# The OpenSE Cookbook: A practical, recipe based collection of patterns, procedures, and best practices for executable systems engineering for the Thirty Meter Telescope

Robert Karban*[a], Amanda G. Crawford[a], Gelys Trancho[b], Michele Zamparelli[c], Sebastian Herzig[a], Ivan Gomes[a], Marie Piette[a], Eric Brower[a]
[a]Jet Propulsion Laboratory, California Institute of Technology,
4800 Oak Grove Drive, Pasadena, CA, USA 91109;
[b]Thirty Meter Telescope, 100 West Walnut Street, Pasadena, CA, USA 91124;
[c]European Southern Observatory, Karl/Schwarzschild-Str 2,
Garching b. Muenchen, Germany 85748
*robert.karban@jpl.nasa.gov
© 2018. All rights reserved

## ABSTRACT

The OpenSE Cookbook is an open-sourced collection of patterns, procedures, and best practices targeted for systems engineers who seek guidance on applying model-based and executable systems engineering (MBSE) using SysML. Its content has emerged from the system level modeling effort on the European Framework Program 6 (FP6) and the Thirty Meter Telescope (TMT). The TMT MBSE approach applied the Executable Systems Engineering Method (ESEM) and the open-source Engineering Environment (OpenMBEE) to specify, analyze, and verify requirements of TMT's Alignment and Phasing System (APS) and the Narrow Field Infrared Adaptive Optics System (NFIRAOS). In these applications, implicit dependencies are made explicit in a formal model through the use of ESEM, OpenMBEE, and SysML modeling constructs. The value proposition for applying this MBSE approach was to establish precise requirements and fine-grained traceability to system designs, and to verify key requirements beginning early in development. The integration of ESEM and the OpenMBEE tooling infrastructure (providing linked-data and web-operability) is a significant added value for the MBSE approach. The APS is responsible for the overall pre-adaptive optics wavefront quality, using starlight to measure wavefront errors and align the TMT optics. In the formally integrated and executable SysML model, simulations are performed to analyze the impact of changed requirements and verify specified constraints for various operational scenarios.

The APS team used several modeling patterns to capture information such as the requirements, the operational scenarios, involved subsystems and their interaction points, the estimated or required time durations, and the mass and power consumption. Adaptive optics systems are designed to sense real-time atmospheric turbulence and correct the telescope's optical beam to remove its effect. The system model for the adaptive optics operational modes was developed to capture sequence behaviors and operational scenarios to run Monte-Carlo simulations for verifying acquisition time, observing efficiency, and operational behavior requirements. The model is particularly useful for investigating the effect of parallelization, identifying interface issues, and re-ordering sequence acquisition tasks. A former version of the Cookbook (which is now updated to MBSE challenges, goals, and lessons learned) included modeling guidelines and conventions for all system aspects, hierarchy levels, and views, which were developed during for the Active Phasing Experiment (APE),

an opto-mechatronical system technology demonstrator for the Extremely Large Telescope (ELT). The Cookbook utilizes the above mentioned system models as real-world case-studies to demonstrate and document the applications of the recipes, providing also instructional examples and addressing the available tooling support. The Cookbook is accompanied by a number of SysML models and aodel libraries which facilitate model authoring and maintenance. The Cookbook covers the different aspects of Systems Engineering such as management of Requirements, Design (behavior and structure), Interfaces, Interdisciplinary Integration, Analysis, Trade Studies, and Technical Resources. This paper presents the background, motivation, architecture, and highlights some key content of the Cookbook. For example, interface management, error budget management, requirements verification, Monte Carlo driven analysis, and timing analysis of operational scenarios. The paper discusses how the capabilities of OpenMBEE contributed significantly to the adoption of executable systems engineering.

**Keywords:** Model-Based Systems Engineering, Executable Models, SysML, Engineering Environment, Telescope

## 1. INTRODUCTION

The Thirty Meter Telescope (TMT) [9], under development by the TMT International Observatory (TIO), applies a “hybrid” systems engineering approach leveraging traditional systems engineering and more modern model based techniques to perform the systems engineering tasks such as requirements management, technical resource management, interface and design management and analysis.

**Figure 1 The Thirty Meter Telescope**

Traditional systems engineering requires clear, defined deliverables, easily accessible processes, a shallow learning curve, and simple traceability among artifacts.

Model based techniques significantly enhance the understanding of the behaviors of a system, and provide rich capabilities to represent complex systems. Model Based Systems Engineering (MBSE) emphasizes rigor and precision helping to manage the complexity of the system and its design. It supports the integration horizontally across the system life cycle, and vertically, across multiple domains.

The main objective of MBSE for the TMT is to capture operational scenarios and demonstrate that requirements are satisfied by the design. For this purpose, a SysML [7] model is created to better understand and communicate system behavior, motivated by optimizing the system design.

The SysML model is executable (i.e. it is built for simulation for analysis purposes) to capture requirements, use cases, system decomposition and subsystem relationships. The resulting system design is analyzed against power, mass, duration and error budget requirements, and used to produce engineering documents such as detailed design documents, interface control documents. In order to enhance the communication of the design, standard languages (e.g. SysML) and techniques are used.

The Jet Propulsion Laboratory (JPL) participates in the design and development of several subsystems of TMT and delivers the complete APS. The TIO is the customer, which provides the APS requirements to JPL, and JPL delivers an operational system to the TIO. The APS team pursues an MBSE approach to analyze the requirements, come up with an architecture design and eventually an implementation.

The TMT project captures the functional and physical architecture, behavior, requirements, and parametric relationships for TMT NFIRAOS LGS MCAO Acquisition Sequence and related use case scenarios at system level to verify timing requirements in the early life-cycle phase through system-level simulation. This is achieved by performing Monte Carlo simulations for acquisition and slew time.

In the course of applying model based techniques, a number of modeling pattern, recipes, and best practices have been identified that are collected and described in the OpenSE Cookbook which is the topic of this paper.

The application of the OpenSE Cookbook practices allows to deliver consistently engineered products using a well-defined modeling approach, called the Executable Systems Engineering Method (ESEM) [4] that is a refinement of Object Oriented Systems Engineering Method (OOSEM), introducing the next phase of system modeling emphasizing executable models to enhance understanding, precision, and verification of requirements.

OOSEM provides an integrated framework that combines object-oriented techniques, a model-based design approach and traditional top-down systems engineering practices. OOSEM is a scenario-driven process coupling top-down decomposition with bottom-up design. It provides guidance on building a system model to analyze, specify, design, and verify the system. An example application of OOSEM is also described in detail in chapter 17 of [14].

ESEM augments OOSEM activities by enabling executable SysML models which verify requirements fulfillment. It includes a set of analysis patterns that are specified by means of various SysML structural, behavioral and parametric diagrams.

## 2. COOKBOOK PRINCIPLES

The OpenSE Cookbook provides a consistent, comprehensive, detailed and background-agnostic set of operational procedures to guide practitioners through MBSE. Unlike existing SysML literature whose goal is to provide the foundations of descriptive modeling to new comers and bring forward the arguments in favor of MBSE, the cookbook merely represents an implementation of what such literature often refers to as best-practices, or organization (or project)-specific procedures. Prerequisites for the Cookbook are SysML

knowledge, and experience with some MBSE method such as OOSEM. It provides goal oriented guidance for systems engineers explained by a set of combinable patterns, e.g. how to verify requirements or roll-up technical resources. Systems engineering workflows drive each of the pattern definitions (e.g. analysis, see Fig 2).

As shown in Figure 2, a TMT requirement change propagates to the configuration managed SysML model, where it is formalized into so-called property-based requirements (2), which allow for a formalized trace of requirements into the design. A property-based requirement captures the quantifiable textual content (e.g. values, constraints) of a requirement as distinct SysML model elements, in this case properties that enable formal evaluation of said properties. The as-specified conceptual design (3) and/or the realization design (4) are verified against the changed requirement, providing immediate feedback of the change impact to the design, resulting in a pass or fail. The design information and associated analysis reports are delivered in various engineering documents (5).

