

Update – Concept of Operations for Integrated Model-Centric Engineering at JPL

Todd J. Bayer, Matthew Bennett, Christopher L. Delp, Daniel Dvorak, J. Steven Jenkins, Sanda Mandutianu
NASA Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California, 91109
Mail Stop 301-230
818-354-5810
Todd.J.Bayer@jpl.nasa.gov

Abstract—The increasingly ambitious requirements levied on JPL’s space science missions, and the development pace of such missions, challenge our current engineering practices.^{1,2} All the engineering disciplines face this growth in complexity to some degree, but the challenges are greatest in *systems* engineering where numerous competing interests must be reconciled and where complex system-level interactions must be identified and managed. Undesired system-level interactions are increasingly a major risk factor that cannot be reliably exposed by testing, and natural-language single-viewpoint specifications are inadequate to capture and expose system level interactions and characteristics. Systems engineering practices *must* improve to meet these challenges, and the most promising approach today is the movement toward a more integrated and model-centric approach to mission conception, design, implementation and operations. This approach elevates engineering models to a principal role in systems engineering, gradually replacing traditional document-centric engineering practices.

JPL’s Integrated Model-Centric Engineering (IMCE) Initiative seeks to lay the foundation for improvement of systems engineering practices. The deployment of such an initiative must address many questions. What kinds of models are needed? What modeling languages should be used? How do different engineering disciplines collaborate? How should design artifacts change? Infusion of model-centric processes into JPL’s engineering culture will require, among other things, the construction of model-centric analogues corresponding to the current processes, in each technical domain and at each phase of a project’s lifecycle. To accomplish this it is necessary to understand the processes we are attempting to transform. The Concept of Operations, initially reported on in [1] during its development, describes key processes which flight projects execute, in a ‘before’ (document-centric) and an ‘after’ (model-centric) view. In the process, we have taken the first steps toward recasting the NASA Flight Project Lifecycle gate products and gate review success criteria into model-centric terms. The approach was to focus on small, almost atomic, pieces of this process, in order to identify the

architectural patterns of interest. The Concept of Operations (ConOps), now published [2], is a key input to the IMCE design, and will also help future user projects to understand and ultimately embrace IMCE.

The Concept of Operations focuses on specific areas of project engineering that would have the largest beneficial impact on project development and operations, as well as those that are needed for early infrastructure development. Key benefits described include: reuse of engineering products between missions and throughout a mission’s lifecycle; continuous integration of engineering development processes throughout a project’s lifecycle, across multiple disciplines and domains; continuous automated validation and verification of mission and spacecraft designs and implementations; automated creation of review gate products; automated verification of technical design budgets and margins; architectural design viewpoints on integrated model-based system descriptions across multiple disciplines and domains; integration and management of model development. As part of the development of the operations concept, model-based scenario and document generation processes were explored using SysML.

TABLE OF CONTENTS

1. INTRODUCTION	1
2. APPROACH AND METHOD	2
3. ANALYSIS	4
4. EXAMPLE: PHASE A SCENARIO DEVELOPMENT	5
5. EXAMPLE: CAPTURING THE DESIGN	12
6. FUTURE WORK	14
7. CONCLUSION	14
ACKNOWLEDGEMENTS	14
REFERENCES	14
BIOGRAPHIES	15

1. INTRODUCTION

This document reports the results of the Integrated Model-Centric Engineering (IMCE) Concept of Operations (ConOps) study. Review and use of this report will help determine if additional study is needed, which would be documented in future releases of this report. This ConOps Report was intended for two distinct audiences within JPL:

¹ 978-1-4244-7351-9/11/\$26.00 ©2011 IEEE

² IEEEAC paper#1122, Version 6, Updated 09 Jan 2011

- (1) The IMCE Architecting team. This report captures the viewpoints and concerns of the category of stakeholders who are users of IMCE capabilities. (It is important to note here that IMCE is the task and not the system of interest. IMCE is not a system which will be delivered. The system of interest here is the JPL flight project development enterprise. IMCE as a task will deliver targeted capabilities into this enterprise in order to enable its transformation into a more model centric one. One of the key stakeholder groups (those with a stake in the enterprise) is the flight project engineers. These stakeholders will be users of IMCE-delivered capabilities.)
- (2) Potential adopters, users and funders of IMCE capabilities. This report is intended to convey a more concrete sense of what it will mean to develop flight projects in a model-centric way.

As chartered by IMCE in March, 2009, the following were the goals and associated objectives for this effort: *Goal 1:* The IMCE Team understands and is able to articulate to project users why project users should, and how they can, use IMCE to accomplish a model-centric project development.

Objective 1A: Collect, synthesize and document significant problem areas in the current flight project development environment, and how IMCE addresses them. (This is now documented separately in an internal memorandum [3]).

Objective 1B: Describe, through development of selected scenarios, how people supporting a JPL flight project would interact with models, model-centric tools, and each other throughout the project life cycle

Objective 1C: Discover and document the characteristics of a flight project lifecycle as executed in a model-centric environment.

Goal 2: The IMCE Team has valid use cases upon which to base the IMCE Architecture.

Objective 2A: Deliver the Concept of Operations with form and content appropriate for the IMCE Architecting work and the Architecture Description Document.

2. APPROACH AND METHOD

This section describes our approach to developing a concept of operations for JPL flight project development in a model-centric engineering environment. A key assumption behind the ConOps is that JPL's established engineering lifecycle for flight projects is a mature and successful roadmap. Therefore, the ConOps is intended to help flight project engineers understand how to change from document-centric to model-centric methods while still following the existing engineering lifecycle. The approach was to agree on which flight project engineering scenarios would be most fruitful

to model. Phase A (concept formulation and technology development) was selected as the starting phase due to its being a key junction between early formulation efforts such as TeamX and project formulation/implementation. Within Phase A, a set of scenarios was selected and prioritized. The full set of selected scenarios, as well as the current state of completion, is shown below in Table 1, "Selected Scenarios".

Within those scenarios, it was found that a useful technique was to focus on small, almost atomic, pieces of these processes, in order to identify architectural patterns of interest. Small teams composed of modelers and flight project subject matter experts developing the use cases. Weekly reviews with the full team as well as two large workshops provided guidance and feedback.

Development of the ConOps was a learning exercise that paired young engineers with seasoned experts. The approach emphasized the process as much as the product, and the choice of languages and tools used in the modeling and report production, as described below

Use of SysML.

The IMCE Team chose SysML as the modeling language because, in addition to other benefits mentioned later, it explicitly contributed to the project goals of: (1) Demonstrate how a visual modeling tool captures and conveys the complexities of flight project development, and (2) Provide team members an opportunity to gain or improve their skill level with the SysML language, tools and techniques.

Use of Automated Document Generation.

We intentionally chose to produce this report directly from the SysML model for several reasons. First, to follow the guiding principle that all information should have a single identifiable source of truth, in this case the SysML model. The common labor-intensive step of creating a separate Word document containing the interpretive prose, and cut and pasted diagrams from the analysis violates this principle, as well as being wasteful. By placing the interpretive prose directly into the model and then automatically generating the report from it we have remained true to the principle and eliminated an unnecessary step. This method had a substantial learning curve and some additional overhead compared with the Word document method, but the benefits have been substantial as well, and we believe more than compensate for the overhead, described more below. Second, to enable future updates to be published in the most efficient way. Third, to help refine the techniques and tools for producing auto generated documents, which we believe will be a major part of a model-centric environment. And fourth, as in the choice of SysML, to provide an education/training opportunity for the team.

