Exploring Model Based Engineering for Large Telescopes - Getting started with descriptive models

R. Karban, M. Zamparelli, B. Bauvir, B. Koehler, L. Noethe, A. Balestra
European Southern Observatory, Karl-Schwarzschild-Str. 2, 85748, Garching bei München, Germany

ABSTRACT

Large telescopes pose a continuous challenge to systems engineering due to their complexity in terms of requirements, operational modes, long duty lifetime, interfaces and number of components. A multitude of decisions must be taken throughout the life cycle of a new system, and a prime means of coping with complexity and uncertainty is using models as one decision aid. The potential of descriptive models based on the OMG Systems Modeling Language (OMG SysML™) is examined in different areas: building a comprehensive model serves as the basis for subsequent activities of soliciting and review for requirements, analysis and design alike. Furthermore a model is an effective communication instrument against misinterpretation pitfalls which are typical of cross disciplinary activities when using natural language only or free-format diagrams. Modeling the essential characteristics of the system, like interfaces, system structure and its behavior, are important system level issues which are addressed. Also shown is how to use a model as an analysis tool to describe the relationships among disturbances, opto-mechanical effects and control decisions and to refine the control use cases. Considerations on the scalability of the model structure and organization, its impact on the development process, the relation to document-centric structures, style and usage guidelines and the required tool chain are presented.

Keywords: systems engineering, modeling, control system, project organization, SysML, requirements management, interface control, architectural design

1. INTRODUCTION

A Systems Engineer takes care of seven major tasks: State the problem, Investigate alternatives, Model the system, Integrate, Launch the system, Assess performance, and Re-evaluate. Those tasks are defined by the International Council On Systems Engineering (INCOSE, [22]) which was established in the beginning of the 1990's in response to the ever increasing complexity of modern systems and the need to recognize systems engineering as a profession.

Its mission is to advance the state of the art and practice of systems engineering in industry, academia, and government by promoting interdisciplinary, scalable approaches to produce technologically appropriate solutions that meet societal needs. As part of this mission several strategic initiatives have been started, where one is the Systems Engineering Vision defining a 15 years view of the evolution of the systems engineering discipline.

This initiative addresses several areas. The relevant area for this paper is Model-Based Systems Engineering (MBSE). It is considered one of the main means to tackle problems of current and future systems development. In this framework the European Southern Observatory (ESO) is collaborating with the German Chapter of INCOSE to form of a “MBSE Challenge” team. The task is to demonstrate solutions to "challenging" problems using MBSE. The Active Phasing Experiment (APE) [19], a European Union Framework Program 6 (FP6) project, has been offered as subject to the SE^2 Challenge team [18]. There are several other active challenge teams and general roadmap activities as shown in Figure 1.

Some engineers at ESO recognize that it is worthwhile to adopt a systematic and formalized modeling approach in order to manage a large and complex system like the European Extremely Large Telescope (E-ELT). As a result some independent but correlated activities to evaluate modeling in real world scenarios were started.

This paper presents results of MBSE using the System Modeling Language (SysML, [20]), both from the experiences within the E-ELT project and the MBSE Challenge project. The latter creates SysML models by reverse engineering existing documentation and from interviews with systems engineers, whereas for the former, the practices are applied to a new system.