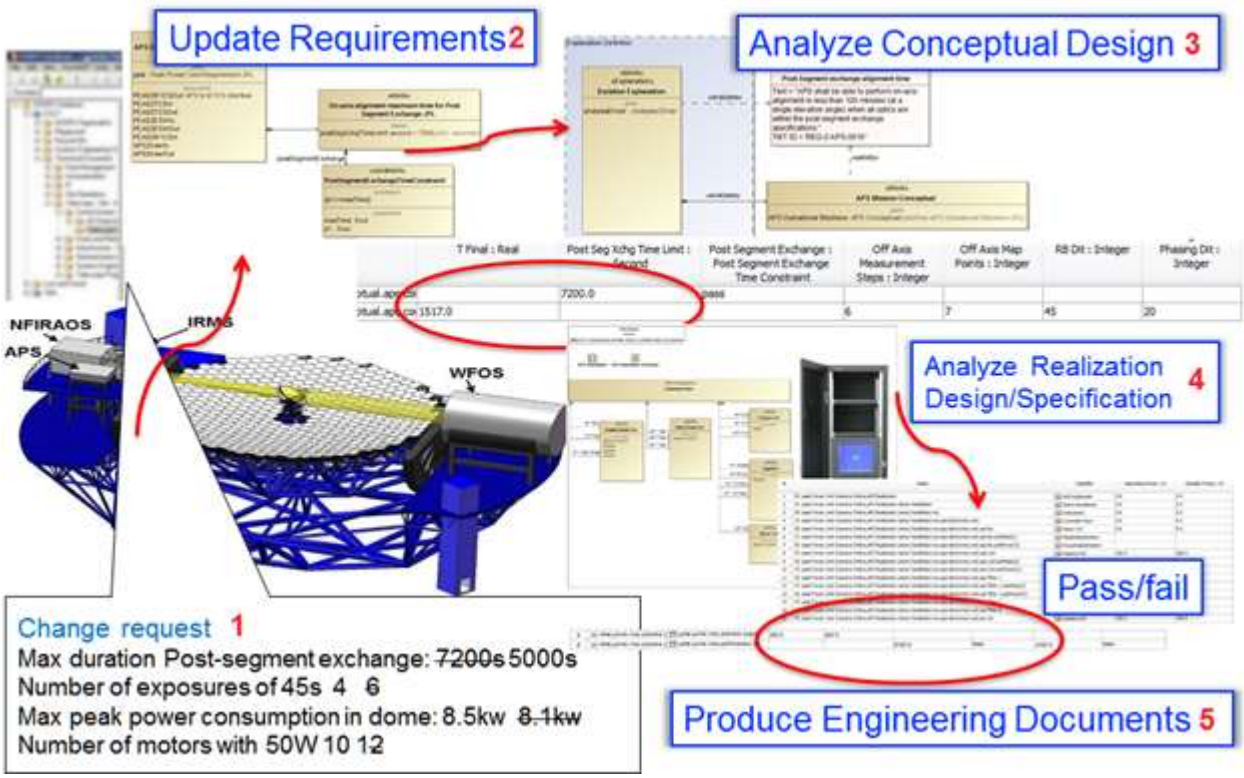


**Figure 2 TMT analysis workflow**

The OpenSE Cookbook demonstrates how to build and analyze system models using OpenMBEE [8] as applied to educational examples as well as actual usages in the TMT production model. The overall goal of the OpenSE Cookbook in conjunction with OpenMBEE is to commoditize the Executable Systems Engineering Method, i.e. remove the cost and barriers to entry that allows for expanded innovation and broader operations driven by increased user access and decreased costs, in order to foster broadest adoption.

The methodology, best practices and patterns are supported by a rich set of tooling, aggregated in a so-called systems environment, see Fig 3. The systems environment supports an orchestrated pipeline for systems engineering tasks and products. Engineers use a multitude of tools which are integrated through the systems

environment infrastructure allowing ubiquitous data access, transparent to the individual tools users, and avoiding peer-to-peer integrations. Note, that not all the shown tools are required but they are merely brought to the “engineering table” and integrated according to the engineer’s needs. The tools are a mix of commercial, open source, and in-house built. Each of the mentioned tools serves one or more roles in the above pipeline. Different engineers use (sometimes slightly) different tools for the same or different purpose and yet want to integrate with other tools and propagate or relate data from multiple heterogeneous sources. The inner part of the “table” enables this integration and propagation for the user and avoiding costly peer-to-peer integrations.

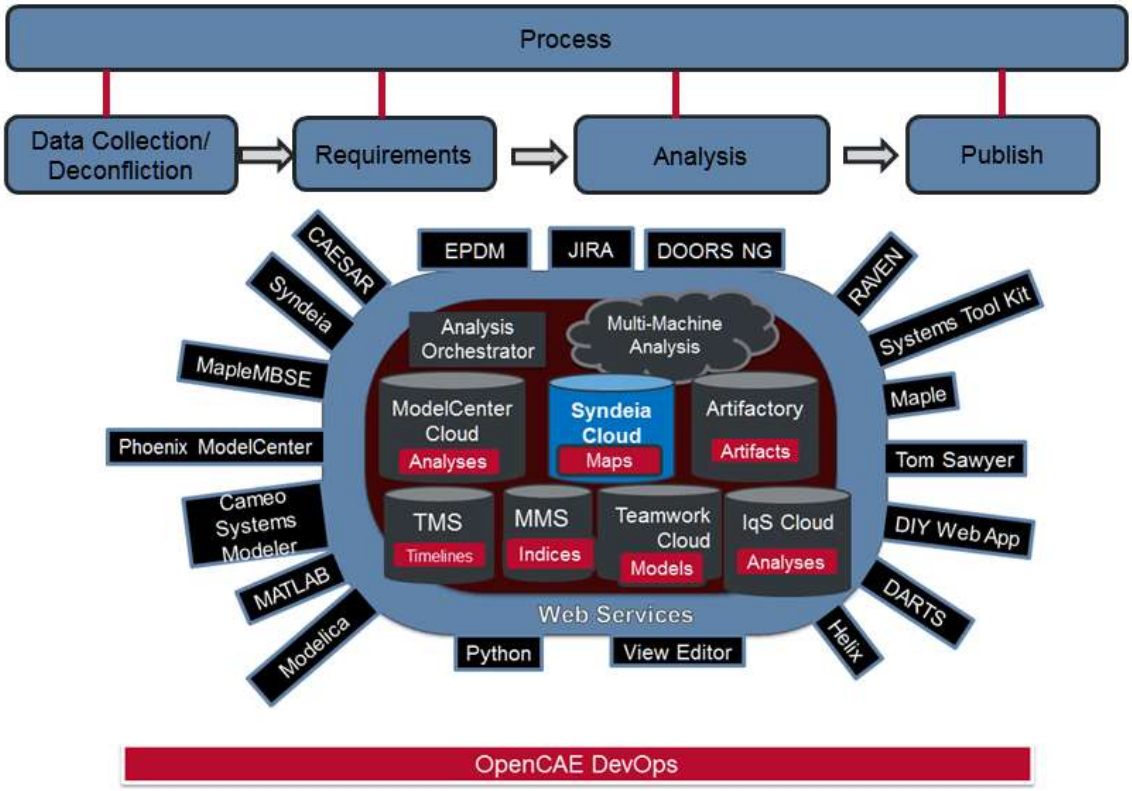


**Figure 3 JPL Systems Environment for the entire development lifecycle**

The systems environment is continuously evolved through so-called Case Studies which capture engineering workflows and uses cases, analyzing them, and providing the appropriate support within the environment.

Those Case Studies address for example the following areas:

- Requirements Management
- Interface Management
- Design Management
- Trade Studies
- Interdisciplinary Integration
- Analysis Management

- Resource Management
- Timeline Management

Each Case Study has a corresponding volume in the OpenSE Cookbook that contains best practices, patterns and procedures to support the identified engineering workflows and use cases.

The following table shows a subset of the patterns which are captured in the Cookbook, that were informed by the TMT effort.

Table 1. Subset of Cookbook Patterns

| **Pattern** | **Intent** | **Volume** |
|---|---|---|
| Customer/Supplier | Capture Customer Supplier specifications and their relationship | Requirements Management |
| Black Box Specification | Specify a system as a black box | Analysis Management |
| Property Based Requirements | Link requirements management and system design | Requirements Management |
| Requirements Verification | Validate requirements, verify as designed system against requirements and publish analysis results | Requirements Management |
| Duration Analysis | Analyse timing and duration constraints | Analysis Management |
| Static Roll-up | Roll-up static technical resources | Resource Management |
| Dynamic Roll-up | Roll-up dynamic technical resources depending on operational modes | Resource Management |
| Monte Carlo Analysis | Perform Monte Carlo Analysis of technical resources | Analysis Management |
| Time Modeling | Capture Timing and duration constraints | Resource Management |

| | | |
|---|---|---|
| Quantities Units Dimensions and Values | Specify quantities and units for technical resources | Resource Management |
| Error Budget | Manage an error budget of a technical resource such as wave front error | Resource Management |

The patterns are captured in a standard format as described below.

Table 2. Standard format for Cookbook Patterns

| **View** | **Description** |
|---|---|
| Intent | A brief description of the problem which is addressed by the design pattern. The intent should prove useful when searching through design patterns, and to quickly understand the purpose of the pattern. |
| Motivation | Explains a representative problem of a broad class of problems that the pattern seeks to address. The representative problem is a widespread concern and not trivial. In addition, the Motivation provides conditions that must be satisfied in order for the pattern to be used. After the conditions have been satisfied, the goals that the designer is trying to fulfill can be met. Any complicating design aspects and design constraints will be mentioned as well. |
| Concept | A description of the structure of the pattern using SysML diagrams. All elements are generalizations of those that appear in the specific example given in the Motivation view. Each element of the structure is described, and a description of their responsibilities, purpose, and important relationships/interactions among participants are provided. |
| Consequences | A description of the results, side effects, and trade-offs caused when applying the design pattern to the modeler's system. The actions and the positive or negative consequences are described. Additionally, a listing of possible conflicts that can |

| | |
|---|---|
| | occur while applying or using the pattern are provided. The possible scenario is described, and an explanation of the conflict is provided. It should be understood to the reader of the design patterns that both positive and negative consequences could occur. The designs are a compromise between available options and no design is completely devoid of negative consequence. |
| Implementation | A sample model for instructional purposes that addresses the problem which is presented in the Motivation view. The example illustrates in detail how the pattern is applied to a particular problem. |
| Known Uses | Other efforts that have dealt with the design problem which is addressed by the design pattern. An example system may have used a variation or the same design pattern as a solution. |
| Tooling | Tooling information that can aid the implementation of the pattern is provided. Note, the information provided is meant only to aid and provide a faster method for implementing an aspect of the pattern. Tooling support is provided in several ways. If a specific software or plugin is recommended, information and links regarding such will be provided. Assuming the reader will proceed with the suggested tooling, screenshots of the expected input and output will be shown. |
| Related Patterns | Other patterns from the Cookbook that could be used in combination with the selected pattern. The similarities and differences between the patterns serves to provide guidance as to which pattern may prove more useful. |

A previous version of the cookbook, produced by the INCOSE MBSE initiative [22] in 2012, which focused on structure and requirements, has been integrated. It was created by reverse engineering of the European FP7 Active Phasing Experiment (APE) project as a case study. Whereas the 2012 version focused on demonstration of the application of SysML, the 2018 version focuses on the systems engineering concerns, guided by the ESEM, and capturing behavioral aspects of system design and analysis. It is accompanied by a model library which provides commonly used elements to facilitate the authoring of the system model. In addition to the TMT

and APE examples, the 2018 version contains a variety of instructional examples, showing the how systems engineering tasks are accomplished.