Table 1 Selected Scenarios

IMCE Operations Concept - Scenario Modeling Plan		High: design drivers which are also a high priority for JEO focus	
Rev 08 July 2010		Med: additional design drivers but not necessarily JEO priority	
		Low: other	
Functional Group	Scenarios	Priority	
		H-M-L	Status
Flight Project Lifecycle	Flight Project Lifecycle	L	
	Phase A-B Transition	L	
Mission Scenario Development/Use	Phase A Mission Scenario Development	H	Complete 3/4/10
Engineering Analysis	Phase A Trade Studies	H	Complete 6/2009
	Revisiting an Earlier Trade	L	
	Dynamics & Control Analysis	H	not started
	System-to-Domain Model (e.g., SysML to Cielo)	L	
	Early Mission Design Exploration	M	
	Development of SMF	H	not started
	Point to Point vs Repository Mechanisms	L	
	Development of Mission Fault Tree	M	
Requirements	Phase A Requirements Development	H	Complete
	Requirements Allocation to Functions and Verification Events	L	
	Requirements Linking and Tracing	L	
	Integrating System Model with Requirements Management	H	Part 1 Complete 2/4/10
Design Capture and Management	Capture the Design	H	Complete 7/5/10
	Obtaining Approval for Proposed Design Changes at Project CCB	M	
	Promulgating Changes into the System Design	M	
	Tracking/Reporting Technical Budgets	H	Complete 1/28/10
	Managing Risk	L	
Reviews	Producing Required Gate Products	H	Complete 7/12/10
	Making Assertions about the Design	H	Complete 7/12/10
	Conducting Line Peer Reviews	L	
System Modeling	Building a System Model	H	Complete 7/12/10
	Administering a System Model	H	Complete 7/12/10
	Validating the System Model	H	not started
	Archiving a System Model	M	
	Creating System Model Library	M	
	Inheriting, Modifying, Reusing System Models	M	
Issue Management		H	not started

The Concept of Operations is a familiar engineering product with a well-established history of usage. In a larger context, however, it can be seen as a specific set of views in an architectural description as specified in [ISO/IEC 42010:2007](#) [4]. In ISO 42010 parlance, an architectural description of a system is organized by a set of Views, each of which conforms to a Viewpoint that is used to cover one or more Stakeholder Concerns.

For the Concept of Operations, the system stakeholders are mainly its users, along with those who operate and maintain the system. Their concerns are primarily related to functional capabilities, workflow, dependencies, performance, generated products, etc. Other concerns (e.g., system acquisition cost) while important to certain stakeholders, are typically not in the scope of the ConOps.

In keeping with this approach, we took some care to identify the specific concerns applicable to each operational scenario. We then envisioned and captured the specifics of each scenario so as to cover the concerns.

3. ANALYSIS

The approach to analyzing the scenarios simply compares the as-is and the to-be views of the scenario with the IMCE principles in mind. After the scenarios as-is and to-be state were modeled, the models were compared to discover advantages and disadvantages between the two states. Given the assumption that the current life-cycle road map is valid, the comparisons mainly focused on the capture and management of the design information with particular focus on dependencies across roles, disciplines and lifecycle gates. Some scenarios had the additional benefit of looking at this in collaborative engagements like trade studies or the exchange of information between 2 engineering roles responsible for information dependent on the others data. This technique born out of enterprise architecture, was not only effective, but helped to set the stage for understanding the flight project development effort as an enterprise with an architecture setting the stage for the ConOps to lead into a more architecturally driven approach to IMCE.

Advantages noted in model-centric scenarios (compared to current practice)

The current practices revealed that there is much variation in the current practice functions for the as-is flight development enterprise. Models are developed ad-hoc to solve specific problems, and processes for using those models to determine the requirements, trade-offs, design or architecture capture. This compounded by the fact that critical and expensive design information in the form of parameters and models are stored manually with multiple duplicates and emails of various versions flowing around the system. In general it is an expensive, labor-intensive, and error-prone practice. Engineers are forced to spend much of their time manually managing information shared

between office productivity tools (tools that were never designed or intended to develop deep space missions).

The scenarios based on a to-be model based paradigm were successful in reflecting different concepts related to IMCE principles like having a single source of truth. This is not by accident, since Model Based Systems Engineering and even systems engineering as a discipline to a lesser extent was conceived to deal with these problems. The process of modeling the concept of operations for an integrated model-centric environment yielded some key advantages of using models as the basis for engineering.

- High-fidelity Collaboration among engineering roles/teams within a disciplines
- High-fidelity Collaboration between different disciplines
- Integrity of captured design information, analyzed and represented as gate products or other products
- Richer choices for analysis
- Enables richer stakeholder driven analysis

Many of these advantages are the same advantages that stand-alone or domain specific modeling tools provide. By integrating these tools, the integrity and fidelity of information exchange between them compounds the benefit.

Implied Capabilities (Proto-Requirements)

Another aspect of the analysis involved capturing the initial capabilities that were assumed or implicated by the to-be scenarios. Integrated Model Centric Engineering does not imply that models are only used as the basis of the engineering within the individual disciplines but that the modeling across disciplines synergistically enable a gestalt which provides benefits to all stakeholders above and beyond the benefits within each community. The scenarios are generated as a conceptual product, which means they are fairly unrestricted in terms of what they describe. Once completed, each scenario was analyzed for what capabilities are assumed to be in existence by the scenario. This provides some insight in what would be needed in terms of an actual environment. Looking across the scenarios, the following capabilities are assumed:

- Ability to construct engineering disciplines models
- Ability to integrate those models in a useful way
- Collaborate within and across different models
- The ability to version manage and configuration manage models
- The domain knowledge to properly codify both the models and the rules checking needed to verify assertions of both institutional practices and domain specific knowledge about space systems
- Institutionally supported models that are invariant between missions
- Ability to generate reports, artifacts and products

These capabilities are a synthesis of what all the scenarios imply at some level with the additional benefit of looking at the whole collection. In general, modeling can be supported by a core set of capabilities.

Assessment of techniques used in the ConOps effort

As an early demonstrator of MBSE itself, the IMCE ConOps is built using MBSE. The scenarios for the ConOps were modeled using the SysML modeling language and the complete IMCE ConOps report was generated directly from the model. This allowed the ConOps to serve as a demonstration of the techniques its contents advocate.

The use of SysML was an enormous benefit to this work. Because the scenarios were modeled with SysML, they facilitated a much more focused analysis of the behavioral concepts they depict. Instead of being limited by the size of a power point slide or an 8x11 printed page, the team could focus on using the syntax of the language to explicitly reflect the scenario. The language forced the explicit description as well as providing an immediate understanding of what had been modeled. The team also did not require multiple and varying chart legends and notional symbols. Instead the team focused on engineering the scenario according to the semantics of the language until they could agree that the model represented the correct representation of the scenario. Often this surfaced issues that in a notional view graph world would have gone unnoticed.

Using a modeling language also opened the door for automated product generation. The team was able to generate the ConOps using the Doctimus Prime app and meta-model for documents [5]. Being able to generate a report from the model provided more freedom to focus on collaborating on modeling the scenarios. It also allowed a step towards being able to analyze and reason about the scenarios to create other views like tables and matrices as well as validate assertions about the scenarios.

Collocating the documentation with the model makes model readers efficient in browsing specifications and explanations. Models are not only parameters to analytical processing but are explanations of present or future systems to stakeholders; they therefore should include explanatory prose in addition to their specifications of structures, behaviors, and constraints.