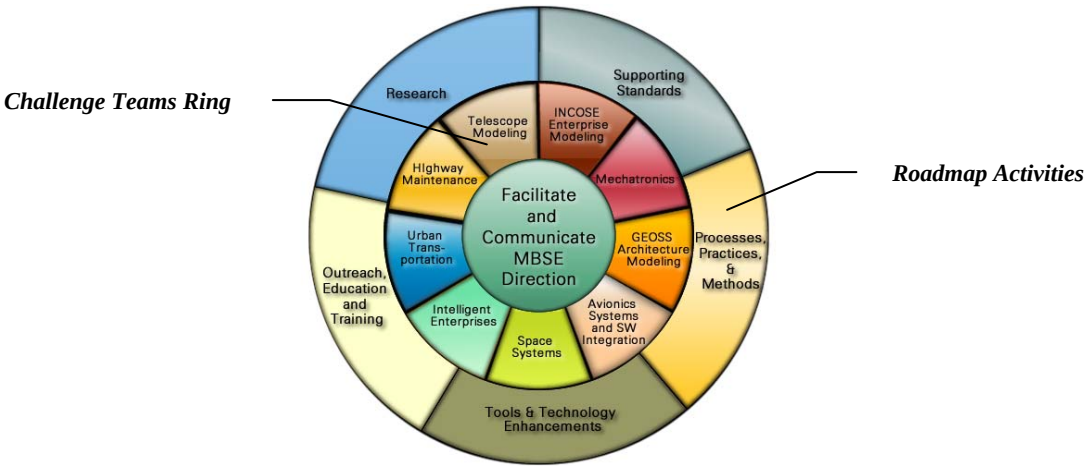

Figure 1 MBSE Initiative Organization Diagram (Courtesy [21]).

The requirements and objectives for MBSE are introduced, followed by definitions of common terms and concepts. The concerned projects, used as case studies, are described together with examples of MBSE applications. Based on this applications general modeling issues and critical areas are discussed.

2. REQUIREMENTS FOR MBSE

Basic Requirements are given in [5] and [3]. They are extended herein.

1. Facilitate communication with client and builder and among builders.
2. Help to reduce and control uncertainty and maintain system integrity.
3. Support the process of scoping, aggregation, partitioning, integration, certification, and performance prediction.
4. Provide a consistent integration of a multiplicity of system aspects, views and models for mechatronic systems.
5. Provide an overview of high complexity systems
6. The models must be understood without extensive education or experience.
7. Everything expressed in the model must have a single, defined and obvious meaning.
8. Reduce ambiguities in particular in cross-cultural, cross-language projects (e.g. outsourcing activities) and natural language.
9. Provide conceptual overview to support all (sub-) contracting activities (for suppliers it is important to know what they are working for, otherwise innovation is restricted. A model helps to ensure a common understanding among them).
10. Provide to system maintenance the necessary understanding of how the parts of the system contribute to its mission.

These requirements need to be met by carefully establishing modeling principles and practices outlined hereafter.

3. MODELLING FOUNDATIONS

Model-based systems engineering (MBSE) is the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases (INCOSE-TP-2004-004-02, Version 2.03, September 2007, [22]).

A model is an approximation, representation, or idealization of selected aspects of the structure, behavior, operation, or other characteristics of a real-world process, concept, or system (IEEE 610.12-1990, [23]), i.e. an abstraction.

A model usually offers different views in order to serve different purposes. A view is a representation of a system from the perspective of related concerns or issues (IEEE 1471-2000).

Many technical products in the telescope domain show an increasing integration of mechanics with electronics and information processing. The resulting systems are called mechatronic systems. Telescopes also contain optical

components and can be rightly considered as opto-mechatronic system. Building telescopes is therefore an interdisciplinary field which uses the synergy of the disciplines to deliver outstanding results.

The products of engineering fields cannot be designed and built standalone as they influence each other mutually and an optimal balance is to be found. In particular the interfaces cannot be conceived independently. Integrated and simultaneous engineering has to take place with the goal of designing an overall integrated system. The system has to be partitioned and the tasks distributed. A consistent integration of different system aspects and views is needed - a model. A set of models or aspects is considered consistent if they are abstractions of the same object, i.e. if at least one implementation exists of which the models are abstractions [3].

The main objective of a model is to provide the systems engineer the means to state the system problems comprehensively, to ensure that all requirements for a system are satisfied throughout the life cycle of the system and to develop a model on the basis of which a real system can be built, developed or deployed [1].

The basic tasks of systems engineering, namely to focus on the system as a whole (its total operation), leading the concept development and bridging the traditional engineering disciplines, is also reflected in the model. A model supports the understanding of the system as a whole, its operational objectives, and how all its parts work together.