The application to an actual engineering team on TMT has proven its practical value in production and ironed out the practices and patterns described in the Cookbook. Template models and recommended model organizations provide an additional value in getting started with Executable Systems Engineering.

The OpenSE Cookbook is layered in different application domains which promotes re-use and enhances the general practices by institutional and project specific adaptations.

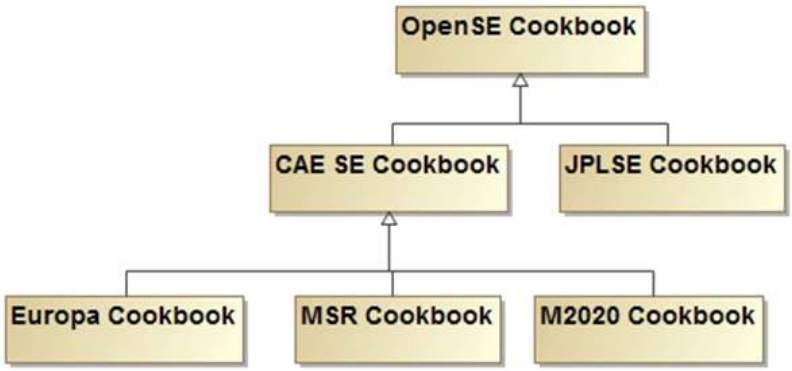


**Figure 4 OpenSE Cookbook Hierarchy**

Fig 4 shows the hierarchy established at JPL. The generic OpenSE Cookbook is elaborated by institutional patterns and practices addressing JPL specific tool services (CAE SE Cookbook) and JPL specific systems engineering practices (JPL SE Cookbook). Additionally, project specific adaptations (e.g. Europa Clipper, Europa Lander) are captures in project specific cookbooks.

## 3. RUNNING EXAMPLES

For brevity only the APS is shown in more detail to provide context for the patterns described below.

Following the ESEM, first the APS mission boundaries are defined, by identifying the interfaces and interchange of APS with other subsystems within the TMT. This is achieved using SysML Block Definition (BDD) and Internal Block (IBD) diagram types as can be seen in Fig 5.

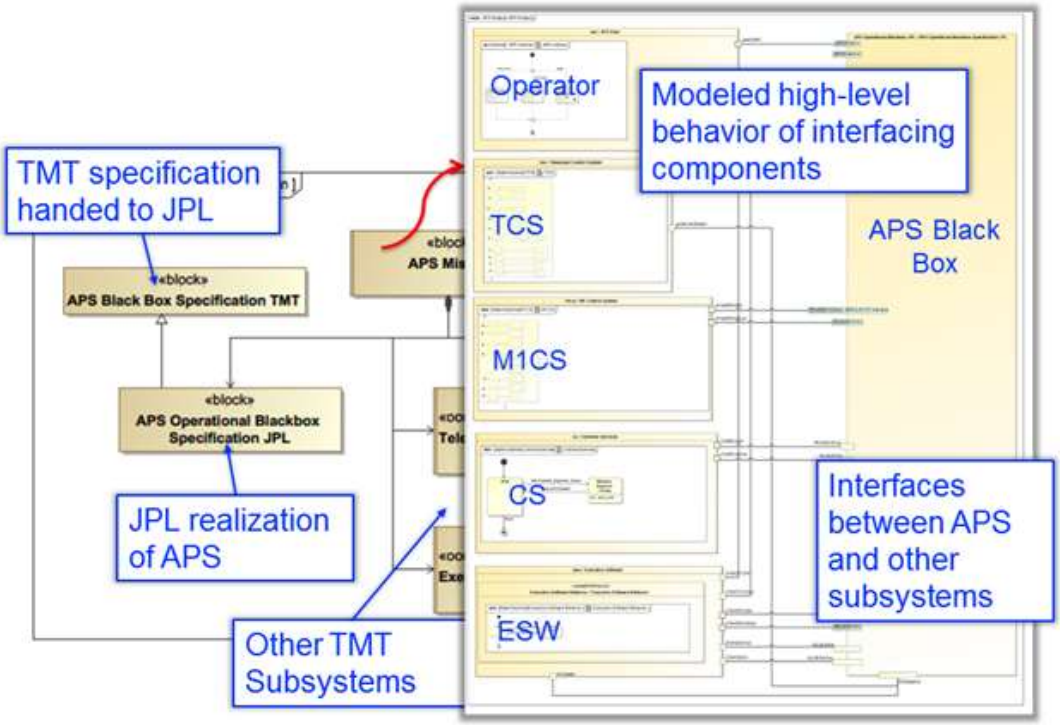


**Figure 5 TMT Block Diagrams**

Then, the conceptual architecture is elaborated to identify the functional components of the APS itself, again using BDD and IBDs, see Fig 6

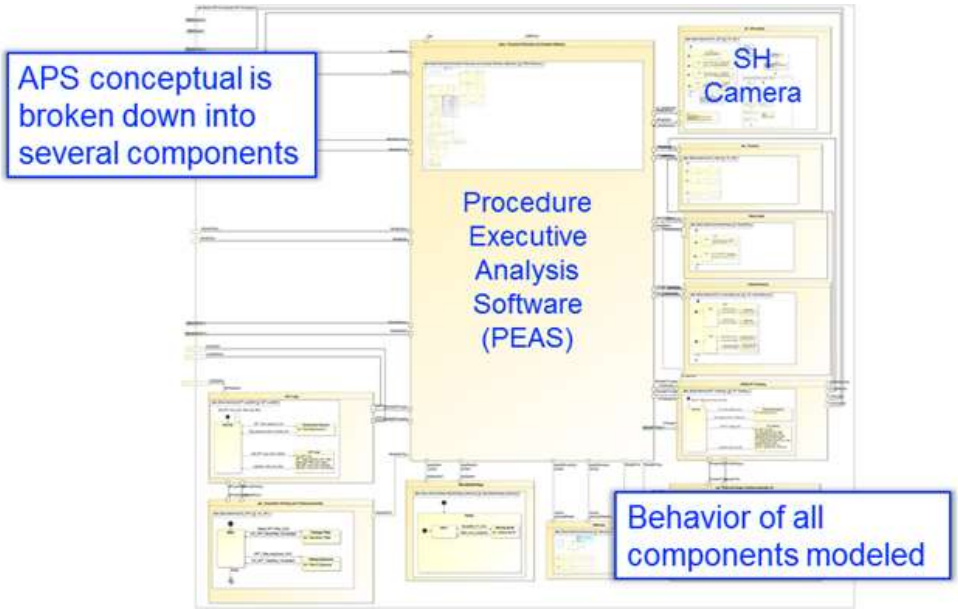


**Figure 6 APS Block Diagrams**

## 4. SELECTED COOKBOOK CONTENT

A selection of patterns from the Cookbook are described here in an abridged format. The unabridged content can be found at https://mms.openmbee.org

**Requirements Verification Pattern**

### Intent

The intent of this pattern is to validate requirements, verify as designed system against requirements and publish analysis results.

### Concept

The pattern follows the OOSEM structure, specifying the system of interest within the context of an enterprise. The specification is then elaborated into the three OOSEM white box designs, Conceptual Design, Conceptual Node Design, and Physical Design, see Fig 7. Additionally, property based requirement are used to specify requirements on the system of interest. For further analysis of one of the white box designs, the enterprise is specialized into a Context block, see Fig 8. In this example, the Context's system specification property is redefined by the Conceptual Node Design that allows interactions between the white box design components and the external systems in the enterprise context to be shown.

