

Model-Based Systems Engineering for Capturing Mission Architecture System Processes with an Application Case Study – Orion Flight Test 1

Kevin H. Bonanne

Purdue University, West Lafayette, IN 47906

with mentoring from

Oleg V. Sindiy, Brian Giovannoni, and Kim Simpson

Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109

Model-based Systems Engineering (MBSE) is an emerging methodology that can be leveraged to enhance many system development processes. MBSE allows for the centralization of an architecture description that would otherwise be stored in various locations and formats, thus simplifying communication among the project stakeholders, inducing commonality in representation, and expediting report generation. This paper outlines the MBSE approach taken to capture the processes of two different, but related, architectures by employing the Systems Modeling Language (SysML) as a standard for architecture description and the modeling tool MagicDraw. The overarching goal of this study was to demonstrate the effectiveness of MBSE as a means of capturing and designing a mission systems architecture. The first portion of the project focused on capturing the necessary system engineering activities that occur when designing, developing, and deploying a mission systems architecture for a space mission. The second part applies activities from the first to an application problem – the system engineering of the Orion Flight Test 1 (OFT-1) End-to-End Information System (EEIS). By modeling the activities required to create a space mission architecture and then implementing those activities in an application problem, the utility of MBSE as an approach to systems engineering can be demonstrated.

Nomenclature

<i>EEIS</i>	End-to-End Information System	<i>JSC</i>	Johnson Space Center
<i>EHM</i>	Europa Habitability Mission	<i>KSC</i>	Kennedy Space Center
<i>FDD</i>	Functional Design Document	<i>MCC</i>	Mission Control Center
<i>GDS</i>	Ground Data System	<i>MBE</i>	Model-Based Engineering
<i>GDSE</i>	Ground Data System Engineer	<i>MBSE</i>	Model-Based Systems Engineering
<i>GDSSE</i>	Ground Data System Software Engineer	<i>MOS</i>	Mission Operations System
<i>HEFT</i>	Human Exploration Framework Team	<i>MOSE</i>	Mission Operations System Engineer
<i>IEEE</i>	Institute for Electrical and Electronics Engineers	<i>MPCV</i>	Multi-Purpose Crew Vehicle
<i>IRD</i>	Interface Requirement Document	<i>MSA</i>	Mission Systems Architecture
<i>JEO</i>	Jupiter Europa Orbiter	<i>OFT-1</i>	Orion Flight Test 1
<i>JPL</i>	Jet Propulsion Laboratory	<i>SysML</i>	Systems Modeling Language

I. Introduction

This project focuses on using Systems Modeling Language (SysML)^{1,2} to capture the processes of two efforts: Section 318's Mission Systems Architecture Process Modeling and Orion Flight Test 1 (OFT-1) End-to-End Information Architecture Modeling. The two different, but related, efforts emphasize the use of

model-based systems engineering (MBSE) in order to capture the full scope of a ground data system (GDS) architecture. The first portion of the project focuses on capturing the system engineering activities that occur when designing, developing, and deploying a mission systems architecture. On the other hand, the OFT-1 segment centers around an application problem, which implements some of the activities modeled in the former project. For both aspects of this project, the SysML package in a modeling program called MagicDraw was implemented for model creation.

Model-based systems engineering is notable as a powerful way to improve on system development processes.³ Model-based engineering (MBE) pertains to “elevating models in the engineering process to a central and governing role in the specification, design, integration, validation, and operation of a system”.⁴ Among many benefits, MBSE can be invaluable as a method to communicate ideas to stakeholders, where old-hat formats like documents and presentation slides often fall short.⁵ Using a system modeling tool, in my case MagicDraw,⁶ to develop these models both increases the awareness about the system(s) and allows for an easier method to communicate the information to stakeholders.

