system for bodies in space through Coordinate Reference Frames; (ii) a naming conventions for well-established Reference Frames; (iii) descriptions of common time scales; (iv) federation agreements for execution control and common types of time management, with focus on time stepped simulation; and (v) support for physical entities (e.g. space vehicles, rovers and astronauts).

1.2 Previous work

Several earlier papers on the SpaceFOM have been published and are recommended as background reading. The 2016 paper "A First Look at the Upcoming SISO Space Reference FOM" [30] provides an introduction to distributed simulation in the space domain and describes a number of early federations. It also provides an overview of the structure and functionality of the SpaceFOM. The technical description is further expanded in a number of additional papers [31][17][18][19]. When building federations, the SpaceFOM provides extensive flexibility, for example in the selection of reference frames and time steps. These parameters shall be documented in a Federation Execution Specific Federation Agreement, which is further described in a presentation [32]. The SpaceFOM itself is of course the ultimate document to read, and drafts have been publicly available in the SISO Digital Library since 2017.

1.3 Development process, and current status

The idea of developing of a standard was informally discussed within SISO during the period 2011-2012. In 2013 a meeting was held at a SISO Workshop and a group decided to propose the development of a standardized SpaceFOM. The standards development process within SISO is known as the SISO Balloted Product Development and Support Process (BPDSP) [29]. A Product Nomination, the first step in BPDSP, was developed in 2014 and submitted to the SISO Standards Activity Committee (SAC) and approved in May 2015. The next step was drafting of the actual product, which is where the main technical effort lies. During the second half of the development, several teams spent substantial time working on prototypical implementations. After this balloting was carried out, where the standard passed balloting and 37 comments were given. These have since been resolved and as of December 2019, the standard has been submitted to the SAC and SISO Executive Committee for approval.

1.4 How to test a standard?

It is both critical and a challenge to ensure that a standard, like SpaceFOM, actually works. Some standardization organizations require that a complete reference implementation is implemented to prove the solution, although SISO doesn't. It is up to each development group to decide the appropriate level of testing to be performed.

Testing can be done as a final phase of the standards development. When a standard is incrementally developed, it is also highly useful also to perform testing throughout the development, or at least to start testing once a baseline of the standard has been developed. This may even make the standards development a somewhat exploratory activity, so that different design candidates can be evaluated.

Standards need to be tested for several aspects, such as functional correctness, robustness and clarity. The functional aspect of testing is reasonably well understood in the software testing community. One key difference, compared to standards development, is that they have a bigger focus on testing implementations rather than specifications. For a standard, it may be just as important to test that the written agreements, such as distributed algorithms, are described in a way that different implementers will interpret the standard in the same way. This means that the entire process from reading the standard, to implementation, development of sample scenarios, and execution needs to be performed, preferably by different teams that may ultimately compare, and cross-test different implementations.

The SpaceFOM development group chose a more ambitious level of testing than many other SISO groups. Three teams, one from government (National Aeronautics and Space Administration - NASA), one from academia (University of Calabria) and one from industry (Pitch Technologies) participated. Existing tools and frameworks were adapted to the SpaceFOM. Some new tools and simulations were also developed. Several of these tools were also used to test the SpaceFOM in university programs. This paper describes these tools, the testing approaches and some observations and conclusions.

2. NASA JSC tools

This section of the paper provides a brief discussion of NASA's motivations for participating in the development of the SpaceFOM, a few of the principal tools used in the development, and select experiences using the standard. This also includes a brief history of distributed simulation development in support of some of NASA's human space exploration.

2.1 About NASA JSC

NASA's Johnson Space Center (JSC) is one of NASA's lead centers for human space flight. JSC is the location of NASA's human space flight Mission Control Center (MCC) and the home to NASA's astronauts. In addition to flight operations, JSC also supports important collaborative engineering design and development activities involving international partners. This includes development of distributed simulations for joint training and operations. Much of the core distributed simulation technology development, specifically associated with the SpaceFOM, is done by the NASA Exploration Systems Simulations (NExSyS) team. While the NExSyS team itself formed in 2012, members of the team have simulation experience that reaches back into simulation support activities for both NASA's Space Shuttle and International Space Station (ISS) programs. The NExSyS team's home is in the Simulation and Graphics Branch (ER7) at JSC and they work with groups across the agency and internationally.
Experienced members of the NExSyS team are contributing to SpaceFOM development and are members of the Product Development Group (PDG). They are working with SISO and the PDG to incorporate some of the lessons learned in previous NASA distributed simulation projects.

2.2 Tools and frameworks

The NExSyS team employed some of NASA's principal modeling and simulation tools to explore, develop and test the SpaceFOM. One of NASA's principal simulation development tools is the Trick Simulation Environment (see [27], [20]). Trick provides a common simulation infrastructure used by many of NASA's space systems simulations. Trick itself is not a simulation, it aggregates constituent application specific models and provides a simulation framework for application specific simulations. In order to provide general support for distributed simulation, a simulation interoperability framework has been developed for Trick-based simulations. The framework, called TrickHLA, is a Trick compatible implementation of HLA (see [3], [28]). This, in conjunction with other space systems models, provides the simulation framework for implementing NASA federates for HLA based distributed simulations like the proximity operations and capture flight controller training simulation for the ISS and the H-IIA Transfer Vehicle (HTV). NASA's NExSyS team has been modifying and using Trick and TrickHLA to help develop and test the SpaceFOM.