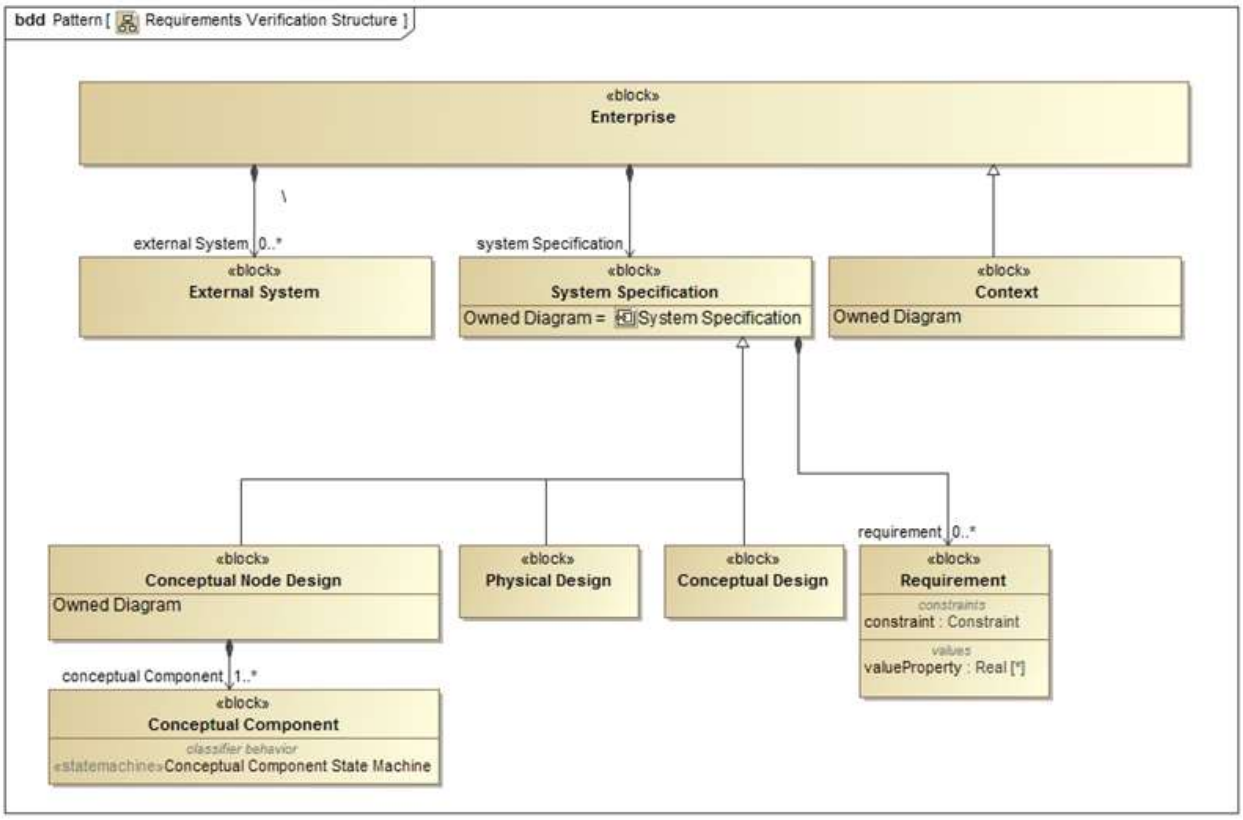


**Figure 7 Enterprise Context**

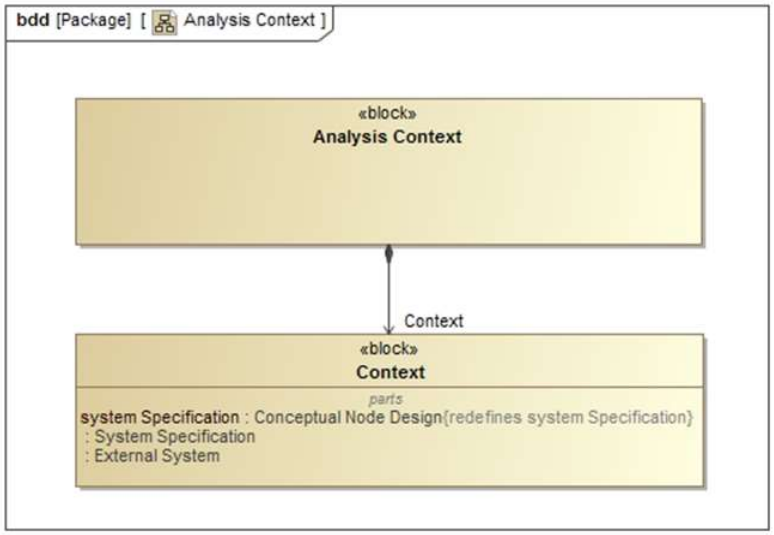


**Figure 8 Analysis Context**

## Known Uses

The as specified system of APS is analyzed to see if it meets the timing requirement of the post-segment exchange operational scenario.

First the textual requirement is formalized into a property-based requirement, see Fig 9.

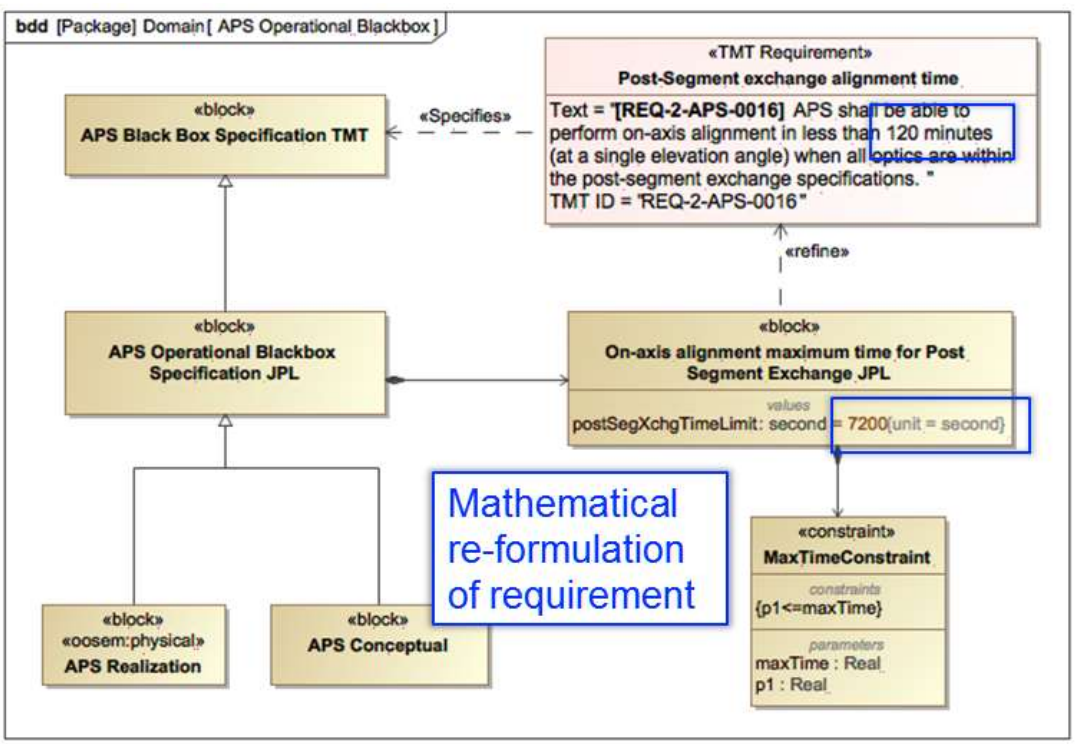


**Figure 9 Property-based Requirements**

The analysis context aggregates the conceptual model, binds the formalized requirements to the properties of the system design, in this case the total duration of an operational scenario, see Fig 10.

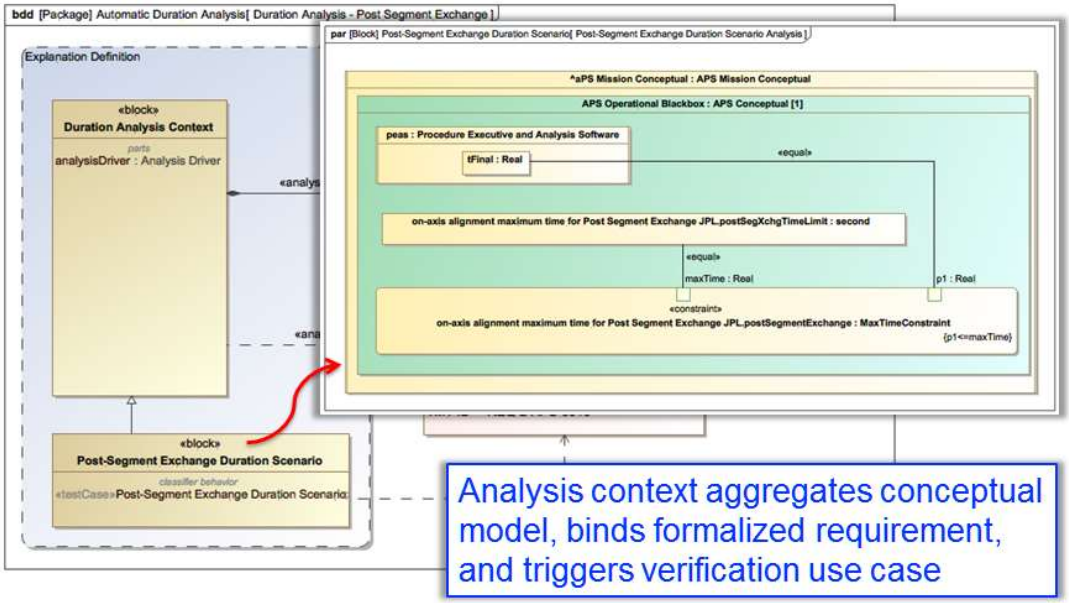


**Figure 10 Binding Parameter in Analysis Context**

Then, the operational scenario is initiated through a sequence diagram by passing a message, see Fig 11