The concept of a Common Project Model (CPM) is introduced in [5], to establish a common understanding of the system and the expectations. It is called the "project backbone".

A model supports furthermore the creation of system architecture. Architecture is the structure of a product, process or element [3].

The term architecting is promoted in [3] as a mean of handling today's systems that are very high quality, real-time, closed-loop, reconfigurable, interactive, software-intensive and largely automatic. Architecting is the process of creating and building architectures.

The concrete, deliverable products of an architect are models of the system. Important to architects are modeling methods that tie otherwise separate models into a consistent whole.

A promising means to formalize such modeling efforts is the Object Management Group's (OMG) System Modeling Language which was adopted as industry standard in 2007.

3.1 Taxonomy of models

Engineers in every field have always used models: physical models, architectural drawings specifications, prototypes, shop drawings, etc. In order to cope with complexity different models are required at different times. They can be classified in three basic categories:

Prescriptive Models: a predefined framework that serves as the basis for constructing a model of a system. They are the foundation for Descriptive Models.

Descriptive Models: a model used to depict the behavior or properties of an existing system or type of system; for example, a scale model or written specification used to convey to potential buyers the physical and performance characteristics of a computer (IEEE Std 610.3-1989).

Predictive Models: a mathematical model that predicts how some aspect(s) of a system will perform against specified boundary and initial conditions. They refer to all kinds of simulations (hardware in the loop, wind tunnel tests, etc.).

3.2 Life-cycle

The production of any system follows a life-cycle, either formalized or ad hoc. A model shall support the tasks and the creation of deliverables to meet system development process demands.

Using a model certainly has an impact, in particular in a CPM framework, because it will enforce a concerted interaction among all involved parties. A model shall be used to promote a common understanding of the project's mission.

There are of course different levels of model “infiltration” in the process. It ranges from simply creating consistent diagrams, which are used in the documentation, to a fully model driven process.

Model shall be used in accordance to the organization's cultural background. To move from document-centric to model-centric is definitely a major change which cannot happen overnight.

As mentioned by [7] capabilities and limitations of technology must be considered when developing a systems engineering development environment. Technology should not be used "just for the sake of technology". It should facilitate the systems engineering efforts.

Definitions of the elements of any system life cycle are mentioned in [7] and form together a methodology:

A Process (P) is a logical sequence of tasks performed to achieve a particular objective. A process defines WHAT is to be done, without specifying HOW each task is performed. A Method (M) consists of techniques for performing a task, in other words, it defines the HOW of each task. A Tool (T) is an instrument that, when applied to a particular method, can enhance the efficiency of the task. An Environment (E) consists of the surroundings, which enables (or disables) the WHAT and the HOW. Modeling has primarily an impact on the Method and on the Tools.

4. APE CASE STUDY

4.1 Project Description

APE represents a technology evaluation breadboard for large telescopes. The essential purpose of the experiment is to explore, integrate, and validate active wave front control schemes and technologies for a European Extremely Large Optical Telescope. This includes the evaluation and comparison of the performance of different types of wave front sensors in the laboratory and on the sky on the one hand, and the integration of the control of a segmented aperture control into an already existing active system and driving both the active system and the control of the segments on the other hand. APE is close to completion and deployment in an operational environment. APE will be deployed in the lab, standalone, but also in an already existing telescope.

APE, as any complex system, has a large number of functional, performance, physical and interface requirements which have to be satisfied. This implies the need for a professional requirements engineering and management during the project. This is the first application of SysML during the development.

APE consists of various elements, like wheels, translation stages, lenses, detectors, (segmented) mirrors, light sources, an interferometer, sensors and actuators (19 small axes, 10 TCCDs, 11 other devices, 183 actuators for segmented mirror). The control system alone consists of 12 computing nodes. These elements offer all kinds of optical, mechanical, electronic and software interfaces, both system internal and external to other systems. Their management alone is very challenging for the systems engineering team. Besides these challenges, which apply for many complex systems, APE has some other aspects:

It also challenges the control, since there are several open and closed loop systems required. A significant amount of data is produced by image processing data flows. Since APE will be deployed in the lab and in an already existing telescope, slightly different functional aspects are active depending on the deployment mode. Therefore different interfaces to existing systems are needed.