The chosen document generation method simplifies production of later versions of the same document, or even quite different documents from the same model. For example, although the model was constructed to describe an operations concept for IMCE, the resultant model could be analyzed many other ways such as a detailed stakeholder and concerns analysis, and reports could be easily generated for any of those other analyses, all from the same source SysML model

This style of working provided the team with some experience of the work-flow of constructing model-based products they usually build manually out of the models they were making. Without being able to tie models directly and easily to deliverable products, it is usually too laborious to build the products separately and modeling gets set aside for lack of time.

Using the approach to make the ConOps which it was itself advocating was a useful experience for the team, who were able to experience an MBSE environment first hand. The result is that the team can now say we've "practiced what we preach" and experienced the benefits first hand.

Of the ten scenarios completed for this version of the ConOps, we next summarize two of them as examples: Phase A Scenario Development, and Capturing the Design.

4. EXAMPLE: PHASE A SCENARIO DEVELOPMENT

Description/Objective/Scope

- Concepts for how scenarios are represented and built during Phase A of the project lifecycle.
- Concepts for how scenarios reference models of system model elements.
- Concepts for what projects do with scenarios in phase A, including validation of mission and spacecraft design against science and engineering objectives, requirements, and constraints.

Concerns

This scenario addresses a set of concerns, organized into 4 areas as shown in Figure 1:

- Science and Mission – Concerns related to mission and science objectives.
- Health and Safety – Concerns related to the health and safety of the spacecraft, and to planetary contamination. Could also include concerns related to launch operations.
- Schedule – Concerns related to mission events and their timing.
- Resources – Concerns related to mission resources (flight and ground) consumed during operations.

Scenarios

The scenario for Phase A Mission Scenario Development and Use is shown in Figure 3, and is composed of six constituent scenarios as shown in Figure 2. The following constituent scenarios are described in more detail below:

- Create Mission Scenario
- Evaluate and Assess Mission Scenario

Both of the scenarios reference the concepts of a "mission scenario" and a "timeline".