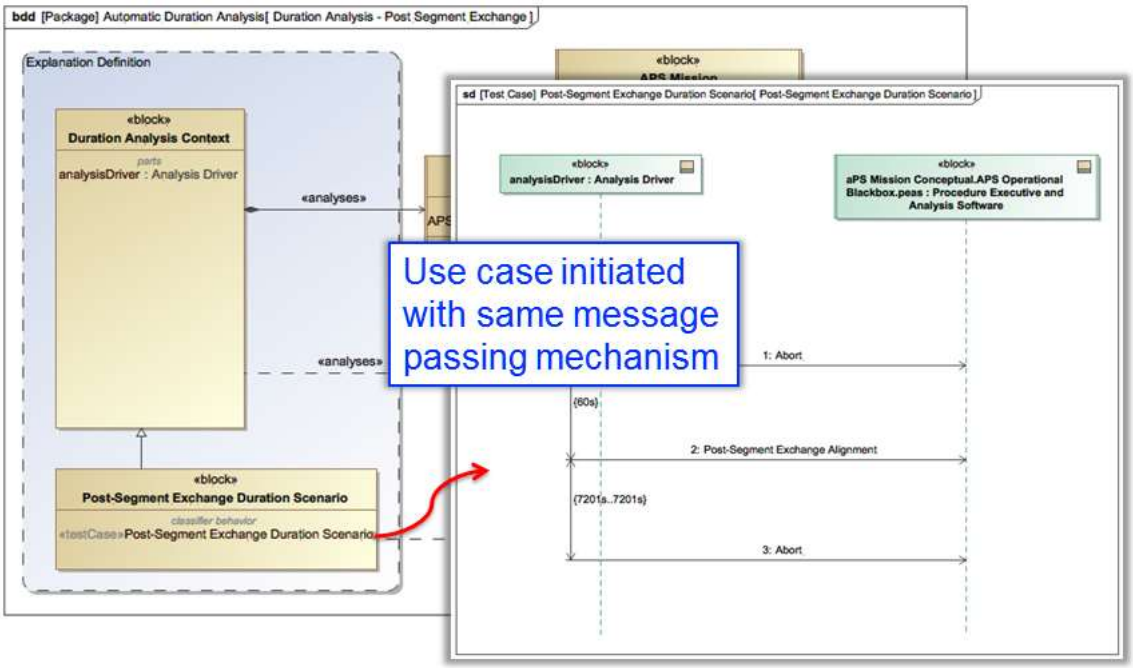


**Figure 11 Analysis Scenario**

Finally, a table with the results is produced showing if the constraint associated with the requirement is violated or not see Fig 12. In this example, the requirement is 7200 seconds and the as-specified system performs the operation in 4804 seconds.

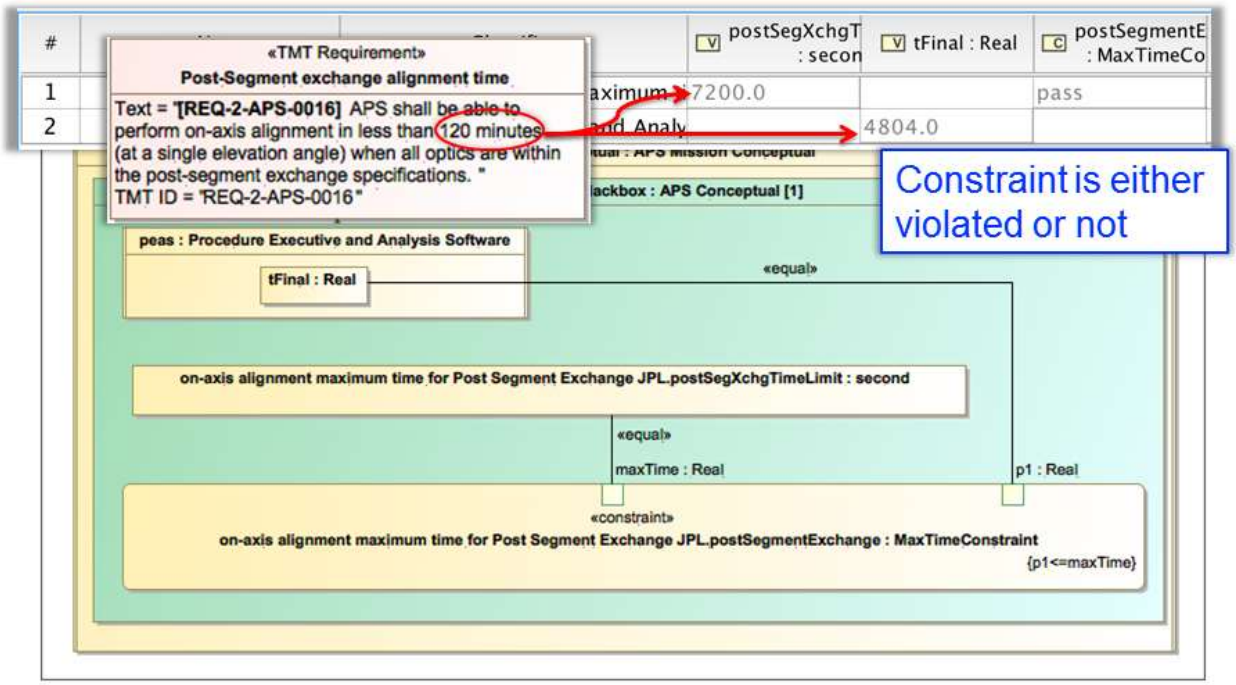


**Figure 12 Requirements Verification**

# Tooling

Cameo Systems Modeler and Simulation Toolkit are used to capture the model and perform the system level behavior simulation. View Editor is used to publish the analysis report.

Fig 13 shows a screenshot of the educational example of the autonomous ferry's internal structure while the simulation is run where different states are indicated by the elements highlighted in green, yellow, and red. Red for the currently active element, yellow for the last visited element, and green for the ones which have been active during the simulation.

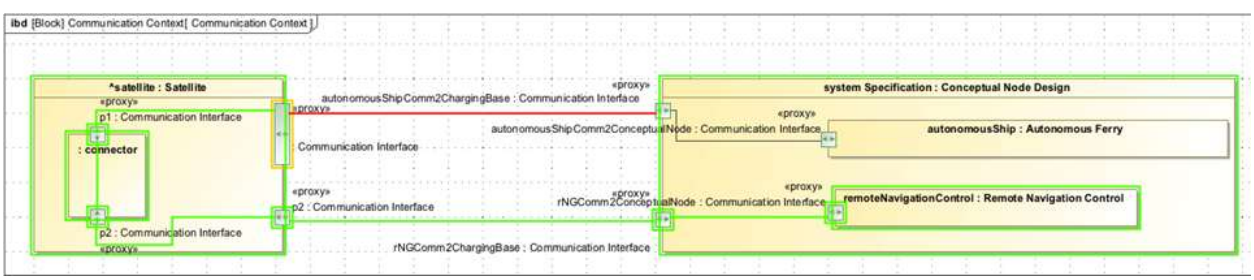


**Figure 13 Internal Block Diagram**

Fig. 14 shows a user interface produced with the simulation environment, plotting several state variables of the as-designed system, the battery state of charge, the data volume and the operational states of the ferry.

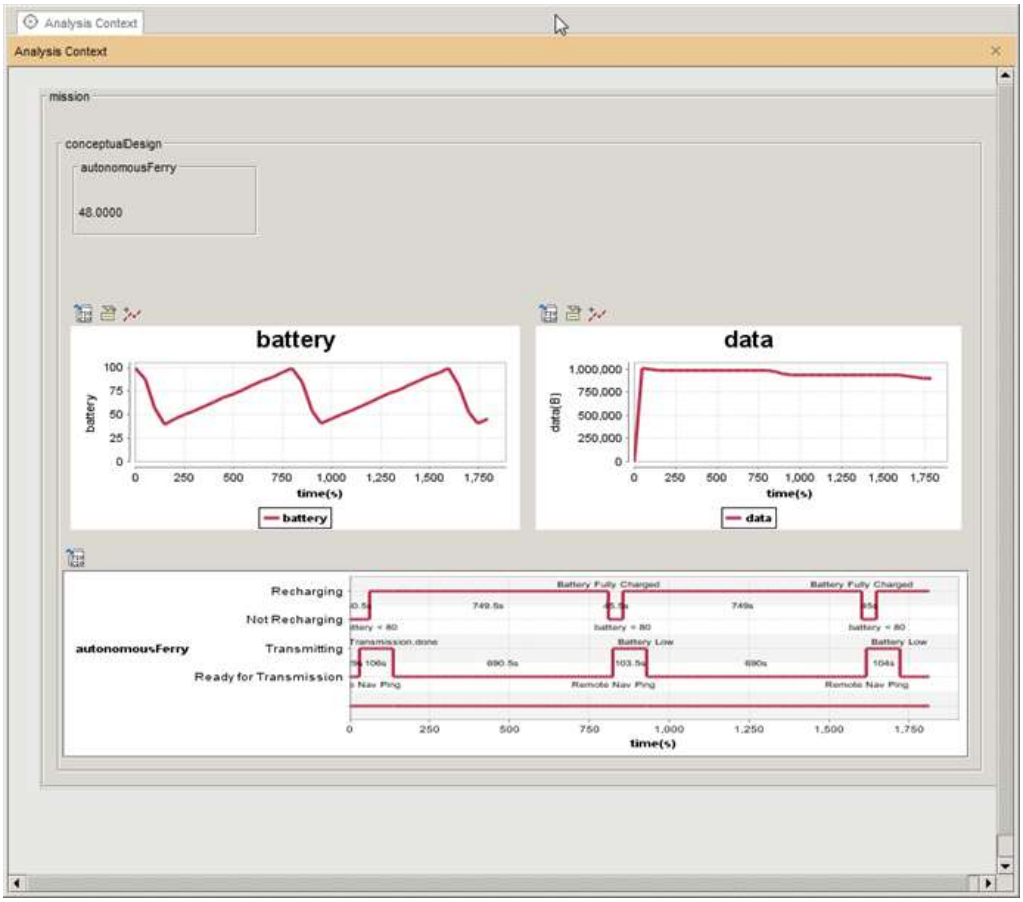


**Figure 14 Plot of System State Variables**

**Monte Carlo Analysis Pattern**

## Intent

The intent is to estimate the characteristics and probability of a particular behavior.

## Concept

The properties to be estimated can be expressed in SysML as either attributes to activities or as properties of the structural elements (typically blocks) whose classifier behavior the activity represents. Such an approach is in fact not restricted to Activities but may also be applied to other behavioral elements (conditional to tool support), see Fig 15

Probability at the edges of decision nodes can be modeled as value properties of systems.