All these characteristics make APE well suited to evaluate SysML potential in tackling similar issues.

4.2 MBSE Challenge goals and outcome with APE

SysML is only a graphical language. It defines a set of diagrammatics, modeling elements, a formal syntax and semantics. As any language (formal or informal) it can be used in many different ways and many wrong ways too. Most notably the creation of nonsense models is possible.

The SE^2 MBSE challenge team [18] defines as its main goals to

- Provide examples of SysML, common modelling problems and approaches.
- Build a comprehensive model, which serves as the basis for providing different views to different engineering aspects and subsequent activities.
- Demonstrate that SysML is an effective means to define common concepts.
- Demonstrate that a SysML model enhances traceability

The results will be presented at the INCOSE Symposium 2008 and made publicly available [18]. The results consist of a list of Frequently Asked Questions (FAQ), which is illustrated by a SysML model. Available APE documentation and interviews are used to derive a formal model and guidelines for any new project which wants to use MBSE with SysML.

The FAQ covers topics like naming conventions, model annotations, external references; necessary system models, aspects and views; heuristics for modeling system requirements; guidelines for modeling the system structure (product tree, context variants, design variants, re-usable parts, system hierarchies); Interface modeling (logical and concrete, mechanical and flow, ports); allocation strategies; test case tracing; parametrics for performance models; style and layout issues.

5. E-ELT CASE STUDY

5.1 Project Description and MBSE activities

The E-ELT is a telescope with a primary 42-m diameter mirror is composed of 984 hexagonal segments, each 1.45 m in size, while the secondary mirror is as large as 6 m in diameter. In order to overcome the fuzziness of stellar images due to atmospheric turbulence the telescope needs to incorporate adaptive mirrors into its optics. A tertiary mirror, 4.2 m in diameter, relays the light to the adaptive optics system, composed of two mirrors: a 2.5-m mirror supported by 5000 or more actuators able to distort its own shape a thousand times per second, and one 2.7 m in diameter that supports field stabilization. This five mirror approach should results in an exceptional image quality, with no significant telescope aberrations in the field of view. The E-ELT will be more than one hundred times more sensitive than the present-day largest optical telescopes, such as the 10-m Keck telescopes or the 8.2-m Very Large Telescope (VLT) telescopes. Together with its sensitivity also the number of subsystems, interfaces and interactions among them has equally increased, thereby making the engineering challenge much more complex.

The applications of descriptive model based systems engineering in the E-ELT project are presented in the following. There are many applications of predictive modeling which are not considered here. The most important step before creating a descriptive model is to define a prescriptive one. The prescriptive model is found iteratively by modeling a (part of a) system which is well known. In our case the VLT served as a test bed.

MBSE can in principle be applied to the analysis and design of large projects in the areas listed in the next chapters with improvements to communication, problem understanding, system definition and design. However, most of its activities have been neither systematically nor widely applied in the E-ELT context yet (i.e. requirements analysis, documentation of error budgets).

Figure 2 shows indicatively the steps of the iterative systems engineering process where the different MBSE methods are applied.