The Mission Systems Architecture Process Modeling portion of this project focuses on modeling the activities of a mission systems architecture (MSA) through all phases of a mission. A MSA encompasses all of the capabilities, functions, software, and hardware. It also describes how the entire mission will operate, covering teams, roles, processes, and procedures involved in the support services provided by the GDS and MOS of the MSA. Currently, there are several documents with differing information on what the major activities for the section are when creating a MSA. To further complicate things, many of these documents use different nomenclature and organizational structure when referring to similar tasks or deliverables. To improve this menagerie, a model-based systems engineering approach was adopted and a SysML model of the section’s processes was created. This model organizes the activities and artifacts over Phase A of the flight project life cycle into one easy-to-understand, visual model. In the future, more phases will be added, but the problem was scoped to Phase A to serve as a proof of concept. What is defined in this paper is an example application of our developed ontology to phase A processes; further use of the model will be performed with feedback from subject matter experts from various sections at JPL.

The second portion of this project deals with modeling the EEIS of the OFT-1 mission. OFT-1 will be the first test flight of the Orion Multi-Purpose Crew Vehicle (MPCV). The spacecraft will be launched from Cape Canaveral and land in the Pacific Ocean. The purpose of the flight test is to ensure proper flight operation of select avionics, structural, and thermal hardware and software on the MPCV. The operation of the flight test will require a number of different entities to cooperate between various NASA centers and supporting entities as well as Lockheed Martin, who was the primary contractor in charge of building and operating the MPCV.¹⁰ Because of this, a large architectural framework is necessary to coordinate all processes before, during, and after the test. Though much work was done on improving the OFT-1 model in various ways during my internship, this paper will solely focus on the effort to create a traceable relationship between existing Interface Requirement Documents (IRDs) and the connections in the various diagrams of the OFT-1 model.

The rest of the paper will be organized as follows: Section II will cover the background motivation and goals for this research; Section III will overview the steps taken to accomplish those goals; Section IV will discuss the specific findings of the research; and Section V will outline how this work can and will be continued from this point forward.

II. Motivation

The motivation behind both aspects of this project is to move to a model-centric approach to systems engineering through previously defined languages, such as SysML, and tools to utilize them, such as MagicDraw. For both portions of the project, the application of MBSE aims to consolidate information into a central location, intended to function as a single source of truth (i.e., the model), and increase comprehension of that information.

As previously stated, information on the major activities required for creating a MSA is distributed amongst three different documents. Each document has different levels of detail when describing these activities. By consolidating this information into a single model, two major issues can be absolved – granularity and consistency. A common granularity, or level of detail, is obtained by abstracting all activities and artifacts to a common base level (what is later identified as “basic elements”). This abstraction also creates a consistency amongst the different processes, as all are generated using the same method and model.

Comprehension of information of MSA processes is obtained through the introduction of MBSE as well. The motivation here is to increase traceability and clarity. Traceability is realized in the MSA Process Modeling effort by showing the relations between activities and the artifacts that are produced by them – the end products that are delivered after the completion of activities. Further traceability can be observed by showing how the new model equates to the existing documentation. Clarity is inherent to MBSE, as often a diagram can show much more than words can alone.

The OFT-1 effort also focuses on consolidating information provided from various stakeholders into one model. A central portion of this work concentrated on generating traceability throughout the model. The end goal of traceability within this paper is to be able to determine exactly how much of the end-to-end information system (EEIS) was realized by existing Interface Requirement Documents (IRDs). The motivation behind this is to identify easily any missing IRDs in order to notify stakeholders.

III. Approach

As stated in the Motivation section, the key reasons to applying MBSE to both portions of this project are to increase consolidation and comprehension of information. The MSA Process Modeling portion thus focuses on creating an entirely new taxonomy and abstraction to model the activities utilized to create a MSA. On the other hand, the OFT-1 effort applies many of the same ideas to model an existing MSA. This section will describe the approach taken to accomplish these goals.

The first step in creating a model to describe the activities required for generating a MSA was to perform a literature review of the existing documentation on the subject. Three primary documents were identified on JPL Rules! that contained said information – *Mission Operations System Engineering (MOSE)*,⁷ *Mission Ground Data System Engineering (GDSE)*,⁸ and *Ground Data System Software System Engineering (GDSSE)*.⁹ For the sake of simplicity, only Phase A processes were examined. Furthermore, information from existing gate transition products and the JPL Rules! document, *System Engineering Practices (SEP)*,¹¹ were examined for more background on MSA SE artifacts and activities.