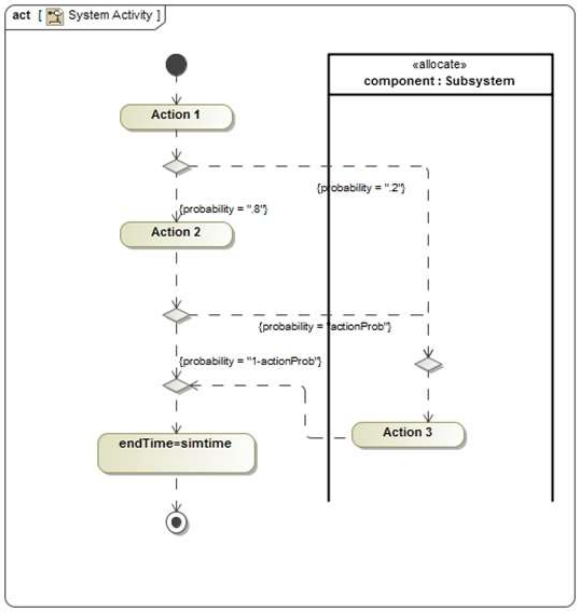


**Figure 15 System Behavior**

## Implementation

A Quadrupedal Robot is used as an educational example. The Monte Carlo simulation is combined with the Requirements verification pattern in order to verify the as-specified system against the requirement. The behavior of the robot shall be performed within 17 seconds, see Fig 16.

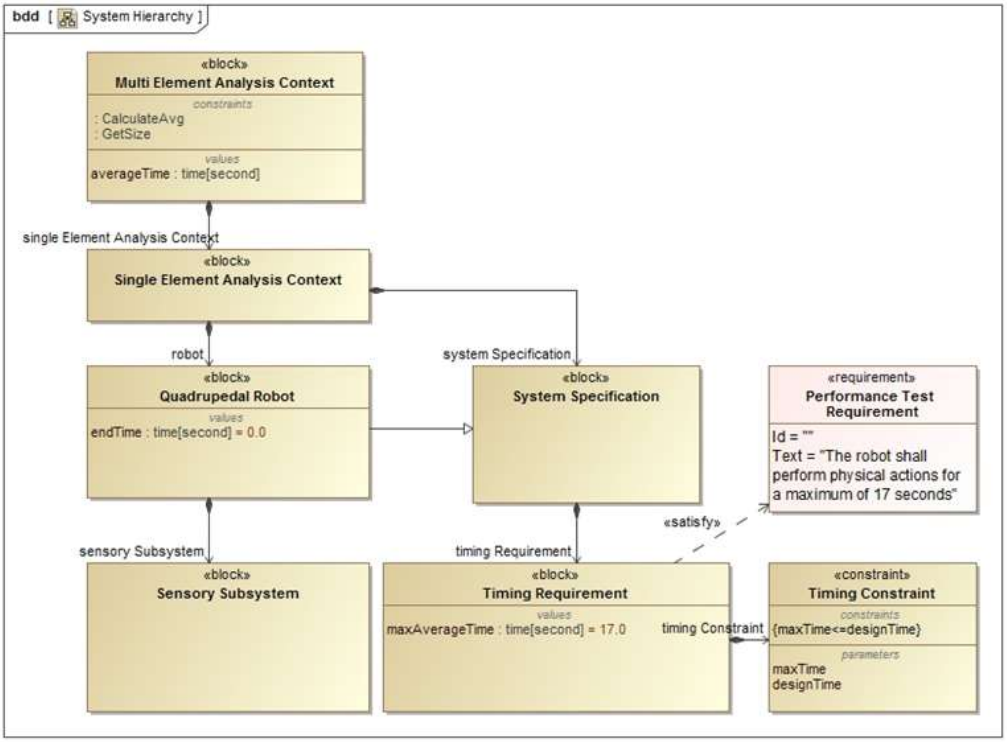


**Figure 16 Analysis Context**

The behavior of the robot is specified with an activity model and duration constraints on the actions. Each decision node in the activity flow has a certain probability to be taken, as shown in Fig 17.

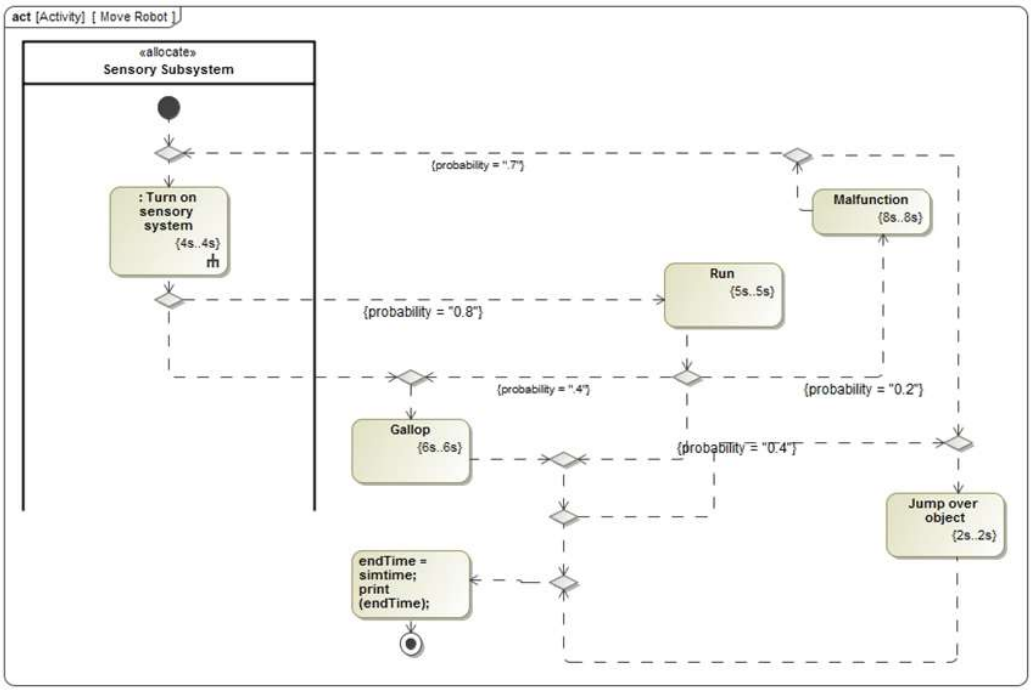


**Figure 17 Behavior of Robot**

Performing multiple runs of the system's behavior results in a table showing different times. In Fig 18. Resuls from five different runs are shown.

| # | Name | endTime : time[second] |
|---|---|---|
| 1 | robot at 2017.07.20 17.38 | 10.0 |
| 2 | robot at 2017.07.20 17.38 | 12.0 |
| 3 | robot at 2017.07.20 17.38 | 15.0 |
| 4 | robot at 2017.07.20 17.38 | 12.0 |
| 5 | robot at 2017.07.20 17.38 | 17.0 |

**Figure 18 Timing Results**

Those runs are then combined using a parametric model to verify the requirement against the as-designed system.

## Known Uses

In the TMT project this method is used to capture and analyze the Acquisition Process with IRIS and NFIRAOS LGS MCAO mode, shown in Fig 19.

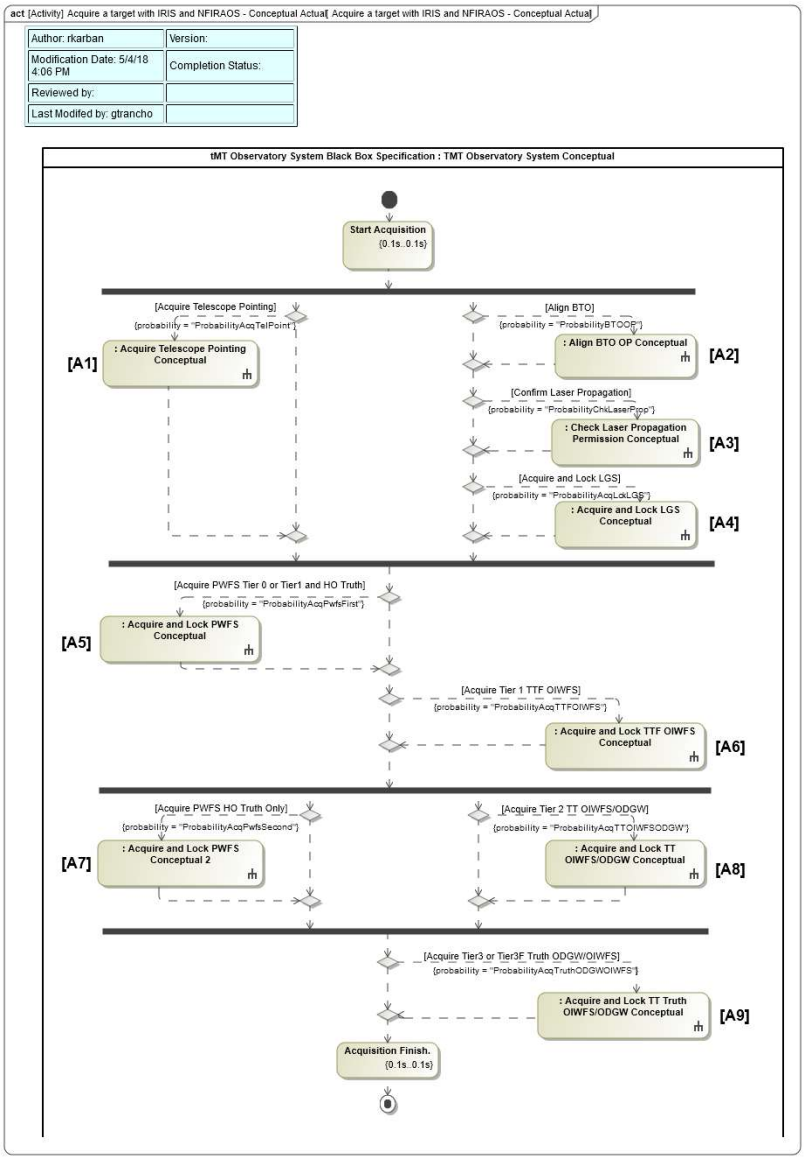


**Figure 19 IRIS and NFIRAOS LGS MCAO operational Acquisition Scenario**

## Tooling

Cameo Systems Modeler is used to perform the simulation runs, which also produce a trace of the actions taken for each run.