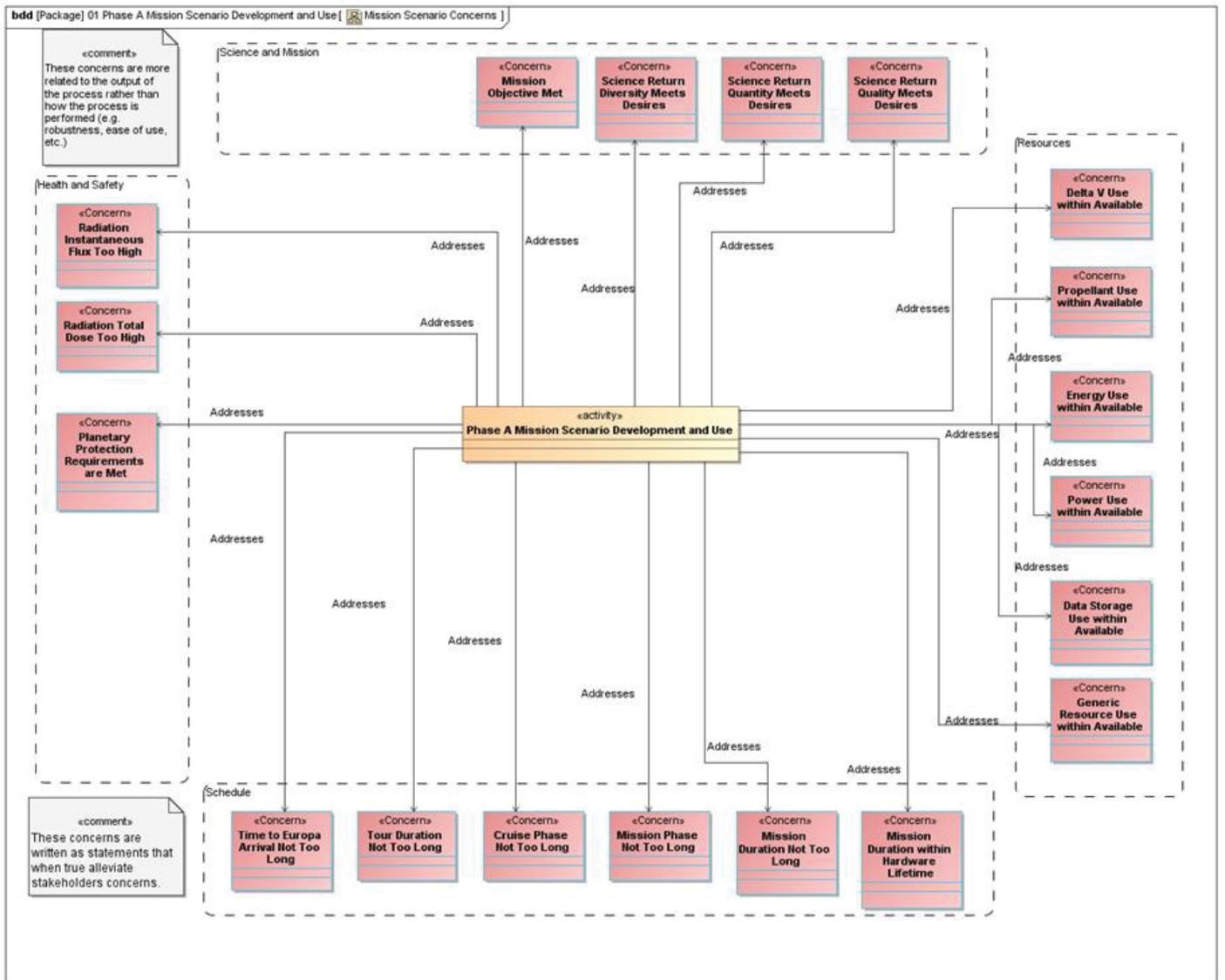


Figure 1 Phase A Mission Scenario Concerns

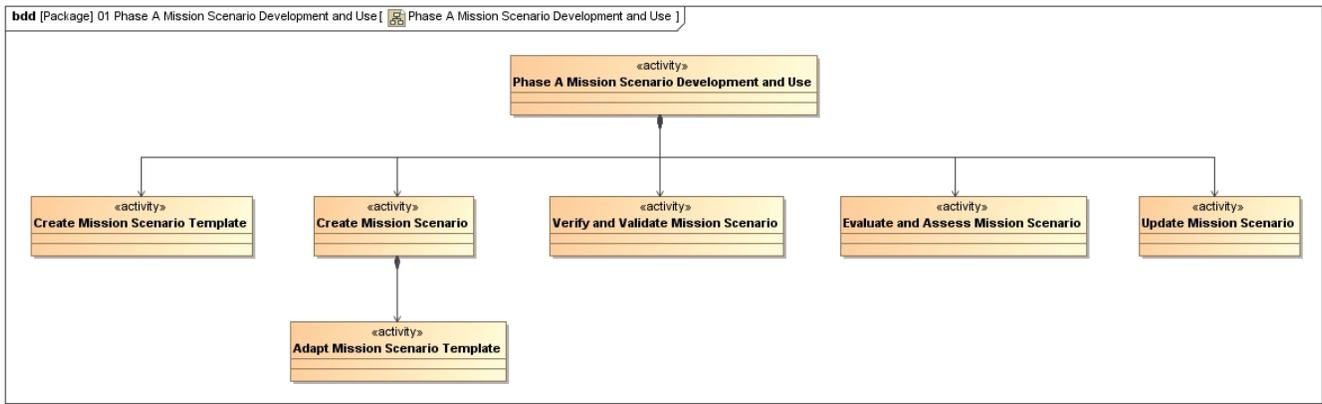


Figure 2 Phase A Mission Scenario Development and Use – Block Diagram

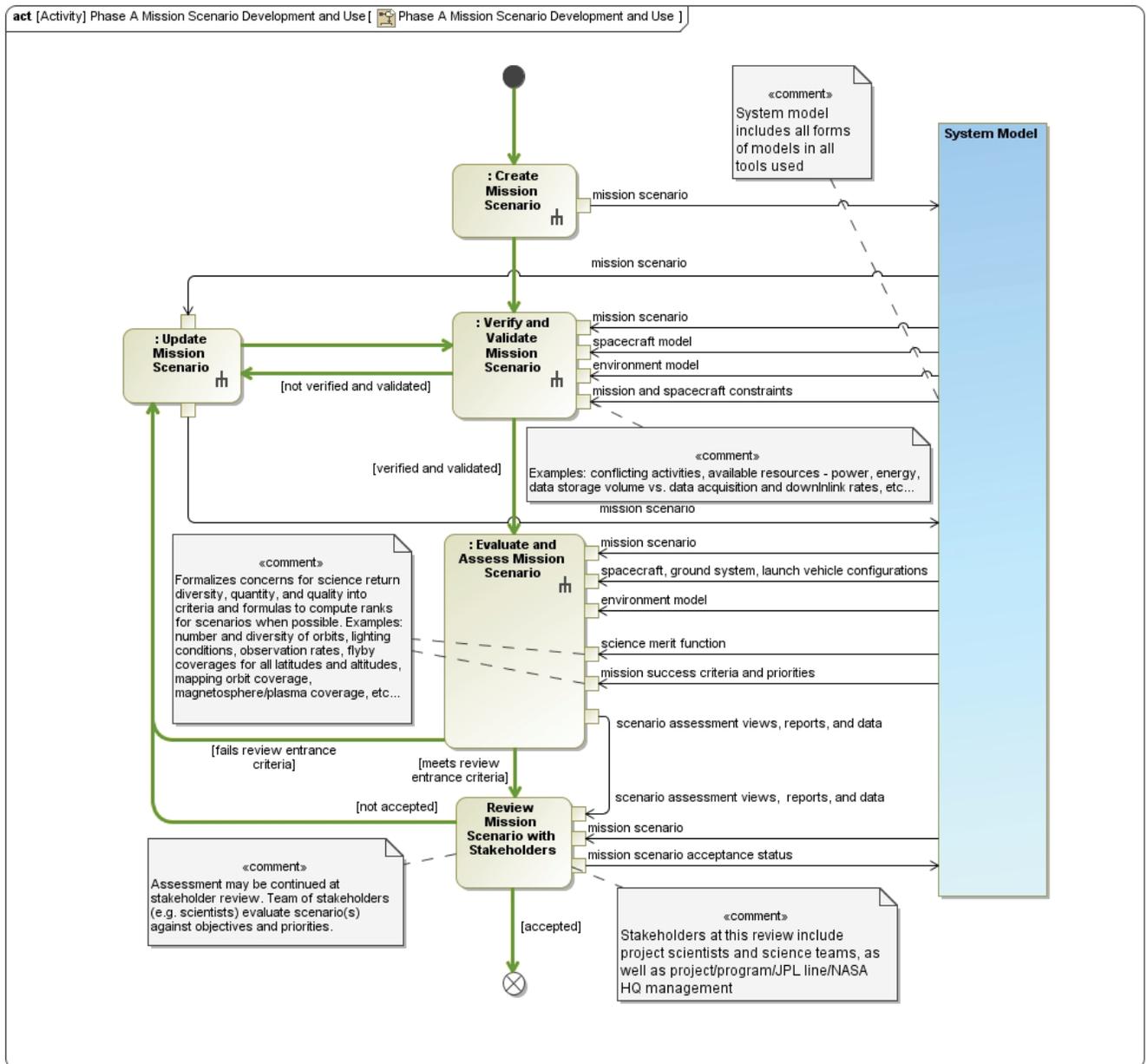


Figure 3 Phase A Mission Scenario Development and Use – Activity Diagram

A Mission Scenario (shown in Figure 4) is a description of what a mission does during operations and how the modeled elements of the system change over time. It is composed of timelines that describe what changes, and what the mission intends to do. A mission scenario has a start and end time. Each timeline models an element of the "system model". Each element is part of either a flight system, or ground system, or launch system, or environment, or the element may be a relationship between two parts in any of these

systems (i.e. the distance between the spacecraft and the Earth). Each scenario references a "configuration" that describes which version or option of a model element is selected for the scenario. For example, one scenario may have a solar power system, while another may be powered by radio-isotopic thermonuclear generators. For the mission scenario to be "valid", it must be consistent with all the constraints and requirements placed on these elements. Note that constraints and requirements are part of system model.