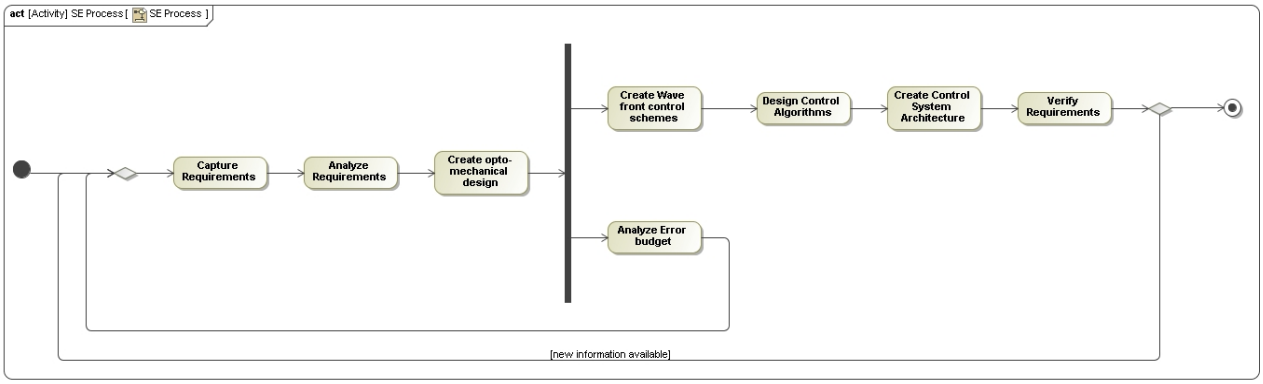

Figure 2 Partial Systems Engineering Process

5.2 Requirements Engineering

When multiple contributors to requirements come into play, version control becomes necessary and a Requirements Management Tool (RMT) can substantially improve managing requirements' dependencies and structure. SysML and graphical representation alone are not sufficient. Nevertheless, a requirements model is needed to define both.

SysML introduces the concept a requirement modeling element, defines a number of dependencies among requirements (refine, derive, contain, trace) and between other modeling elements, like design or tests (satisfy, verify) with a clear definition. It offers a graphical notation to visualize the structure (user, system, subsystem levels and their relationships)

of the requirements model. Since such diagrams often do not lend themselves to a suitable display on printed paper, tabular listings can also be used.

In a document-centric world, requirements belong to a document which might become applicable to other documents or only referenced by others, as specified in the respective sections of applicable and referenced documents. Several kinds of relationship are required for different contexts and have different effects on the applicability/reference tree. This is essential to properly represent single elements in requirement or design hierarchies and is also indispensable when using traceability reports. The SysML relationships are accordingly used as link types in the RMT. This facilitates the information exchange between an RMT and a SysML modeling tool (see 6.2).

At the beginning of a project it is often unclear how requirements are related and how they are structured. A graphical visualization is a big help in clearing the fog. As soon as the basic structure is defined it can be set up in the RMT. The graphical notation is only used to visualize the most relevant parts for the architecture of the system because it becomes difficult to read, display or print if the number of requirements exceeds the size of the printable page. Many different diagrams with many boxes and lines would have to be created.

A color code is used to visualize the stability of the requirements. After a first draft of requirements and opto-mechanical layout has been established the high level behavior to control the wave front can be defined.

5.3 Modeling Wave front control schemes

The bigger the telescope the more complex usually becomes its wave front control scheme. In the E-ELT, which is an adaptive telescope, there are many dependencies among its subsystems and a continuous flow of information.

We need to describe the wave front control scheme of the overall telescope, the relationships among internal/external disturbances (Figure 3), opto-mechanical effects, and control decisions (Figure 4), to understand the context of those decisions and to refine the control use cases. The wave front control schemes are derived by analyzing the system requirements and the opto-mechanical system architecture. From this fairly static view of dependencies, a wealth of information can be derived, like interfaces among components, communication network dimensions, synchronization requirements and required sensors and actuators.