SysML probability concepts and distributed properties capture operational knowledge in system model.

Probabilistic distributions can be simulated within Cameo Systems Modeler using standard SysML probability distributions defined for value properties.

**Error Budget Management Pattern**

## Intent

The intent is to manage error budgets of technical resources such as wave front error.

## Concept

Conceptually, the errors are rolled up recursively in the system using different formulas, such as Sum, RootSumSquared, or Product, as shown in Fig 19 and Fig 20. Additionally, a margin is calculated and verified against a requirement.

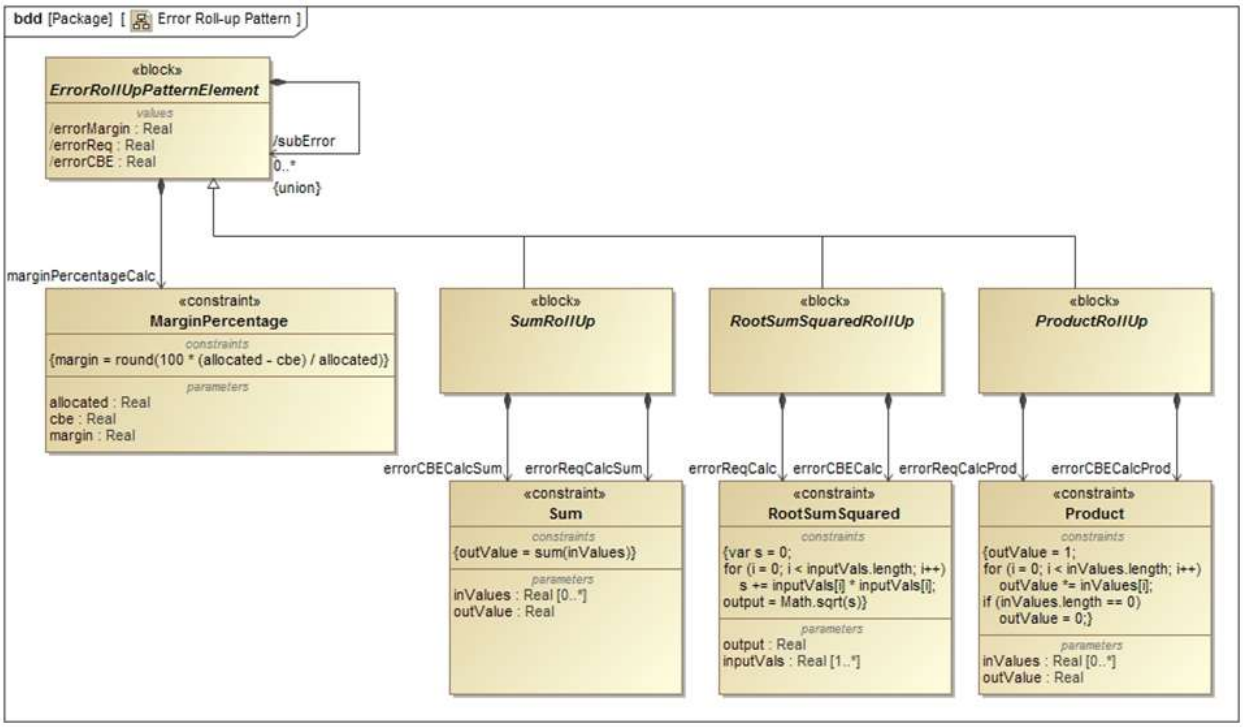


**Figure 20 Different types of Roll-ups**

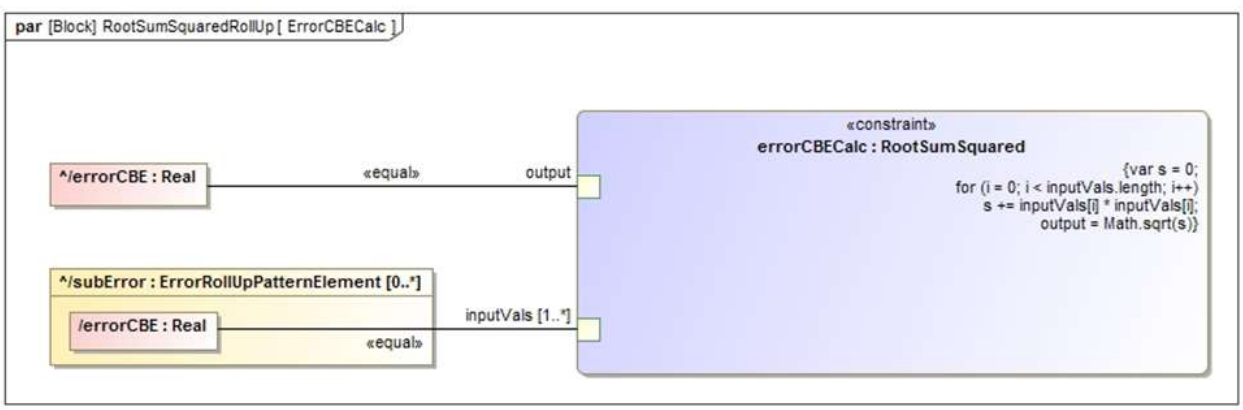


**Figure 21 Parametric Model of RSS Roll-up**

## Known Uses

Within the TMT, the APS uses this pattern to capture the alignment error of the M3 to APS interface. Fig 22 shows the alignment requirement and the required accuracy (0.03 percent) as a property-based requirement, refined from a textual requirement.

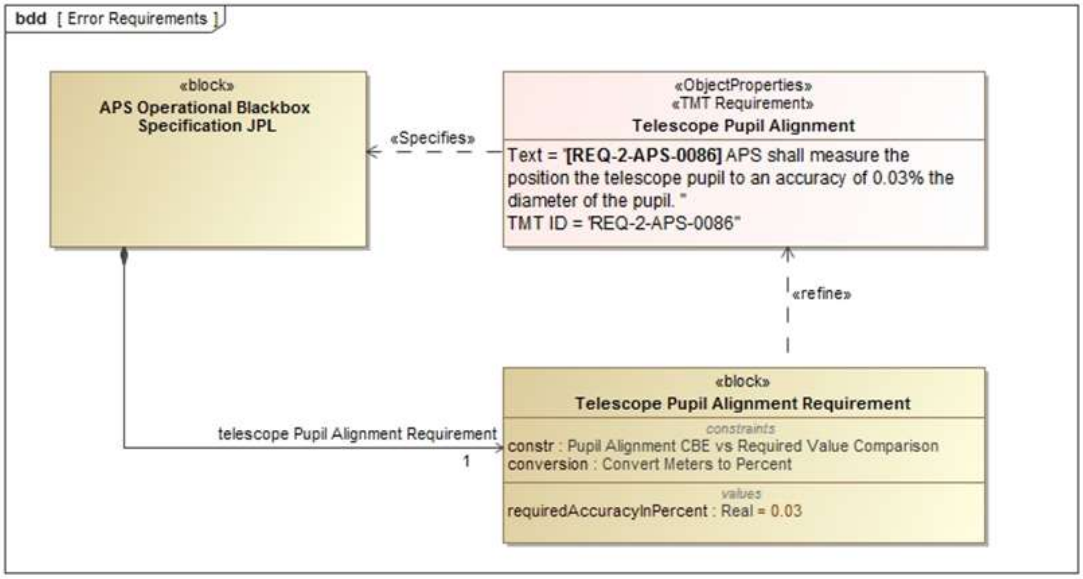


**Figure 22 Property-based Requirement**

The structure of the error budget is defined by an error budget tree that shows the hierarchy of system blocks that contain error values that contribute to the overall budget. The figures below show excerpts of the error breakdown tree, as shown in Fig 23

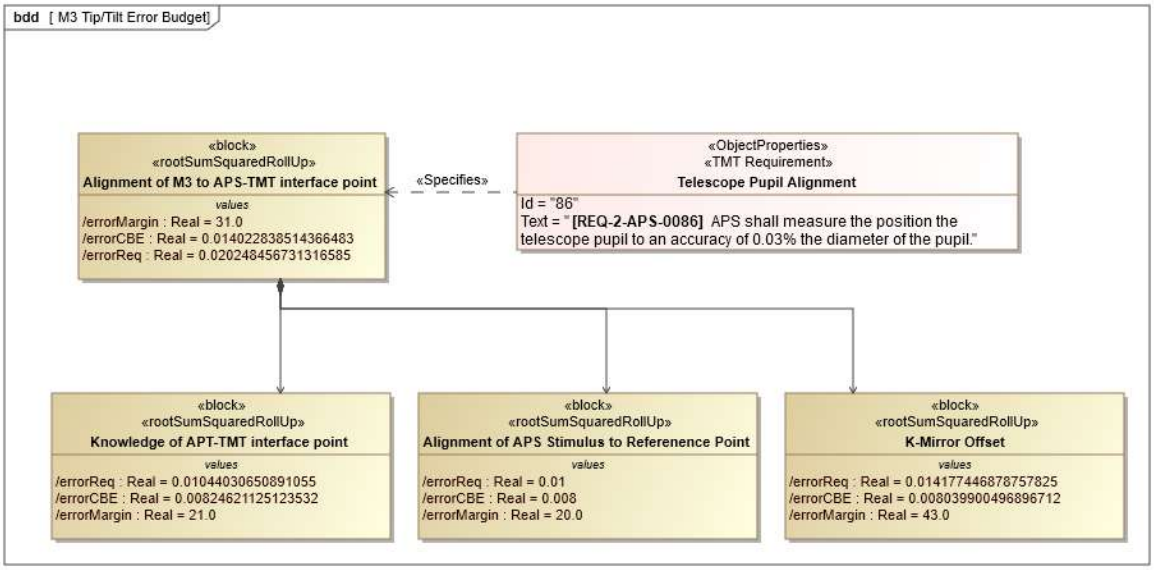


**Figure 23 Error Budget Tree**

Fig 24 shows the top of the M3 alignment error budget that ties the requirement to the top of the error budget tree.