The next step taken was to begin abstracting the existing information into a taxonomy that could both consolidate and clarify the existing documentation. A generic taxonomy was created (by Oleg Sindiy) by uniting the essential elements of various existing architecture definitions. These architecture descriptions came from the Human Exploration Framework Team (HEFT),¹² the Institute for Electrical and Electronics Engineers (IEEE) Standard 1471,¹³ and JEO-EHM's proprietary architecture description.¹⁴ Further abstraction saw the decomposition of high-level gate transition products into low-level basic elements, which were then used to create the major viewpoints in each gate transition product. More detail will be given on this process in the Results and Discussion section.

The final step was to apply the developed abstraction and taxonomy. To do this, the three JPL Rules! documents were used again. Activities were created based on the abstracted elements. Then, these activities were mapped to the processes described in the documents. An example of the mapping can be seen in the Results and Discussion section (specifically Figure 4).

The OFT-1 effort to create IRD traceability in the existing model began by identifying the existing IRDs. From there phase-independent data exchange blocks were created. The blocks allow for an intermediary step between the IRDs, which are phase-independent, and the phase-dependent model diagrams. After all data exchange blocks are created relationships are drawn between the interactions on the diagrams and the blocks. Then the blocks are related to the IRDs. By creating these relations, a mapping is produced between the IRDs and the actual interactions in the model. This is then exported to a customized table for ease of interpretation.

IV. Results and Discussion

Section 318's Mission Systems Architecture Process Modeling

Data was gathered on the processes and artifacts required during Phase A of the development of a MSA. This information was stored on a JPLWiki website for ease of access by the entire team. After gathering all the required information from the literature review onto the JPLWiki, an effort was made to transfer this information into a formal SysML model utilizing the MagicDraw modeling tool.

In my team's first attempt, activities were created for each of the process outlined in the reviewed system

engineering definition documents (found on JPLRules!) and utilized a series of artifacts (e.g. data, papers, information) that were also retrieved from those documents. For example, an activity would accept a flow of several levels of requirements and produce from those a flow of “key and driving requirements”. However, during this modeling effort it was discovered that the discrepancies between the different levels of abstraction and detail found in the various documents caused similar discrepancies in the model – an outcome that was hoped to be resolved by application of MBSE. Because of this issue, it was decided that a new architectural taxonomy was necessary to properly model the processes involved in developing MSA. An image showing an overview of the developed taxonomy can be seen below in Figure 1. Note that the taxonomy shown is subject to change as more feedback is received from subject matter experts.