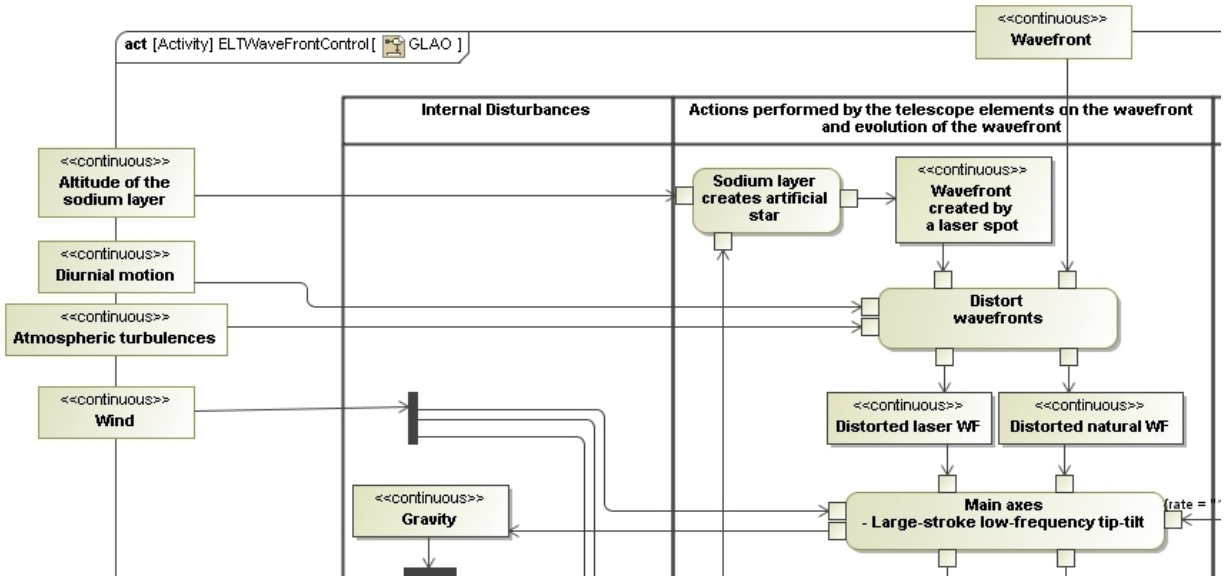

Figure 3 Disturbance modeling

A main requirement of MBSE is met here, i.e. facilitating the communication, in particular among Analyst, Control Engineer and Electronics/Software engineer. The model of the Wave front control becomes the central piece for further activities. The modeling element of choice is an activity diagram, using the SysML specific add-ons for systems modeling (rate, continuous flows, etc.). This prescriptive model maps the relevant information to activity elements (e.g. disturbances to object nodes, opto-mechanical effects to actions) and all the relevant information is kept in one diagram, thus enabling proper effects and relationship analysis.

The analyst's model (which defines the basic relationships) is extended by the control engineer with data volumes, sampling rates and latencies. This can be used in turn by the electronics/software engineers to conceive an appropriate control system architecture where those actions are allocated to.

Allocation is an extremely important SysML relationship. It allows formal tracing of all actions (describing the behavior) to the static system structure, which is constructed by the control system engineers (e.g. allocating "Sense phase steps" to a piece of Software which is allocated to a hardware node). The same model is used to describe behavior, analyze and quantify it, allocate it to appropriate control system structure. Using the same model ensures that those different views remain consistent. A modeling tool can automatically generate an N^2 matrix to visualize the dependencies among the actions.

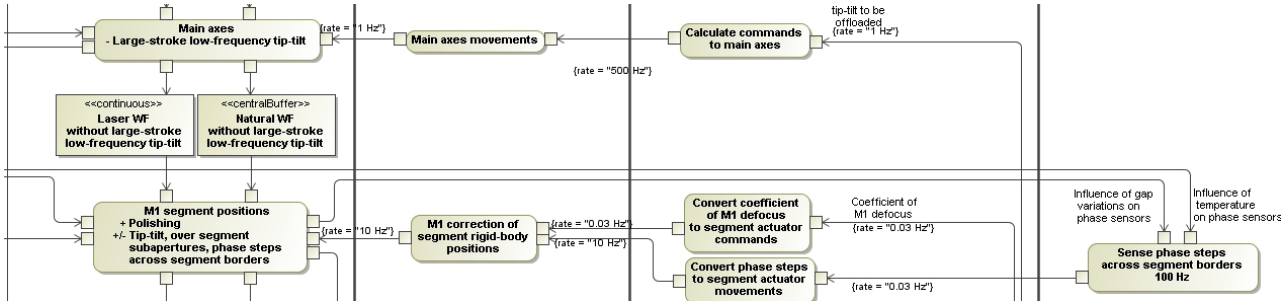

Figure 4 Opto-Mechanical effects and Control decisions

Analyzing behavior, requirements and control decisions an appropriate architecture for the control system can be created.