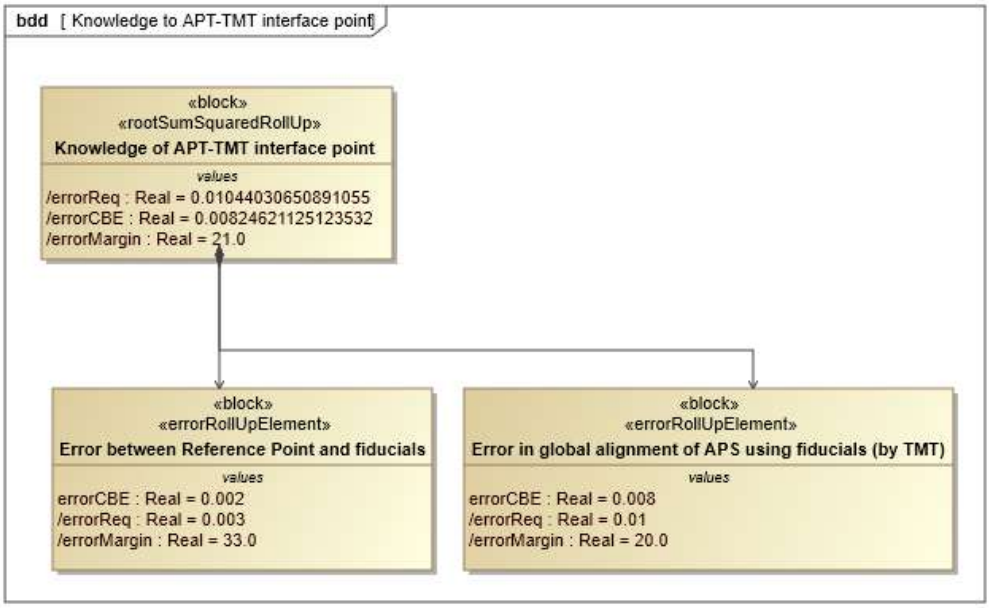


**Figure 24 Alignment Error Budget**

## 5. MODEL LIBRARIES

The OpenSE Cookbook is accompanied by the OpenSE model library that provides model templates (e.g. to support ESEM), structural elements (e.g. for organizational charts), and behavioral elements to facilitate the authoring and analysis of models following the described patterns.

## 6. DISCUSSION AND LIMITATIONS

The Cookbook is organized into a specification and tooling part. The specification part focuses on how the relevant information is captured in SysML and serves also as a specification for the tooling in the sense that the tools should support such SysML models in terms of syntax and semantic. Tools are also expected to use the examples as test cases to demonstrate their capabilities. The tooling part is extended as tooling support grows, which may consist of both open-source and commercial tools. However, to efficiently apply the specified patterns, adequate tooling support is a requirement. The current Cookbook version (2018) describes the usage of commercial and open-source tools (e.g. Cameo Systems Modeler, MDK-Systems Reasoner). It is expected that the tools supporting the patterns will expand.

## 7. RELATED WORK

The first version of a cookbook was the “COOKBOOK FOR MBSE WITH SYSML” [23] authored by the

Telescope Challenge Team [23] as part of the INCOSE MBSE Initiative. It focused on demonstrating the use of SysML for a non-trivial interdisciplinary system with the Active Phasing Experiment (APE) as a case study. APE is a technology demonstrator for extremely large telescopes. The case study and examples focused mainly on structural and requirements modeling, whereas the additions from the TMT focus on behavioral modeling.

## 8. CONCLUSIONS AND FUTURE WORK

The OpenSE Cookbook addresses systems engineering concerns by providing a collection of best practices, procedures, and patterns which are targeted on guiding systems engineers in their day to day work. The cookbook is organized in different volumes according to typical systems engineering tasks, describing how to capture, manage, and analyze system design related information and produce the required systems engineering products.

The Cookbook is built on proven patterns applied in TMT and APE production models, and supported by available tooling which facilitates the application of the recommended practices.

## 9. ACKNOWLEDGEMENTS

This research was carried out at the Jet Propulsion Laboratory (JPL), California Institute of Technology, under a contract with the National Aeronautics and Space Administration (NASA), the Thirty Meter Telescope Project, and at the European Southern Observatory.

The TMT Project gratefully acknowledges the support of the TMT collaborating institutions. They are the California Institute of Technology, the University of California, the National Astronomical Observatory of Japan, the National Astronomical Observatories of China and their consortium partners, the Department of Science and Technology of India and their supported institutes, and the National Research Council of Canada. This work was supported as well by the Gordon and Betty Moore Foundation, the Canada Foundation for Innovation, the Ontario Ministry of Research and Innovation, the Natural Sciences and Engineering Research Council of Canada, the British Columbia Knowledge Development Fund, the Association of Canadian Universities for Research in Astronomy (ACURA) , the Association of Universities for Research in Astronomy (AURA), the U.S. National Science Foundation, the National Institutes of Natural Sciences of Japan, and the Department of Atomic Energy of India.

## REFERENCES

[1] Trancho, G., Analyzing the Operational Behavior of NFIRAOS LGS MCAO, Acquisition on the Thirty Meter Telescope using SysML

[2] Karban, R., Dekens, F., Herzig, S., Elaasar M., Jankevicius, N., “Creating systems engineering products with executable models in a model-based engineering environment”, SPIE, Edinburgh, Scotland, 2016

[3] Karban, R., Hauber, R., and Weilkiens, T., “Mbse in telescope modeling," Insight 12(4), 24{31 (2009).

[4] Karban, R., Jankevicius, N., and Elaasar, M., “Esem: Automated systems analysis using executable sysml modeling patterns," in [INCOSE International Symposium], 26(1), 1{24, Wiley Online Library (2016).

[5] Karban, R., Jankevičius, N., Elaasar, M. “ESEM: Automated Systems Analysis using Executable SysML Modeling Patterns”, INCOSE International Symposium (IS), Edinburgh, Scotland, 2016

[6] Karban, R. “Using Executable SysML Models to Generate System Engineering Products”, NoMagic World Symposium, 2016

[7] OMG SysML, [Systems Modeling Language (SysML) Version 1.5], OMG, 2016

[8] OpenMBEE https://github.com/Open-MBEE

[9] TMT, “Thirty Meter Telescope.” http://www.tmt.org

[10] ISO/IEC, ISO/IEC 15288:2008(E), Systems and Software Engineering – System life cycle processes, International Organisation for Standardisation/International Electrotechnical Commission, February 1, 2008

[11] INCOSE “International Council on Systems Engineering.” http://www.incose.org

[12] OMG, “Object Management Group.” http://www.omg.org
[13] Friedenthal S, Moore A., and Steiner R., [A Practical Guide to SysML 3rd Ed.] Morgan Kaufmann OMG Press, 2014
[14] NoMagic “MagicDraw” http://ww.magicdraw.com
[15] NoMagic “Cameo Simulation Toolkit” https://www.nomagic.com/product-addons/magicdraw-addons/cameo-simulation-toolkit
[16] TMT model https://github.com/Open-MBEE/TMT-SysML-Model
[17] ISO/IEC, ISO/IEC 42010:2011, Systems and software engineering - Architecture description
[18] http://www.omgsysml.org
[19] Karban, Robert, M Zamparelli, B Bauvir, B Koehler, L Noethe, A Balestra, **Exploring Model Based Engineering for Large Telescopes**, SPIE Astronomical Telescopes and Instrumentation, 2008
[20] Analyzing the Operational Behavior of the Alignment and Phasing System of the Thirty Meter Telescope using SysML Sebastian J. I. Herzig, Robert Karban, Gelys Trancho, Frank G. Dekens, Nerijus Jankevicius, and Mitchell Troy, Adaptive Optics for Extremely Large Telescopes, Tenerife, 2017
[21] Luigi Andolfato, Robert Karban, Marcus Schilling, Heiko Sommer, Michele Zamparelli, and Gianluca Chiozzi, Experiences in Applying Model Driven Engineering to the Telescope and Instrument Control System Domain 2014, SPIE
[22] COOKBOOK FOR MBSE WITH SYSML 1.0 19/01/2011, http://mbse.gfse.de/documents/faq.html
[23] Telescope Challenge Team: http://omgwiki.org/MBSE/doku.php?id=mbse:telescope