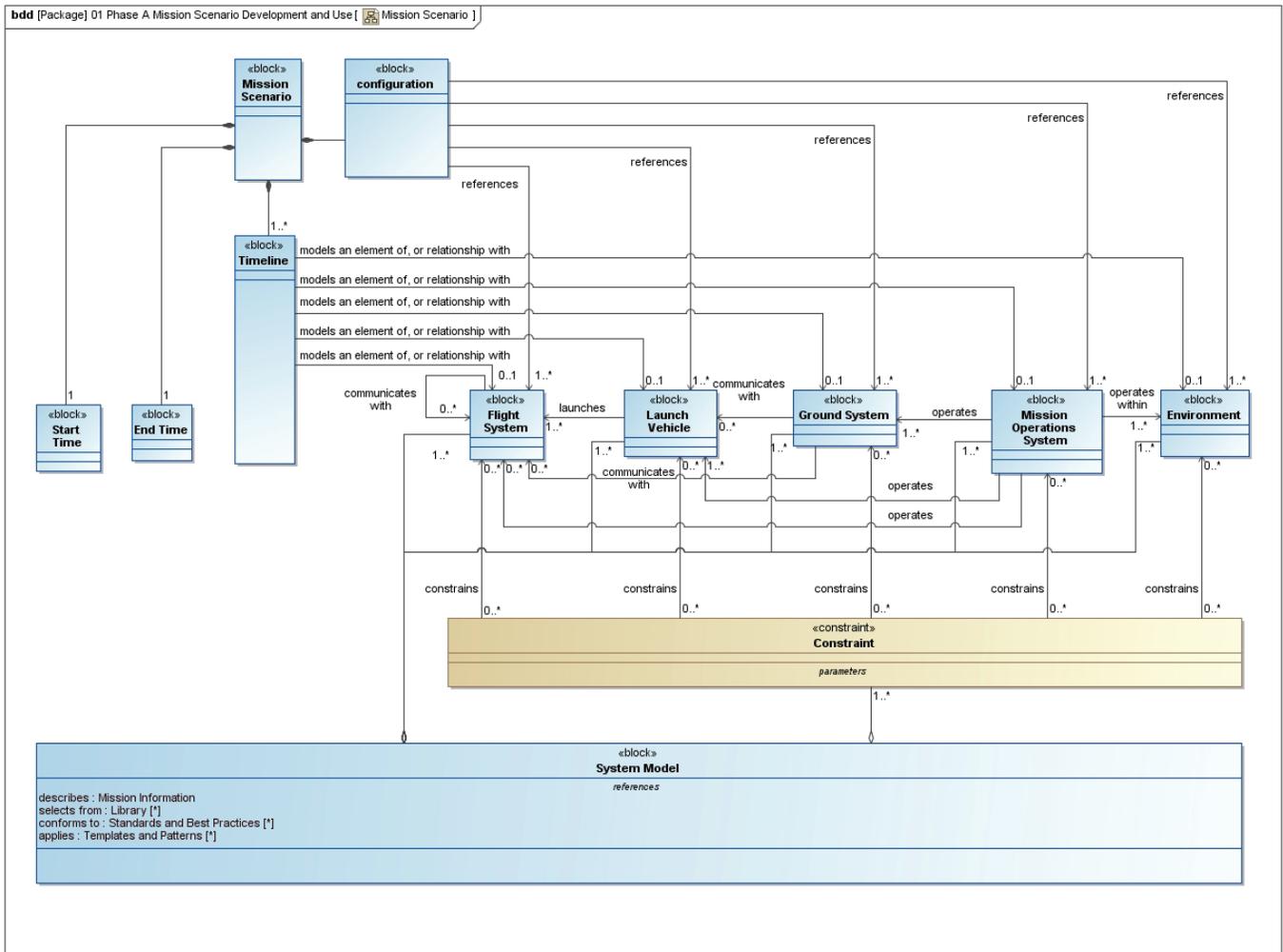


Figure 4 Mission Scenario Block Diagram

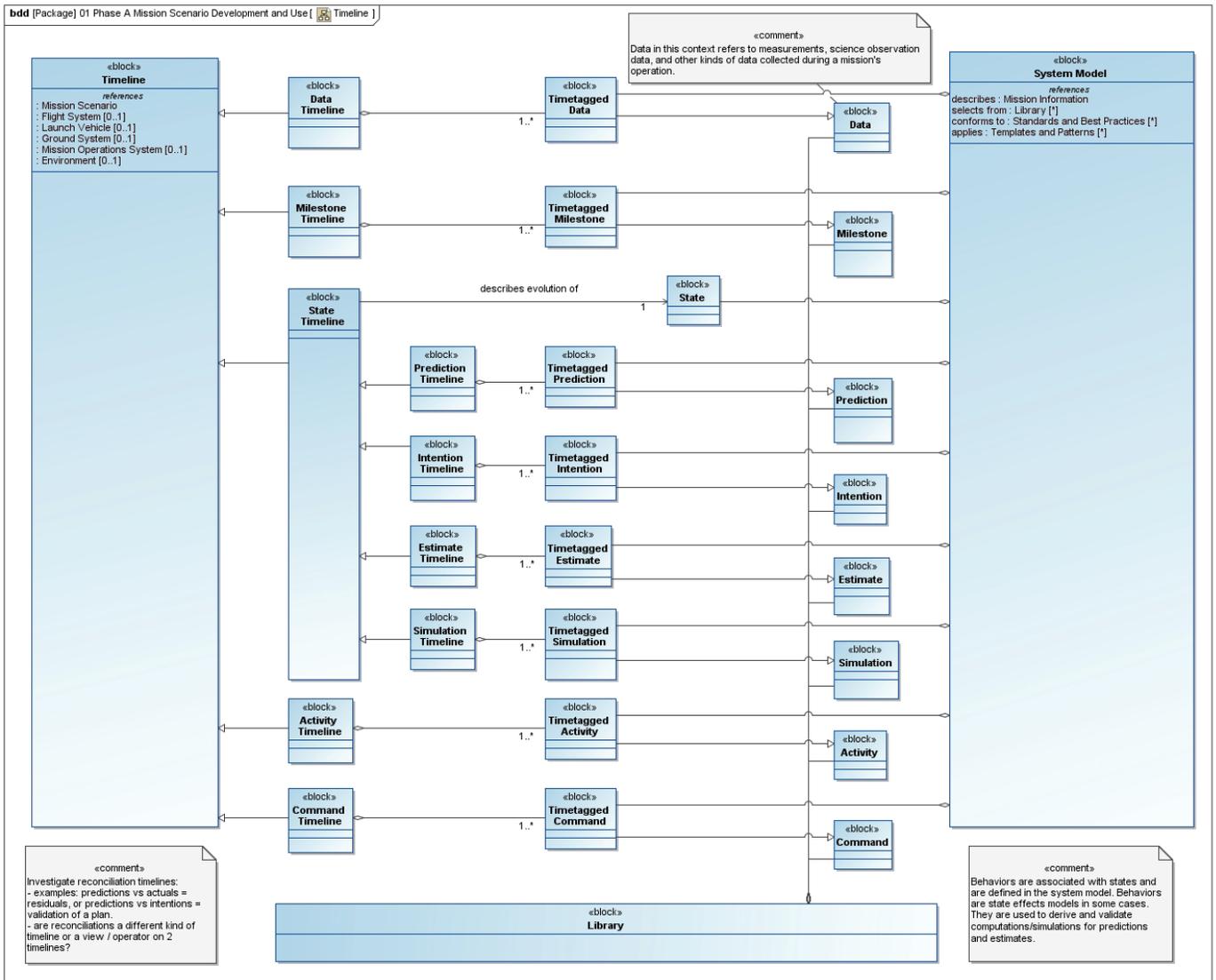


Figure 5 Timeline Block Diagram

A Timeline (as shown in Figure 5) is a series of sequential time tagged items. A Timeline may represent the evolution of a state for an element in the system model, or it may represent a series of events. The events may be commands in sequences, activities in plans and mission scenarios, or mission milestones. Each state may have several distinct Timelines. A particular Timeline for a state may represent a set of predicts, the intentions of the mission, estimates of what has actually happened in the past, or simulated "what if" test conditions or events.

The "Create Mission Scenario" scenario is shown in the activity diagram in Figure 6. A mission scenario needs to reference a particular version of the system model that contains elements of the flight and ground systems, and the environment. The scenario exercises these elements, and needs to be consistent with the kind of elements in the system model version. A scenario may be built using

templates in the Artifact Template Library. A template provides a standard description of a common kind of mission scenario. A template can be customized by mapping roles in the template to elements in the system model or by adding new activities, logic, or flows. Alternatively, a new template can be created, added to the library, and then used to create the scenario. Finally, a scenario may be created without the use of a template (not shown on diagram).

A mission scenario needs to reference a set of configurations of the system model to identify the controllable elements of the flight and ground system and set initial conditions. Configurations take into consideration model differences that represent changes through the lifetime of a mission. Examples include separation or docking events, launch vs. cruise vs. landed configurations, deployment events, consumable amounts like propellant mass or battery energy, environmental effects, and others.

The scenario also needs to reference a set of activity types that the mission is capable of performing. These are stored as part of the system model version, and are assumed to have been developed earlier.

From the set of activity types, a set of activity instances are created, and then prioritized. The priorities are selected to

reflect science objectives and critical engineering needs. These priorities are used to populate the mission scenario in accordance with desirements that are retrieved from the system model. Further definitions and constraints for state prediction, intention, and simulation of milestones, data, and command are also used to populate the mission scenario.