2.3 Adaptation to the SpaceFOM

As is probably evident from the previous discussion, NASA and the NExSyS team have been developing HLA-compliant distributed simulations using Trick and TrickHLA for many years. However, these distributed simulations have not been SpaceFOM-compliant. Compliance with the SpaceFOM standard places additional requirements to those imposed by HLA, these include role responsibilities, initialization sequencing, time standards, reference frame publication, and execution control. While the Trick simulation environment did not require adaptation to support the SpaceFOM, TrickHLA did require adaptation. Although, TrickHLA already supported many of the initialization and execution control concepts, a significant effort was required to design, implement, and test the SpaceFOM-compliant additions and adaptations. Here is a list of the key additions and adaptations:

- Support for the ExecutionControl object class and singleton ExCO object instance infrastructure;
- Support for the ModeTransitionRequest interaction infrastructure;
- Implementing rigorously defined time line classes for the principal operating time lines;
- Implementing the Master, Pacing, and Root Reference Frame Publisher roles;
- Implementing the SpaceFOM-compliant Early Joiner and Late Joiner identification;
- Implementing a flexible framework to support federation execution specific multi-phase initialization;
- Implementing the SpaceFOM-compliant execution control and executive infrastructure;

These additions and adaptations were made over the course of SpaceFOM development and tested with both in-house NASA simulations and SEE activities.

2.4 Experiences

Historically, NASA has engaged with both domestic commercial and international partners when planning and developing human rated space systems and missions. The ISS is a successful example of this. While human space exploration is a technically challenging endeavor by itself, these collaborations bring with them challenges beyond just the technical. Some of the non-technical challenges are political sensitivities, restrictions on proprietary data, and the protection of intellectual property. Collaborative distributed simulation is one technology that has successfully been used to address many of these technical and non-technical challenges.

One of NASA's early successes with distributed simulation was with the Japan Aerospace Exploration Agency (JAXA) and the HTV logistic supply vehicle for the ISS (see [12], [13]). All NASA missions require significant training for flight controllers. The HTV logistic supply missions to the ISS are no exception. In this case, NASA worked with JAXA to develop a training simulation for the HTV and ISS flight controllers that involved a distributed simulation with components executing at JAXA's Tsukuba Space Center in Japan and at NASA's Johnson Space Center in Houston, Texas. Much of the research, development, and deployment for this project formed the knowledge base that would eventually be incorporated into the SpaceFOM.

The ISS/HTV flight controller training simulation was a successful precursor to other NASA programs that benefited from HLA based distributed simulation (see [2], [15]). As NASA programs and their associated distributed simulations advanced, the developers observed a number of repeating challenges with developing, deploying and operating these simulations. In general, the developers observed that HLA provides a necessary simulation interoperability framework but is not sufficient by itself. There are some additional areas that need more formal agreement: simulation execution roles and responsibilities; fundamental data types and representations; initialization sequence and data dependencies; and execution control. Some of the specifics are described in more detail in these references: ([21], [22], [14]). In general, this indicated the need for a more formal standard for interoperable space systems simulation. Specifically, this indicated a need for the SpaceFOM.

Recently, NASA announced the Artemis program to return humans to the Moon by 2024. The NExSyS team is currently involved in developing, testing, and deploying distributed simulation elements of many of the principal components of the evolving Artemis architecture. These elements include the Orion, Gateway, and Human Landing Systems (HLS). The Orion is a human rated spacecraft that will take four astronauts from Earth orbit, to lunar orbit, rendezvous with Gateway and lunar systems, and return the astronauts to Earth. The Gateway can be thought of as a space station in orbit about the moon to provide a staging point to explore the Moon and a departure point for missions to Mars. HLS is the multi-stage space system that will take astronauts to and from the surface of the Moon.

NASA and the NExSyS team recently developed and demonstrated a multi-element simulation of key mission phases involving Orion, Gateway, and HLS. These simulations were Trick and TrickHLA-based SpaceFOM-compliant federates running in multiple SpaceFOM-compliant federation executions. The principal federates were the Gateway federate, the Orion federate, the Orion Environmental Control and Life Support System (ECLSS) federate, and the HLS federate. These were used to demonstrate a number of different mission phases: rendezvous and docking of the HLS with the Gateway; rendezvous and docking of the Orion with the Gateway/HLS stack; operation of the combined Gateway/Orion/HLS stack; HLS landing on the surface of the Moon; and HLS departure from the Moon.

By using HLA and the SpaceFOM as the standard for constructing these federates, the simulation teams could concentrate on the details of modeling the space systems. As these were all Trick-based simulations, the NExSyS team could leverage off the work already done in transforming Trick-based space systems simulation into HLA federates using TrickHLA. Making these federate SpaceFOM-compliant required little additional work since TrickHLA provided most of the SpaceFOM infrastructure. Any additional work was mainly related to supporting the new PhysicalEntity object class types for vehicle state exchange.

As the Artemis program goes forward, these SpaceFOM-compliant tools will provide an important technology to support NASA's collaborations with domestic and international partners as we return to the Moon, explore Mars and move out into our solar system.