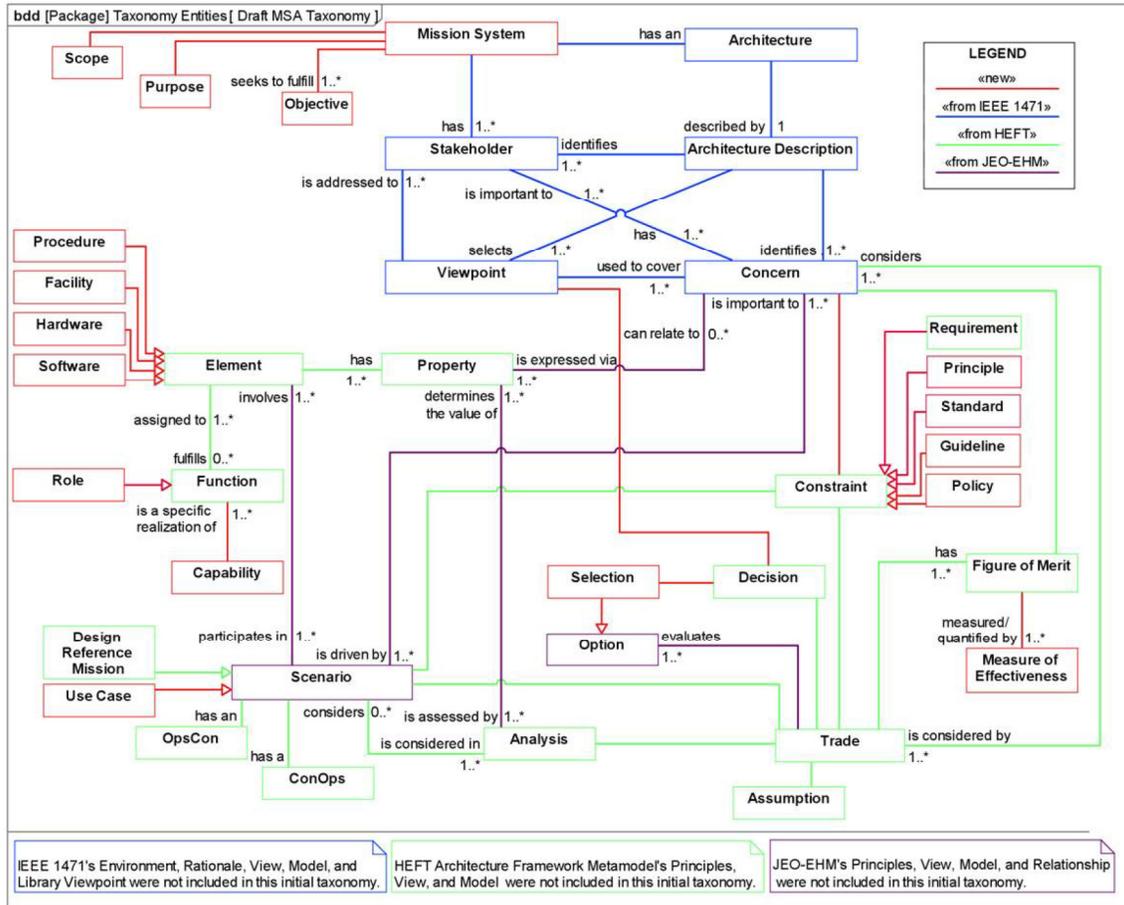


Figure 1. DRAFT overview of the new taxonomy developed by our team. It is a combination architecture descriptions from HEFT, IEEE 1471, and JEO-EHM with some new components (see legend).

The building blocks for this architectural description consist of three different levels of artifacts. These levels were defined as “basic elements”, “viewpoints”, and “gate transition products”, where basic elements are the most atomic and gate transition products are most complex. The process used to aggregate all of the necessary artifacts uses a top down approach. First, the major gate transition products were identified for Phase A. A gate transition product is often an important paper, report, or presentation that allows the team to move onto the next step in the mission architecture development process. The example below in Figure 2 uses the GDS Functional Design Document (FDD).

From documentation on how to create these documents and examples from other missions, the major components, what we call viewpoints, can be identified. In the example below, the GDS FDD contains a work breakdown structure, requirements information, and a mapping of GDS functions to MOS capabilities, as well as some basic elements that define the mission. Each viewpoint is then decomposed into basic elements. For example (as seen below), a work breakdown structure is composed of a number of teams and the roles that those teams complete. Some things that this diagram does not show is *how* the basic elements under