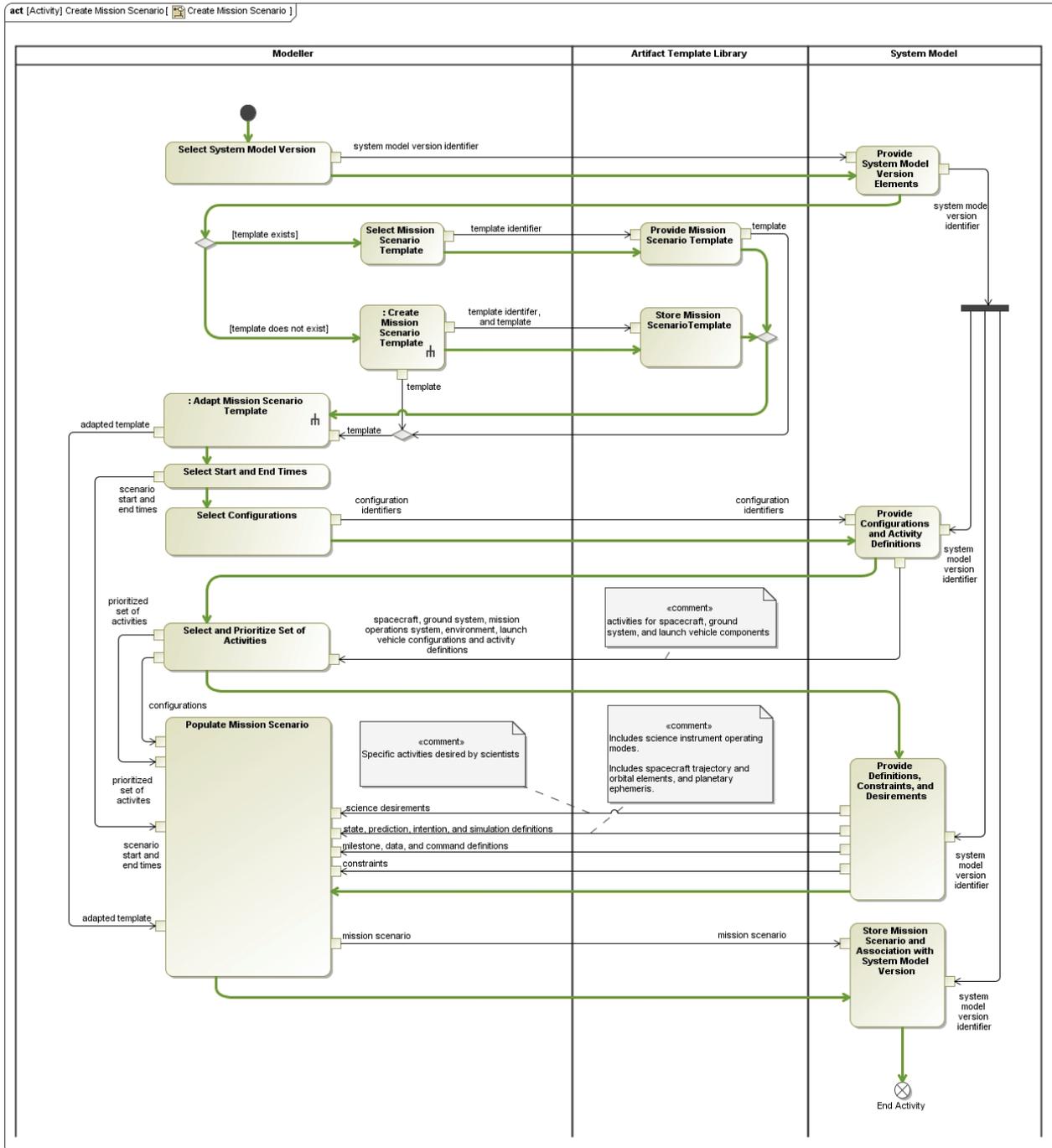


Figure 6 Create Mission Scenario Activity Diagram

The populated mission scenario is stored with an association to the referenced system model version in the system model for subsequent evaluation and inspection.

The "Evaluate and Assess Mission Scenario" scenario is shown in the activity diagram in Figure 7. The scenario begins with selecting a previously created mission scenario from the system model. Evaluation of the scenario may consist of time-based simulation (e.g. mission lifetime radiation modeling) or static analysis (e.g. a power use rack up). In either case, the result is the generation of data used for scenario assessment. In addition, the results of a simulation may be further analyzed to produce assessment data.

Viewpoints are selected or defined that describe the selection of data to be assessed and the form in which it is

presented. The data resulting from the simulation and/or analysis is processed into view instances that conform to the viewpoints selected. The views contain the data to be assessed. Example views include a science merit functions computation results, a table of total science data volume returned for each instrument, planetary body coverage maps, expected mission lifetime figures, tables of consumable usages (including, perhaps propellant), for example.

An assessment of the views is performed. This includes checks against requirements, constraints, science and mission objectives, and desirements. The results of the assessment are fed into a report generation process. Reports are provided to the project stakeholders for use in other activities such as trade studies, and project design validation.