3. SMASH-Lab tools

This section presents the main and most interesting and relevant experiences, libraries and frameworks derived from the exploitation of the SpaceFOM standard by the SMASH-Lab of the University of Calabria. Firstly, a brief discussion on the SMASH-Lab mission and motivations for participating in the development of the SpaceFOM and experience in using the standard are presented. After that, the libraries and frameworks developed in the context of the SpaceFOM to support the design and development of compliant HLA federates are described. In the end, the experiences in using the libraries and frameworks in the university program named *Smackdown*, later renamed SEE, where university teams simulated space scenarios using HLA are presented [25]. The SEE is an international project promoted by SISO, in collaboration with NASA and other research and industrial partners. The main objective of the project is to provide to undergraduate and postgraduate students a practical experience in M&S, especially, in developing distributed simulations in the space domain based on the HLA standard.

3.1 About SMASH-Lab

The SMASH-Lab carries out research activities on analytical models to support the development of simulators, Modeling and Simulation (M&S) of complex systems to support the organization and operation of such systems and performance analysis. Other research activities regard paradigms, architectures, models and simulation methods for distributed computing environments. Members of the SMASH-Lab are contributing to SpaceFOM development and are members of the PDG. They have been working with SISO and the PDG to incorporate some of the lessons learned in research activities on distributed simulation systems.

3.2 Libraries and frameworks for the SpaceFOM

In the context of the SpaceFOM standardization activities, the SMASH-Lab developed a set of open source frameworks and libraries to support the development of HLA federates, which is usually a quite complex task. Specifically, the SMASH-Lab developed the *HLA Development Kit* and *RxHLA* frameworks and the *Java Space Dynamics Library*.

3.2.1 The HLA Development Kit framework

The *HLA Development Kit (DKF)* is a general-purpose, domain-independent framework, released under LGPL license, which eases the development of HLA federates [9]. The first version (version 1.0.0) was released in April 2014, whereas the latest (version 1.4.2) was released in March 2018. The framework allows developers to focus on the behavior of their simulation models rather than dealing with common HLA-based functionalities (see [9], [10] for details). The framework does not represent another implementation of the HLA standard but it represents a framework placed on available HLA/RTI implementations providing an additional layer of abstraction suitable to hide the complexity of the HLA standard and DKF low-level functionalities. The DKF is implemented in the Java language and covers the IEEE 1516-2010 specifications and can interoperate with different HLA/RTI implementations, such as Pitch and VT MÄK [20].

To support the university teams involved in the SEE project, the DKF has been specialized in the *SEE HLA Development Kit (SEE-DKF)* [10]. It provides space domain-specific services, which are used by the core components of the DKF to handle the main aspects related to a SEE Federation, such as transformations among *Reference Frames*, the publishing and subscribing of *PhysicalEntities* and the management of space-time [17], [18], [19]. Specifically, the SEE-DKF is organized in two service sections (see [10] for details): (i) *Frame,* which offers services for representing the position, geometry, and characteristics of space bodies, such as planets and stars; and (ii) *SpaceTime,* which provides functionalities to handle space-time, epochs (e.g., j2000, Julian Epoch (JE) and Modified Julian epoch (MJE)) and dates that are commonly defined by specifying a point in a given time scale. The current version of the DKF and SEE-DKF are SpaceFOM fully-compliant and provides a set of functionalities that make it easier for developers to use the HLA and SpaceFOM standards [9], [10].

3.2.2 The RxHLA framework

In today's HLA space simulations especially the ones compliant with the SpaceFOM, in which federates are highly concurrent, distributed, and the interactions among them are asynchronous, great benefits derive to reactively manage the federates' communication flow instead of explicitly manage it through Callbacks. *RxHLA* is a framework that provides a specific implementation of reactive programming concepts for HLA [24]. The framework allows to build reactive, concurrent, and distributed time/event-driven Federates in a reactive fashion through non-blocking *Observable* streams [7].

3.2.3 The Java Space Dynamics Library

Concerning libraries, the SMASH-Lab developed a space flight dynamics library, named *Java Space Dynamics Library (JSDL)*, which offers high fidelity models and algorithms to manage space bodies according to the SpaceFOM specifications [8]. JSDL is a low-level space dynamics library that facilitates the design and development of space systems, such as space vehicles and satellites. The JSDL provides to developers: (i) the technical documentation, which describes the library with its philosophy and mission; (ii) the user guide, which guides developers in the use of the library; (iii) the source code, which developers can investigate and customize the architecture and functionalities to fit their own needs; and (iv) set of reference examples, which show how to handle space systems with the offered data structures and algorithms.

3.3 Experiences in using the SMASH-Lab frameworks and libraries

Since 2014, the members of the SMASH-Lab have been involved in the SEE project, initially as a university team and later, since 2016, as members of the executive committee. In the last 2019 edition of SEE, that was held from 6th to 7th May 2019 in Cape Canaveral, Florida, USA, 13 universities participated in simulating a space mission compliant with the HLA and SpaceFOM version 0.3, which has been submitted for approval to SISO [26]. The space mission involved the simulation of a settlement on the Moon and on Mars with a dangerous scenario involving an asteroid on a collision course with the Moon. To avoid the impact of the asteroid, a missile was launched from the moon surface to intercept and destroy it. In this edition, the SMASH-Lab members provided, with the NASA team, support to the SEE teams in developing HLA federates compliant with the SpaceFOM also through the use of the DKF and SEE-DKF. Specifically, the frameworks have been successfully experimented by the universities of Bordeaux (France), Brunel (London, UK), the Faculdade de Engenharia de Sorocaba FACENS (Brazil), Genoa (Italy), Munich (Germany), Jaipur (India) and Bulgaria (Bulgaria).