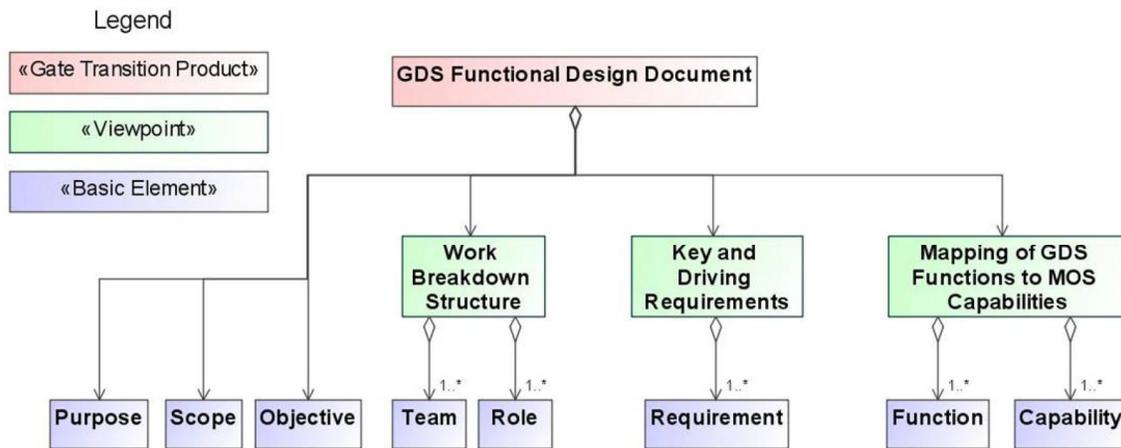


Figure 2. Example decomposition showing the three levels of artifacts

the same viewpoint interact and *what* must be done to create this interaction.

The *how* question is answered by the architecture taxonomy. This defines the basic relationships between all basic elements (see Figure 1). The *what* question is answered by incorporation of activities into the model. As stated previously, our first approach at capturing these processes began by focusing on activities. This did not work at the time because the architecture lacked structure, due to the inconsistencies between reference documents. With the work that has been done so far (and the inclusion of a complete taxonomy), activities can be defined properly. In this sense, an activity will be required to realize the relationship between any two artifacts. For example, to create a work breakdown structure, an activity, “Generate work breakdown structure”, will have to be completed. This activity will take the teams and roles and organize them into a structure (perhaps a table). This activity relationship can be seen in Figure 3, below. Many relationships between basic elements and viewpoints have already been captured with the use of activities; however, more information will be needed from subject matter experts to capture more activities as well as artifacts.

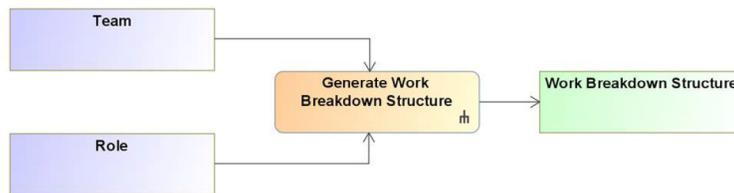


Figure 3. Example of activity bridging the gap between basic elements and viewpoints by mapping teams and roles into a work breakdown structure.

As stated in the Approach section, the final step performed in the MSA Process Modeling task was to apply the newly created abstraction and taxonomy to the existing JPL Rules! documents. This ensures that the actual content is not being changed at all, only the method with which it is being represented. This was a fairly easy task as many of the elements and viewpoints could be directly related to the verbiage used in the papers. For example, “prepare development cost estimates” from the GDSE document⁸ is straightforwardly modeled by the “develop cost estimates” activity, which creates a “cost estimate” artifact. Some selected examples of this mapping can be seen in the figure below (Figure 4).

Orion Flight Test 1 End-to-End Information Architecture Modeling

The OFT-1 EEIS is an application project of the architecture definition activities done in the Mission Systems Architecture Process Modeling task. Much of the model has been created (i.e. blocks and activities are there) but the architecture is constantly evolving and becoming more defined. Thus, the work on OFT-1 is more dedicated to refinement and upkeep rather than creation.