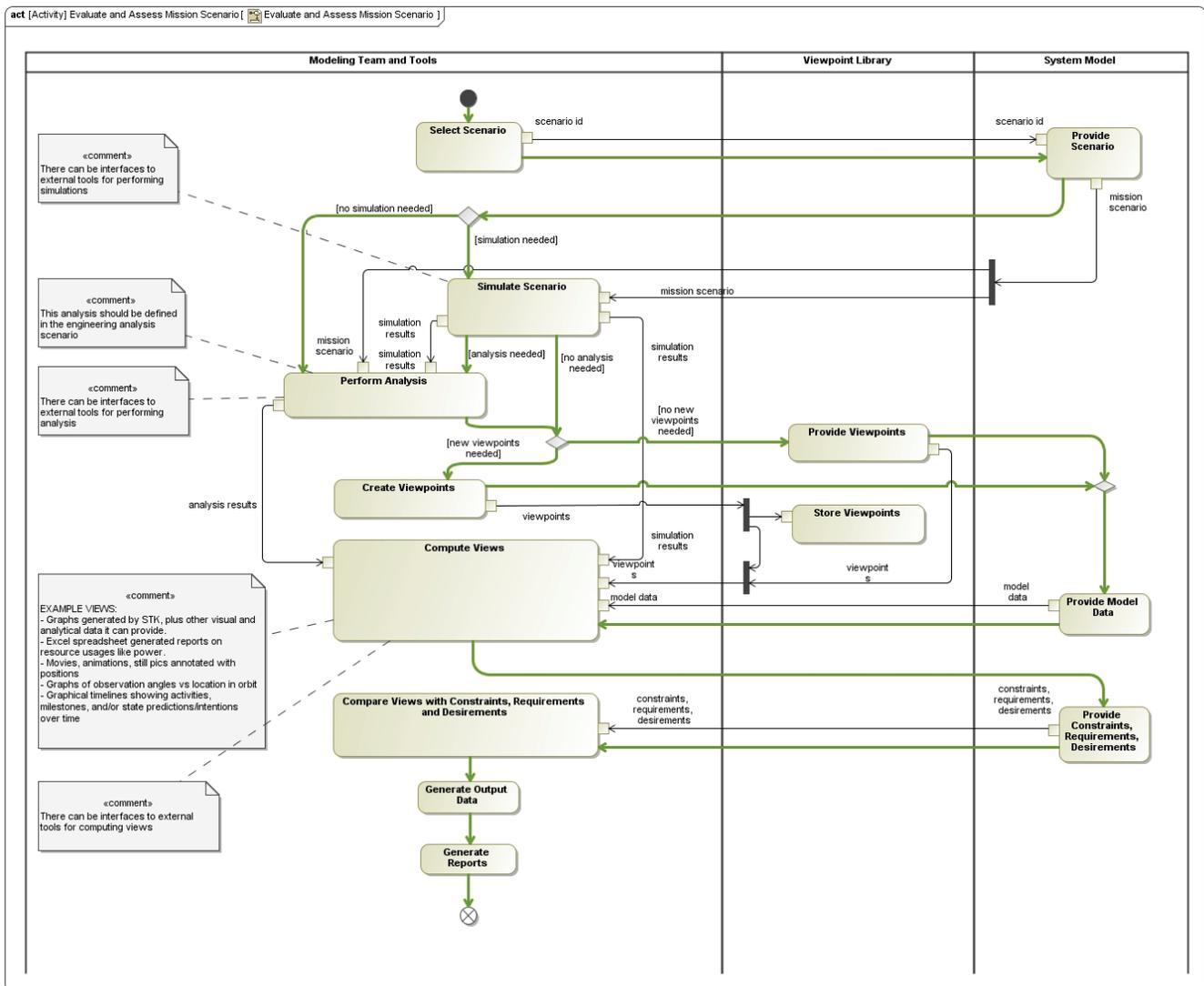


Figure 7 Evaluate and Assess Mission Scenario Activity Diagram

Advantages of Model-Centric Approach

- Model-based scenarios can be used throughout a project's lifecycle in the following ways. The scenarios can help to validate mission designs in phase A, as well as verify and validate subsequent project design decisions against models of the spacecraft, ground system, and environment. Scenarios can be refined into automated procedures for subsystem, system, launch operations, and in-flight operations readiness tests. Further, scenarios can be developed into sequences for operations. Application of scenarios developed early in phase A to later phases of a project using similar representations helps to maintain continuity and to accurately retain design information and assumptions throughout a project's lifecycle.
- Model-based scenarios can be used to evaluate mission and spacecraft design against science objectives, requirements, and constraints using models of expected behavior.
- Model-based scenarios allow for automation of a continuously integrated development process for project design elements that can include simulation of scenarios and evaluation against objectives, requirements, and constraints.
- Model-based scenarios can be used to automate analysis and report generation.

IMCE Capabilities Assumed in this Scenario

- Tools to represent timelines.
- Tools to simulate execution of timelines.
- Tools to evaluate the results from the simulated executed of timelines.
- Scenario evaluations must be able to call upon models of the spacecraft, ground system, and space environments.

These are all capabilities we currently have in one form or another. In some cases, they simply need to be re-packaged to operate in a more integrated fashion with models that can be used throughout a mission's lifecycle.

Future Work

- Develop concerns related to launch operations scenarios.
- Develop scenario for "Create Mission Scenario Template"
- Develop scenario for "Adapt Mission Scenario Template"
- Develop scenario for "Verify and Validate Mission Scenario"
- Develop scenario for "Update Mission Scenario"
- Investigate reconciliation timelines. Examples: predictions vs. actuals = residuals, or predictions vs. intentions = validation of a plan. Are reconciliations a different kind of timeline or views on 2 timelines?

5. EXAMPLE: CAPTURING THE DESIGN

The architect captures the design as architecture descriptions. The architecture descriptions are compliant with the JPL architecture framework.

The architecture is represented as a set of consistent models used to create different views as necessary to communicate them to the stakeholders. The architect chooses and/or develops a set of views that are meaningful to one or more stakeholders in the system. The choice of the particular views is a key decision.

The architecture/design descriptions are used to communicate with the stakeholders, and enable them to verify that the system will address their concerns.

The scenario can be described as follows:

- (1) Choose Architecture Framework. Refer to an existing set of views/viewpoints (architecture framework). Develop a mapping between the established architecture framework such as DoDAF [6] and RASDS [7], and possible JPL custom views/viewpoints. Use this mapping as a reference for further viewpoints/views definitions.
- (2) Select the appropriate viewpoints – based on the stakeholders and concerns that need to be covered by views)
 - Capture concerns, stakeholders, and possible views from the "Flight System Gate Products" developed by Neil Yarnell.[8]
 - Expand with concerns and views.
 - Develop a working document to facilitate further knowledge capture from more JPL key representatives for different systems engineering disciplines, projects, and line organizations. Consolidate the feedback in the document.

Architecture Description

According to the ANSI/IEEE 1471 [4] standard, the architecture is "the fundamental organization of a system embodied in its components, their relationships to each other and to the environment and the principles guiding its design and evolution".

To describe a system's architecture the architect produces a set of material evidence, or "products" that represent the future system, its component parts, how parts function, rules and constraints for functioning, how parts relate to each other and to the environment. Architecture descriptions may be partitioned using different perspectives (operational, functional, structural/configuration, etc.) that are meaningful for one or more stakeholders. These partitions are called views and represent the overall architecture as seen from a particular perspective, or viewpoint. The viewpoint is defined by the concerns expressed by the stakeholders. Stakeholders and Concerns define Viewpoints, from which

multiple Views of the architecture can be constructed. The union of these Views is an Architectural Description. In capturing or representing the design, the architect will typically create architecture models. A view will comprise a selection of model elements, chosen to demonstrate that a particular set of concerns are indeed addressed by the design.

The models provide an intentionally more coherent description of the system's architecture. Multiple views may be created, to enable the architecture to be communicated and understood by the stakeholders and enable them to verify that the system will address their concerns

Choosing an Architecture Framework

The development of the architecture can be guided by using an architecture framework. A framework can be defined as a set of approaches, components, configurations, models, services, standards and principles intended to guide the architecture development.

Existing architecture frameworks prescribe a given set of viewpoints/views, and provide guidance, rules and product specifications for developing and presenting architecture models. These frameworks are typically developed for a specific domain (military, enterprise, etc.), and are focused on some particular goals (acquisition of systems, development, etc), or a particular type of community. There is value in using more than one, and pieces can be added or customized on a "mix and match" basis.

The architect (or rather the organization) can choose to adopt, or adapt an existing architecture framework or to build a custom framework. It is not always necessary to start from zero, and the main focus needs to be on the concerns of the stakeholders.

Mapping JPL Design Descriptions to Architecture Frameworks

Since a "mix and match" solution, using an established architecture framework as a checklist seems to be a reasonable approach, the first step in our process of choosing the framework elements has been to map a list of JPL design descriptions (known also as gate products) to existing architecture frameworks. As a result, we determined DoDAF and RASDS to be the closest to the needs of JPL, in terms of technical content and user community. The JPL "Gate Products for the Flight Systems" [8] has been used as the starting list of products.

The main purposes of the mapping are given below:

- Evaluate the frameworks and determine what framework is more suitable
- Determine the ramifications of a "mix and match" solution

- Define a set of viewpoints that may draw from architectural description standards

In using [8], each product has been associated with concerns addressed by the product. The concerns have been captured from stakeholders, or from IMCE internal sources. The products have been additionally described so that they can be further elaborated, and reviewed by stakeholders, or other interested parties.

The Flight System Design Descriptions/Gate Products

Reference [8] recommends flight system design descriptions. The document identifies various "views" that can be used to describe the design, and cites examples for each view. The preferred flight system description includes the following views:

- (1) Physical. The physical view focuses on the mechanical aspects (like mechanical configuration(s) and packaging lay-outs) of the system.
- (2) Functional. The functional view concentrates on how the basic system functions (like pointing, communications, and time synchronization) are accomplished.
- (3) Operational. The operational view demonstrates that the system can do what is needed of it in each of the key mission phases (like launch and science operations).
- (4) Performance. The performance view relates to how well the system is predicted to do what is required of it.
- (5) Attributes. The attributes view describes system characteristics (like safety and reliability) not apparent in the physical view.
- (6) Behaviors. The behaviors view describes system features (like autonomy and operability) that relate to its operation in flight.
- (7) Interfaces. The interfaces view defines the boundaries between different design agencies in terms of the properties that are of interest to either or both parties to the interface.
- (8) Resources and Consumables. The resources and consumables view identifies the technical quantities that need to be managed during either the development or the operation of the system- to ensure the integrity of the system design.