Other experimentations of the DKF and SEE-DKF were conducted in the Complex Engineered Systems (CES) and Model-Driven Systems Engineering (MBSE) domains. Concerning the first one, the frameworks were used to address, in an integrated way, the issues of *reusability*, *interoperability* and *distribution* of CES simulation models. A solution called *Adapter-based Hybrid Federate* that is based on the integration of the *Functional Mock-up Interface (FMI)* and *HLA* standard has been presented. To demonstrate the feasibility and validity of the defined solution, a case study concerning a lunar lander that descends on the lunar surface has been provided (see [6] for details). The dynamics of the lunar lander and the involved planets, such as Sun, Earth and Moon involved in the simulation have been managed through the JSDL models and algorithms. Regarding MBSE, the DKF was integrated to create the *MOdel-driveN Architecture for Distributed Simulation (MONADS)* model-driven method. It takes as input system models specified in SysML, the reference modeling language in the systems engineering field, and generates as output the final code, based on the DKF, of the corresponding HLA distributed simulation through a chain of *model-to-model* and *model-to-text* transformations [1].
Recently, MONADS has been extended to integrate the reactive features offered by the RxHLA framework [5] in order to manage SpaceFOM compliant federates. More in details, the *Develop and Integrate Simulation Environment* phase of MONADS has been modified by including three development strategies: *No-RxHLA*, *Automatic-RxHLA-configuration* and *Custom-RxHLA-configuration*.

4. Pitch Technologies tools

4.1 About Pitch

Pitch Technologies provides products and services for distributed simulation, with offices in Sweden, United Kingdom (UK), France and the United States (US). The company has been involved in international standardization since 1995, when the US Department of Defense (DoD) initiated the HLA standard [20]. Major involvement, before the SpaceFOM, were HLA, the Federation Development and Execution Process (FEDEP, later DSEEP) and the Real-Time Platform Reference FOM (RPR FOM). Initially Pitch worked mostly in the defense and security market, but already around the year 2000, the Space community started to show interest in Pitch products. Pitch's infrastructure products were for example used in the NASA-JAXA federation for docking the HTV with the ISS, as well as the ATV-ISS ground controller training system, distributed between Korolov, Bremen and Toulouse, developed by the European Space Agency (ESA) and Astrium.

4.2 Generic simulation tools

Pitch provides a number of generic tools for the development and execution of distributed simulations. These are of course also useful for SpaceFOM applications. The development process of a SpaceFOM federation, where Pitch's tools are used, will typically follow the following steps:

1. A federation agreement design document is developed using a word processor. Good guides and checklists for this can be found in the SISO Distributed Simulation Engineering and Execution Process (DSEEP) [33], the SISO Standard for Federation Engineering Agreements Template (FEAT) [34] and the SISO SpaceFOM, in particular the Federation Execution Specific Federation Agreement (FESFA) appendix;
2. The SpaceFOM is extended with additional Object and Interaction classes that meet the needs of the information exchange particular to a federation. Here Pitch Visual OMT is often used for FOM development;
3. To be able to easily integrate new and existing simulations as federates in the federation, C++ or Java middleware that interfaces the HLA federation can be generated using Pitch Developer Studio;
4. Federates connect to an HLA-compliant RTI, like the Pitch pRTI, and execute together;
5. Data is collected using Pitch Recorder for monitoring, playback, debugging or analysis.

4.2.1 About Pitch Visual OMT

This tool provides a graphical environment for developing HLA object models, in particular using FOM modules. Consider

a federation with separate FOM modules for celestial bodies, spacecrafts, sensors, rovers, astronauts and for mission data. A typical extended SpaceFOM may look like this in the overview view of Pitch Visual OMT (see Figure 1).
Object models are developed in a graphical user interface. Correctness, completeness and best practice is monitored and reported, as well as suggestions how to improve the model. Different versions of models can be compared. Documents can be exported in PDF format or in MS Word for inclusion in other documents.

4.2.2 About Pitch Developer Studio
Once a FOM is available, Pitch Developer Studio can support federate development. It generates C++, C# or Java code that handles all details in the information exchange according to the FOM (encoding, decoding) and the use of HLA services. The resulting code provides Set and Get operations for all data. In general, it takes hours instead of weeks to adapt a simulation to a particular FOM. A cookbook for developing SpaceFOM federates is also provided.

4.2.3 About Pitch pRTI
This is an HLA Run-time Infrastructure that implements the services in the HLA interface specification. It is used when connecting the federates to the federation. It has a number of features to help monitoring and trouble-shooting the execution. Some things are of particular interest to Space federation developers: on the Central RTI Component (CRC) GUI it is possible to inspect which federates have joined the federation, their publications and subscriptions, which synchronization points have been registered and achieved, as well as how different federates advance time, both in a graphical and numerical overview. In the Local RTI Component (LRC) GUI, called the Control Center, it is possible to inspect local properties, for example connection status, incoming and outgoing event rate, and to control tracing of how HLA services are used. There is also companion product for Pitch pRTI, called Pitch Booster that makes it possible to run HLA federations across wide area network links, in particular where there may be firewalls and different network address ranges.