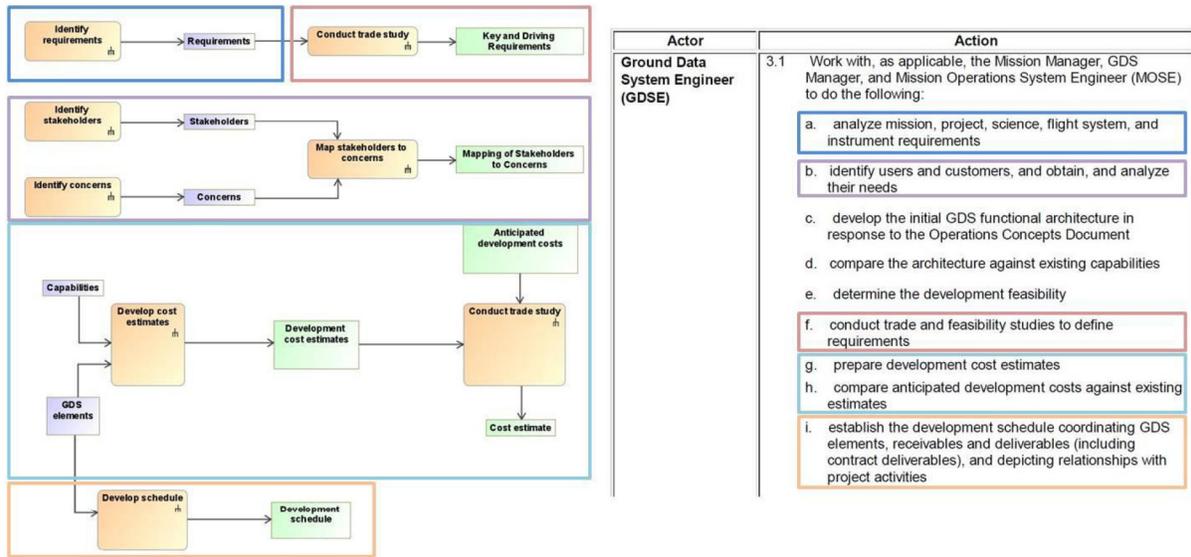


Figure 4. Selected activities and artifacts from the model (left) and their corresponding portions in a JPL Rules! document (right).

The major refinement to the OFT-1 model that this paper focuses on is the mapping of IRDs to the existing model connections. A majority of these occur in a viewpoint of diagrams called needlines. Needlines show the high-level data flows amongst systems in the MSA. Each data flow is created by drawing a connection between ports on two different system blocks and then specifying the conveyed information over that connection. By doing this, the direction of the data flow (unidirectional or bidirectional) and the type of data flowing can both be specified. An example needline diagram is shown in Figure 5.

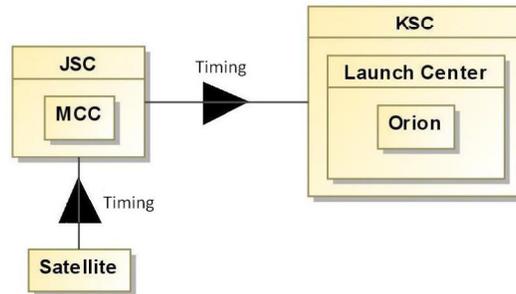


Figure 5. Example needline diagram showing data exchanges between multiple systems.

In the above example, a satellite sends timing data to Johnson Space Center (JSC), which contains the Mission Control Center (MCC). JSC then sends that timing data to Kennedy Space Center (KSC), which contains a launch pad holding the Orion capsule. It should be noted that this is purely an example to demonstrate what a needline diagram could look like; most needline diagrams in the OFT-1 model are much more complex.

Because the needline diagrams are phase-dependent (i.e., their configuration may change with time), yet the IRDs are phase-independent, an intermediary stage must be created to relate the two. To accomplish this, “data exchange” blocks are created, which define a data exchange between two systems. Each data exchange block defines the sending system, the receiving system, and the data that is exchanged. To match the example above, a data exchange block may be named “Satellite - JSC - Timing”.

The final step is to create relations between the three stages. First, to connect the needline to the data exchange, an abstraction is created. Thus, “Satellite - JSC - Timing” is an abstraction of the left-most needline connection. After this, that data exchange must be related to an IRD. Say there exists an IRD that states, “Satellite shall send timing information to Mission Systems”. The “Satellite - JSC - Timing”

data exchange would *satisfy* that requirement (a satisfy relationship would be established between the two). Finally, a customized table can be generated to show all of the connections in every needline, what each one's linked data exchange is, and if it satisfies an IRD. For the purposes of our model, needlines were color coded (via SysML stereotypes) on whether or not they require an IRD. At the end of this effort, it could be observed which needlines satisfied an IRD, which required an IRD, and which did not require an IRD.