Benefits noted

The main benefits of using an architecture framework for capturing the design are:

- Better understanding and communication.
- Focus on independent components for analytical purposes.
- Provide and maintain a disciplined method of depicting design.

- Speed up and simplify architecture development.

6. FUTURE WORK

Future work is in two categories: use of the Concept of Operations in the development of IMCE, and updates to the Concept of Operations itself.

The ConOps has already proven valuable to the IMCE effort, in several ways. Several of the ConOps team members are now working to infuse IMCE into the proposed JEO mission, and they have brought with them the insights garnered by the development of the ConOps. The contents of the ConOps, both in model and in report form, are proving useful in communicating IMCE concepts to new team members as well as project stakeholders. We expect that both of these areas will continue to provide significant benefits to IMCE and JEO. In addition, the ConOps has also proven useful in the SysML tool selection process now underway at JPL, by serving as a significant source of criteria which any tool must satisfy. Finally, the process of developing the ConOps has greatly enhanced our understanding of what capabilities flight projects will need from IMCE. This increased understanding has enabled us to formulate more concrete plans and to communicate them more effectively to our sponsors.

The ConOps was conceived as a living artifact. Because we developed the scenarios in a SysML model and produced the report directly and automatically from the same model, we now find it easy and natural to develop additional scenarios as needed. For example, we are developing a scenario recently identified as needed for JEO: how systems and subsystems engineers interact with each other and with the system model in the course of managing technical resources such as mass. In this way the suite of scenarios covered by the ConOps can grow as needs are identified. We are currently leaving open the question of whether and when to issue a formal update to the ConOps Report, depending on the needs of IMCE and of the user projects.

7. CONCLUSION

The ConOps has met all of the goals and objectives as described in Section 1. Furthermore, we have demonstrated the feasibility and the worth of modeling flight project development scenarios using the SysML language and of the automated generation of cogent and informative reports directly from these models. Most of the ConOps team members have now moved on and are directly supporting various flight projects. Several are engaged in system modeling in support of the JEO pre-project formulation, where they are directly applying the insights gained during the ConOps development. Although we are still in the early stages of IMCE infusion, the Concept of Operations for IMCE is already proving to be a crucial element in the understanding, acceptance, and application of model-centric principals at JPL.

ACKNOWLEDGEMENTS

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

REFERENCES

- [1] "An Operations Concept for Integrated Model-Centric Engineering at JPL", T. Bayer, L. Cooney, C. Delp, C. Dutenhoffer, R. Gostelow, M. Ingham, J. S. Jenkins, B. Smith, 2010, IEEEAC Paper #1120
- [2] "Concept of Operations: Integrated Model-Centric Engineering", 29 July 2010, T. Bayer, M. Bennett, L. Cooney, C. Delp, C. Dutenhoffer, D. Dvorak, R. Gostelow, M. Ingham, M. Jackson, S. Jenkins, S. Mandutianu, R. Rasmussen, internal report.
- [3] "The Need for an Integrated Model-Centric Engineering Environment at JPL", IOM 3100-09-040, 24 Oct 2009, T. Bayer, internal memorandum.
- [4] "Systems and software engineering – Recommended practice for architectural description of software-intensive systems," ISO/IEC 42010:2007. Also IEEE Std 1471–2000, "Recommended Practice for Architectural Description of Software-intensive Systems", <http://www.iso-architecture.org/ieee-1471>
- [5] "Dynamic Gate Product and Artifact Generation from System Models", M. Jackson, C. Delp, D. Bindschadler, M. Sarrel, R. Wollaeger, D. Lam, 2011, IEEEAC Paper #1509
- [6] DoDAF - Department of Defense Architecture Framework V2.0 cio-nii.defense.gov/sites/dodaf20/
- [7] RASDS – Reference Architecture for Space Data Systems, CCSDS, public.ccsds.org/publications/archive/311x0m1.pdf
- [8] "Gate Products for Flight Systems," Neil Yarnell, JPL internal document, 2010

BIOGRAPHIES



Todd J. Bayer is a Principal Engineer in the Systems and Software Division at the Jet Propulsion Laboratory. He is currently the Flight System Engineer for the proposed Jupiter Europa Orbiter mission and the Assistant Manager for Flight

Projects of JPL's Systems and Software Division. He received his B.S. in Physics in 1984 from the Massachusetts Institute of Technology. He started his career as a project officer in the US Air Force at Space Division in El Segundo, California. Following his military service, he joined the staff of JPL in 1989. He has participated in the development and operations of several missions including Mars Observer, Cassini, Deep Space 1, and Mars Reconnaissance Orbiter, for which he was the Flight System Engineer for development and Chief Engineer during flight operations. During a leave of absence from JPL, he worked as a systems engineer on the European next generation weather satellite at EUMETSAT in Darmstadt, Germany.



Daniel Dvorak is a Principal Engineer in the Software and Systems Division at the Jet Propulsion Laboratory, currently supporting JPL's Integrated Model-Centric Engineering Initiative and leading NASA's Software Architecture Review Board. His

interests include model-based systems engineering, goal-driven operations, fault management, and software architecture. Dr. Dvorak holds a Ph.D. in computer science from The University of Texas at Austin, a M.S. in computer engineering from Stanford University, and a B.S. in electrical engineering from Rose-Hulman Institute of Technology.



Matthew Bennett is a Senior Software Systems Engineer in the Flight Software & Data Systems section at the Jet Propulsion Laboratory. His research interests include model based engineering, software architecture, fault protection, and spacecraft autonomy. He has developed mission software for fault

protection, guidance and control, science data collection, performance analysis, and simulation. He holds an MS from the University of Washington in Computer Science, and a BS from the University of California at San Diego in Computer Engineering.



Steven Jenkins is a Principal Engineer in the Systems and Software Division at the Jet Propulsion Laboratory, currently supporting JPL's Integrated Model-Centric Engineering Initiative. His interests include application of semantic and modeling technologies to systems engineering. Dr. Jenkins holds a B.S. in Mathematics from Millsaps

College, an M.S. in Applied Mathematics from Southern Methodist University, and a Ph.D. in Electrical Engineering (Control Systems) from UCLA.



Christopher Delp is the Systems Architect for the Operations Revitalization task in the Multimission Ground Systems and Services Program Office at JPL. He is also a member of the Flight Software Systems Engineering and Architectures Group at JPL. His interests include software and systems architecture, applications of model-based systems engineering, and real-time

embedded software engineering. He earned his M.S. and B.S. degrees from the University of Arizona in Systems Engineering.



Sanda Mandutianu is a senior member of the technical staff at Jet Propulsion Laboratory. She has been task lead and PI on systems and software architectures, autonomy and control, information architecture, artificial intelligence, agent-based and semantic technologies. She led a model-based pilot for the early phase JPL flagship mission Europa

Explorer, and is currently participating in JPL and NASA systems engineering model-based initiatives. Sanda is interested in how the innovative and conceptual aspects of her work can effectively apply. She published peer reviewed papers and book chapters. She holds an MS in Physics from the University of Bucharest, Romania.