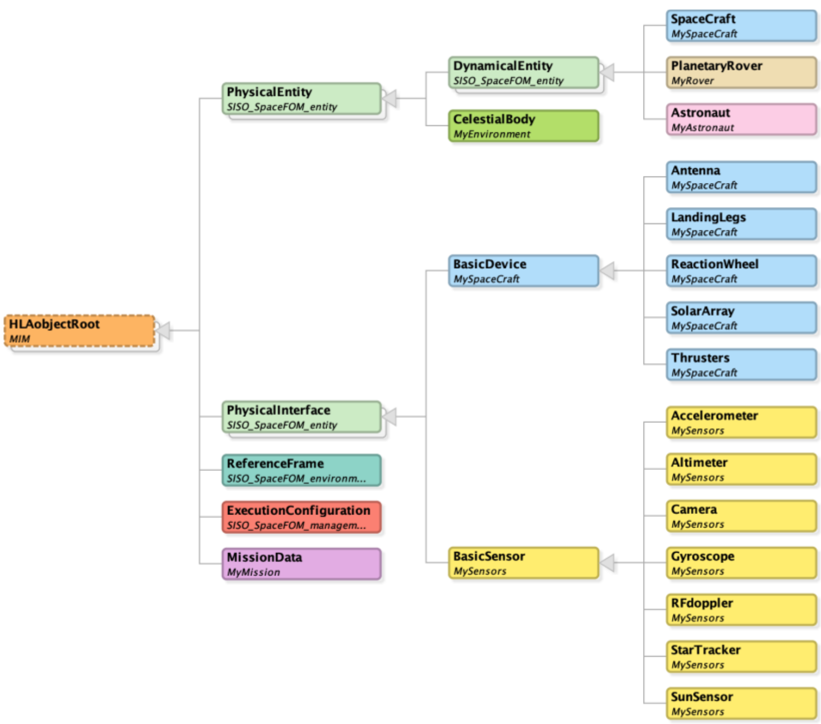

Figure 1: Sample Extended SpaceFOM in Visual OMT

4.2.4 About Pitch Recorder
This tool can record and play back object, updates and interactions for any object model, such as an extended SpaceFOM. Data can be recorded on several channels, for example space craft data on one channel and radio communications on another. Since Pitch Recorder reads the FOM, it can also decode and pretty-print the data. Data can also be exported for example for analysis in other systems. If other types of data exchange than HLA is used, such as proprietary protocols or streamed video, this can be recorded in synch with HLA data.

4.3 Tools and frameworks for the SpaceFOM

Pitch has also developed a few software applications that are specific to the SpaceFOM. These have been used during the testing of the SpaceFOM, in the SEE activities and in commercial projects. Three of them are of particular interest since they can be used in almost any SpaceFOM federation:

1. Space Master. This is a generic federate that can manage the startup, execution and pacing of a SpaceFOM federation;
2. Space Monitor. This is a generic monitoring tool for SpaceFOM federations;
3. Earth Environment. This is a very simple federate that creates an Earth Fixed reference frame as a Root Reference Frame.

4.3.1 About Space Master

This tool implements the design pattern for a Master federate according to the SpaceFOM. It checks that all require federates are available. It can also manage multi-phase initialization in a federation, where the user can specify the phases in a configuration tool. It also manages the startup of time management as well as pacing. Finally, it enables the user to interactively control how the federation is paused, resumed and shut down. It is based on a command line user interface and can thus be scripted.

4.3.2 About Space Monitor

This tool provides a graphic display of all available reference frames and physical entities in a federation, together with their spatial data (Figure 2). This makes it easy to verify if federates have registered the expected objects and they are updating them continuously.

4.4 Further Experiences

Pitch is currently involved in customer projects that use the SpaceFOM to build a distributed simulation for example for ESA. Papers with experiences from this project are planned to be published in one to two years' time.

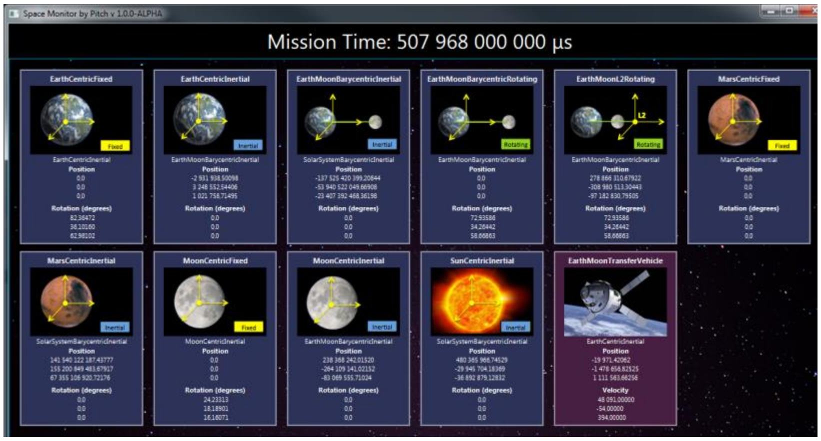

Figure 2: Space Monitor for a SEE federation

5. Testing of the SpaceFOM

The three teams from the development group that performed testing were NASA, University of Calabria and Pitch Technologies. In addition to this, a number of student teams in the SEE activity performed some testing. The results and experiences from testing have led to several discussions and in some cases updates to the SpaceFOM. In a few cases the texts have been updated for clarity.