V. Future Goals

The next steps in developing a new ontology for describing MSA processes is to consult subject matter experts on how to continue to apply our ontology to their work. Though we have developed a more descriptive method for defining the MSA processes, assistance must be provided to properly capture every activity and artifact. The work done in this paper only covers the initial application of our methodology to Phase A. More work will have to be done to capture the other phases and refine the information. Currently, it is planned to use the JPLWiki to distribute information on our model and gather feedback from subject matter experts.

Other improvements our ontology include adding a method for dealing with evolution in artifacts. Currently, artifacts can only be typed, but the model does not describe what information is contained in that artifact at what time. For example, a FDD in phase A, when the architecture is just starting to be defined, will contain much different information that it would contain during phase D, when the architecture is being physically implemented. Some methods from the OFT-1 model may be applicable to capture this evolution, but none have been applied at this time.

Future work in the OFT-1 model will continue with the theme of refining the model. Though much effort must be made in upkeep of the model (i.e., stay consistent with programmatic changes to the mission), some changes can still be made to improve the model. Further traceability throughout the model is still possible. Many of these changes would follow a similar method to what was outlined in this paper of tracing IRDs to needlines.

Acknowledgments

I would like to thank Oleg Sindiy and Brian Giovannoni for their guidance during this internship. I would like to acknowledge NASA, JPL, CalTech and their Student-Faculty Program, and Indiana Space Grant Consortium for giving me the opportunity to perform this research. Lastly, I would like to thank Thom McVittie, Kim Simpson, and Joseph Tirona, as members on the OFT-1 architecture team. This work was funded by the ISGS and OFT-1 projects. This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by Indiana Space Grant Consortium and the National Aeronautics and Space Administration.

References

- ¹Object Management Group, *OMG Systems Modeling Language*, Version 1.2, June 2010.
- ²Object Management Group, *OMG Unified Modeling Language*, Version 2.3, May 2010.
- ³Peukert, Andreas, *S/C Behavior Modeling Using SysML for Model-Based Systems Engineering Support*, AIAA Space 2008 Conference & Exposition, 2008.
- ⁴Estefan, J.A., *Survey of Model-Based System Engineering (MBSE) Methodologies*, Rev. B, INCOSE MBSE Initiative, 23 May 2008, p. 10.
- ⁵Shames, P. and Skipper, J., *Toward a Framework for Modeling Space Systems Architectures*, SpaceOps 2006 Conference, 2006.
- ⁶*MagicDraw: Architecture Made Simple*, No Magic, Inc., 2011, http://www.magicdraw.com/what_is
- ⁷Herrera, Randy, *Mission Operations System Engineering*, Rev. 3, DocID 26912, JPL Rules!, Dec 3, 2010.
- ⁸Giovannoni, Brian, *Mission Ground Data System Engineering*, Rev. 3, DocID 27072, JPL Rules!, Dec 6, 2010.
- ⁹Ko, Adans Y., *Mission Ground Data System Software System Engineering*, Rev. 1, DocID 72192, JPL Rules!, Nov 14, 2005.
- ¹⁰*About the Multi-Purpose Crew Vehicle (MPCV)*, NASA, <http://www.nasa.gov/exploration/systems/mpcv/>
- ¹¹Brown, Mark, *Systems Engineering Practices*, Rev. 1, DocID 75012, JPL Rules!, May 5, 2009.
- ¹²*Human Exploration Architecture Framework*, Draft Rev. 1.0, NASA Internal Document, Dec 1, 2010.
- ¹³*Systems and Software Engineering – Recommended Practice for Architectural Description of Software-Intensive Systems*, 1st Ed., ISO/IEC 42010, IEEE Std 1471-2000, July 15, 2007.
- ¹⁴“Architecture Description”, Ver. 9, JEO-EHM, JPL-CalTech Internal Document, Accessed Aug 15, 2011.