5.1 Testing in the development group

Testing of the standard and its implementation was carried out throughout development, which helped both refining designs and verifying correct operations. All three teams developed at least a basic federation on their own. Generic tool, like the user interface of the RTI and data loggers were used for basic testing. Examples of such tests are the registration of synchronization points, the use of time management services and exchanging updates of attributes.

For the testing between teams, the main focus was distributed algorithms, in particular execution control and time management. The SpaceFOM is probably the most advanced standard when it comes to time management patterns, where

HLA time management services are combined with pause/resume execution control and potentially also hard real-time patterns. These patterns were implemented by all three participants. As an example, the initialization and execution control were executed in a federation using University of Calabria's HLA Development Kit. Later on, the Pitch's Space Master was introduced to handle these services. Another example is that NASA tested the Pitch EarthEnvironment federate together with the NASA TrickHLA.

5.2 Testing in SEE

SEE, originally started in 2011 as the "SISO Smackdown", is an annual college-level modeling and simulation challenge. SISO and the Society for Modeling & Simulation International (SCS) are sponsors. The purpose is to bring practical M&S experiences to students. It attracts student teams from all over the world. Teams are invited to participate in a space scenario where they can contribute their own simulations.

The SEE team has used FOMs that are predecessors to the SpaceFOM from the very beginning. As the standard developed, early versions were fed into SEE. The scope of the FOM that the student teams were required to implement was gradually increased over time.

5.3 Observations about testing

In addition to ensuring a correct standard, some observations about testing has been made. One observation is that testing of basic HLA operations, like the registration of a synchronization point or the update of an attribute using COTS tools is fast and easy. It gives a high degree of confidence since it can be assumed that the COTS tool is thoroughly tested and correct. Another observation is that testing complex interplay between federates, for example a synchronized freeze between several time managed, newly developed federates, is more difficult and time consuming. It may be challenged by minor bugs in federates implemented by different teams. It takes longer time to reach a high degree of confidence in such tests. Yet another observation is that testing has led to a better understanding of how to handle degraded situations, like crashing or misbehaving federates.

5.4 Other testing

Additional use and testing of the SpaceFOM is underway in NASA, ESA, SEE and other projects. Experiences will be further reported in upcoming papers.

6. Conclusions

After extensive testing, the SpaceFOM is now ready to be released. Based on practical results from testing and development of early federations, we are confident that it meets high quality requirements.

Testing of a standard is somewhat different from regular software testing, since the test relates to a specification, not an implementation. This means that both the technical aspects and the clarity of the standard had to be verified. The SpaceFOM is based on HLA, another open standard, so existing COTS tools could be used for fast and efficient testing of basic behaviors. For more advanced functionality, three teams have developed federates and frameworks for testing. Tests have been carried out and some adjustments have been made.

The tools, frameworks and federates have since formed a starting point for developing SpaceFOM-compliant federations in several projects, for example for NASA, ESA as well as SEE.

7. References

[1] P. Bocciarelli, A. D'Ambrogio, A. Falcone, A. Garro, and A. Giglio. A model-driven approach to enable the simulation of complex systems on distributed architectures. SIMULATION: Transactions of the Society for Modeling and Simulation International, 95(12), 2019. doi: 10.1177/0037549719829828.

[2] Crues, Edwin Z., Chung, Victoria I., Blum, Michael G., Bowman, James D. An Overview of the Distributed Space Exploration Simulation (DSES) Project. In Proc. of the Spring 2007 Simulation Interoperability Workshop and Conference, Norfolk, Virginia, March 2007.

[3] Crues, Edwin Z., Lin, Alexander, Hasan, David A. Integrating HLA Into the Trick Simulation Development Toolkit. In Proc. of the AIAA Modeling and Simulation Technology Conference, Austin, Texas, August 2003.

[4] Chung, Victoria I., Crues, Edwin Z., Blum, Michael G., Alofs, Catherine A., Busto, Juan M. An Orion/Ares I Launch and Ascent Simulation - One Segment of the Distributed Space Exploration Simulation (DSES). AIAA-2007-6625. In Proc. of the AIAA Modeling and Simulation Technologies Conference and Exhibit, Hilton Head, South Carolina, 20-23 August, 2007.

[5] A. D'Ambrogio, A. Falcone, A. Garro, and A. Giglio. Enabling Reactive Streams in HLA-based Simulations through a Model-Driven Solution. In Proc. of the 23rd IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DS-RT 2019), Cosenza, Italy, October 7-9, 2019. IEEE Inc., 2019.
[6] A. Falcone and A. Garro. Distributed Co-Simulation of Complex Engineered Systems by Combining the High Level Architecture and Functional Mock-up Interface. Simulation Modelling Practice and Theory, 97(August):101967, 2019. doi: 10.1016/j. simpat.2019.101967.
[7] A. Falcone and A. Garro. Reactive HLA-based Distributed Simulation Systems with RxHLA. In Proc. of the 22nd IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DS-RT 2018), Madrid, Spain, October 15-17, 2018. IEEE Inc., 2018, pp. 1-8. doi: 10.1109/DISTRA.2018. 8600936
[8] A. Falcone and A. Garro. A Java library for easing the distributed simulation of space systems. In Proc. of the 16th International Conference on Modeling and Applied Simulation (MAS 2017), Held at the International Multidisciplinary Modeling and Simulation Multiconference (I3M 2017), Barcelona, Spain, September 18-20, 2017. pp. 6-13.
[9] A. Falcone, A. Garro, S.J.E. Taylor, A. Anagnostou, N.R. Chaudhry, O. Salah, Experiences in simplifying distributed simulation: the HLA development kit framework, Journal of Simulation. 11 (3) (2017) 208-227. doi: 10.1057/s41273-016-0039-4.
[10] A. Falcone, A. Garro, A. Anagnostou, and S. J. E. Taylor. An introduction to developing federations with the High Level Architecture (HLA). In Proc. of the 2017 Winter Simulation Conference (WSC 2017), Las Vegas, NV, USA, December 3-6, 2017. Institute of Electrical and Electronics Engineers Inc., 2017, pp. 617–631. doi: 10.1109/WSC.2017.8247820.
[11] IEEE 1516-2010. High Level Architecture (HLA), [online], available at www.ieee.org, accessed 20 Nov 2019.
[12] Lauderdale, Grace D., Crues, Edwin Z., Snyder, Danielle V., Hasan, David A. A Feasibility Study For ISS and HTV Distributed Simulation. In Proc. of the AIAA Modeling and Simulation Technology Conference, Austin, Texas, August 2003.
[13] Lauderdale, Grace D., Crues, Edwin Z., Snyder, Danielle V., Hasan, David A. Further Studies On The Feasibility Of A Distributed ISS and HTV Simulation. In Proc. of the Fall 2003 Simulation Interoperability Workshop and Conference, Orlando, Florida, September 2003.
[14] Jackson, E. Bruce, Murri, Daniel, Hill, Melissa, Jessick, Matthew, Penn, John, Hasan, David, Crues, Edwin Z., Falck, Robert, McCarthy, Thomas, Vuong, Nghia, and Zimmerman, Curtis. Assessment of the Draft AIAA S-119 Flight Dynamic Model Exchange Standard. In Proc. of the AIAA Modeling and Simulation Technologies Conference, Portland, Oregon, August 8-11, 2011.
[15] Jennings, Esther H., Blum, Michael G., Busto, Juan M., Chung, Crues, Edwin Z., Dexter, Dan E., Hasan, David, and Hawkins, Joe, Integrating Collaborative Distributed Simulations for Space Exploration Missions. In Proc. of the AIAA Modeling and Simulation Technologies Conference and Exhibit, Honolulu, Hawaii, August 18-21, 2008
[16] B. Möller, B. Löfstrand. Getting started with FOM Modules, In Proc. of the 2009 Fall Simulation Interoperability Workshop, 09F-SIW-082, Simulation Interoperability Standards Organization, 2009.
[17] B. Möller, A. Falcone, A. Garro, and E. Z. Crues. Experiences in using the SISO Space Reference FOM standard. In Proc. of the 2019 Summer Simulation Conference (SummerSim '19), Berlin, Germany, July 22-24, 2019. The Society for Modeling and Simulation International Inc., 2019.
[18] B. Möller, A. Garro, A. Falcone, E. Z. Crues, and D. E. Dexter. On the execution control of HLA federations using the SISO Space Reference FOM. In Proc. of the 21st IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DS-RT 2017), Rome, Italy, October 18-20, 2017. IEEE Inc., 2017, pp. 75–82. doi: 10.1109/DISTRA.2017.8167669.
[19] B. Möller, A. Garro, A. Falcone, E. Z. Crues, and D. E. Dexter. Promoting a-priori Interoperability of HLA-Based Simulations in the Space Domain: The SISO Space Reference FOM Initiative. In Proc. of the 20th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DS-RT 2016), London, United Kingdom, September 21-23, 2016. IEEE Inc., 2016, pp. 100–107. doi: 10.1109/DS-RT.2016.15.
[20] Paddock, Eddie J., Lin, Alexander, Vetter, Keith, Crues, Edwin Z. Trick® - A Simulation Development Toolkit. In Proc. of the AIAA Modeling and Simulation Technology Conference, Austin, Texas, August 2003.
[21] Phillips, Robert G., Dexter, Daniel E., Hasan, David A., Crues, Edwin Z., A Coordinated Initialization Process for the Distributed Space Exploration Simulation (DSES). In Proc. of the Spring 2007 Simulation Interoperability Workshop and Conference, Norfolk, Virginia, March 2007.
[22] Phillips, Robert G., Crues, Edwin Z. "Time Management Issues and Approaches for Real Time HLA Based Simulations." In Proceedings of the Fall 2005 Simulation Interoperability Workshop and Conference, Orlando, Florida, September 2005.
[23] Pitch Technologies. The simulation toolkit, [online], available http://www.pitch.se, accessed 20 Nov 2019.
[24] ReactiveX. An api for asynchronous programming with observable streams, [online], available at http://reactivex.io/, accessed 20 Nov 2019.

[25] Simulation Exploration Experience (SEE). The Simulation Exploration Experience (SEE) project website, [online], available at http://www.exploresim.com/, accessed 20 Nov 2019.
[26] SISO Space Reference FOM (SRFOM). The SISO SRFOM Product Development Group (PDG), [online], available at https://www.sisostds.org/StandardsActivities/DevelopmentGroups/SRFOMPDGSpaceReferenceFederationObjectModel.aspx, accessed 20 Nov 2019.
[27] Trick Simulation Environment. 2019. NASAs Trick development website, [online], available at https://github.com/nasa/trick/wiki
[28] TrickHLA 2019, Trick High-Level Architecture (HLA) Framework for Facilitating IEEE 1516 Simulation Integration, website, [online], available at https://software.nasa.gov/software/MSC-24544-1
[29] SISO: "SISO-ADM-003-2011 Balloted Products Development and Support Process (BPDSP), www.sisostds.org, November 2011
[30] B. Möller, E. Z. Crues, D. E. Dexter, A. Garro, A. Skuratovskiy, A. Vankov. A First Look at the Upcoming SISO Space Reference FOM. Proceedings of the SISO 2016 Simulation Innovation Workshop (SIW), Orlando, Florida, USA, September 11-16, 2016.
[31] B. Möller, E. Z. Crues, D. E. Dexter, A. Garro, M. Madden, A. Skuratovskiy. Design and Principles Enabling the Space Reference FOM. Proceedings of the SISO 2016 Simulation Innovation Workshop (SIW), Orlando, Florida, USA, September, 2017.
[32] B. Möller. Developing Space Reference FOM Federation Agreements (Presentation only). Proceedings of the SISO 2019 Simulation Innovation Workshop (SIW), Orlando, Florida, USA, February, 2019.
[33] IEEE: IEEE Std 1730-2010 Distributed Simulation Engineering and Execution Process (DSEEP), www.ieee.org, September 2010
[34] SISO: "SISO-STD-012-2013 Standard for Federation Engineering Agreement Template (FEAT), www.sisostds.org, August 2013

Author Biographies

BJÖRN MÖLLER is the President and co-founder of Pitch Technologies. He has more than twenty-five years of experience in high-tech R&D companies, with an international profile in areas such as modeling and simulation, artificial intelligence and web-based collaboration. Björn Möller holds a M.Sc. in Computer Science and Technology after studies at Linköping University, Sweden, and Imperial College, London. He is currently serving as the chairman of the SISO Space Reference FOM Product Development group, chairman of the SISO RPR FOM Product Development Group. and the vice chairman of the SISO HLA Evolved Product Development Group.

EDWIN Z. CRUES has over 25 years of professional experience in developing spacecraft simulation and simulation technologies. Zack is currently a member of the Simulation and Graphics branch at NASA’s Johnson Space Center in Houston, Texas where he leads the development of simulation technologies and the application of those technologies in the simulation of NASAs current and proposed crewed spacecraft. He has developed hundreds of models and simulations for NASA spacecraft including Shuttle, International Space Station (ISS), Orion, Altair, Morpheus and the Multi-Mission Space Exploration Vehicle. Zack’s recent research focus has been developing and applying distributed computation and distributed simulation technologies. This includes a large-scale distributed simulation of NASAs proposed human space exploration missions. Zack also has international experience in developing simulations of European Space Agency launch systems and Japanese Aerospace Exploration Agency spacecraft.

DAN DEXTER is an engineer in the Simulation & Graphics Branch in the Software, Robotics and Simulation Division of the Engineering Directorate at NASA’s Johnson Space Center in Houston, Texas. He has over 22 years of software and simulation development experience ranging from nonlinear signal and image processing, distributed supercomputing, and flight related software to national and international distributed simulations. He is the principal developer of the TrickHLA software package, a NASA developed middleware software package for using the HLA distributed simulation standard with NASA standard M&S tools.

ALBERTO FALCONE is a Postdoc Research Fellow at the Department of Informatics, Modeling, Electronics and Systems Engineering (DIMES) of the University of Calabria (Italy). From January to October 2016 he was a Visiting Scholar/Researcher at NASA JSC working with the Simulation and Graphics Branch. He is a member of the Simulation Interoperability Standards Organization (SISO). He is a member of the Space Reference Federation Object Model (SRFOM) Product Development Group (PDG) of SISO. He is a Member of the Executive Committee of the Simulation Exploration Experience (SEE) project as Technical Advisor. He is Workshop Chair of the 2020 Spring Simulation Conference (SpringSim’20). His email address is alberto.falcone@dimes.unical.it.

ALFREDO GARRO received a PhD in Systems and Computer Engineering from the University of Calabria (Italy), where he is currently an Associate Professor of Computer and Systems Engineering with the DIMES. In 2016, he was Visiting Professor at NASA JSC, working with the Software, Robotics, and Simulation Division (ER). His main research interests include M&S, multi-agent systems, systems and software engineering, and reliability engineering. His list of publications contains about 100 papers published in international journals, books, and proceedings of international and national conferences. He is the co-founder (in 2014) and director (from 2018) of the Departmental Research Laboratory “System Modeling And Simulation Hub Lab (SMASH Lab)”. He is vice chair of the SRFOM PDG of SISO. He is the Vice-President/Next-president of the ‘‘Italian Chapter’’ of INCOSE. He is a member of the Executive Committee of the SEE (Simulation Exploration Experience) project. He is involved as an IEEE Senior Member in the activities of the IEEE Computer Society, IEEE Reliability Society, and IEEE Aerospace and Electronic Systems Society. His email address is alfredo.garro@dimes.unical.it.

ANDREAS RYDELL is a software developer at Pitch Technologies, Sweden. He holds a BSc in Computer Science from Linköping University. He has five years of experience in developing HLA based simulation for defense and space applications.