5.4 Control system design

The telescope control system uses sensors and actuators to deliver the required behavior, in particular for wave front control. A control system architecture needs to be defined as well as the interfaces to the various subsystems and their parts (Figure 6). The Technology DEMonstrator (TDEM), a control system prototype, and the E-ELT Telescope Control System are described with SysML models.

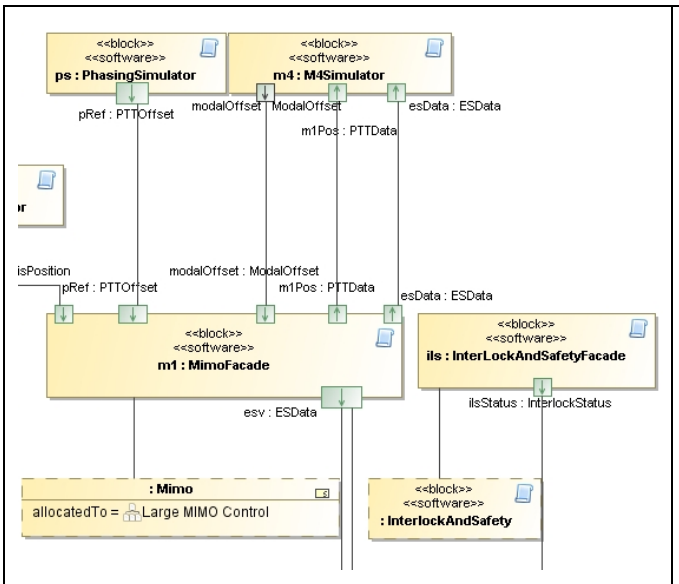

Figure 5 TDEM Data oriented control architecture

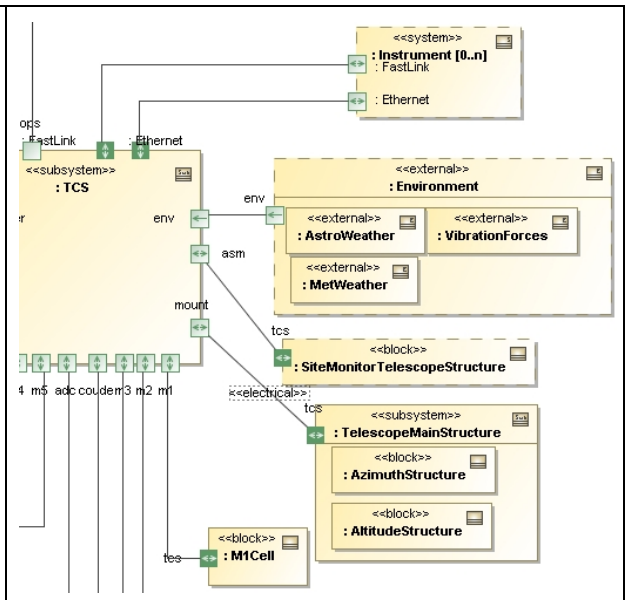

Figure 6 TCS context diagram

A data oriented control architecture is described with blocks and flow ports (Figure 5), behavior of control system components with activity diagrams and state machines.

When the major parts of the system are identified and the control architecture is defined, the elements have to be analyzed by their hazards and the control system needs to be extended to handle them.

5.5 Hazard analysis

The Hazard Analysis engineer's starting point is a collection of Safety Cases (following the Safety Case Methodology, defined in EN IEC62061 and EN ISO13849-1, [24]). These are semiformal textual representation of risk assessment and their possible/recommended mitigation. As a first step the system structure must be described, if not already available. This is used to identify the safety relevant parts. Typically these include sensors, actuators.

An analysis of mitigation information allows deriving a list of safety functions, like *Detect All Axis Overspeed*) (Figure 8). These are then allocated to specific parts, i.e. *altsplc* (Figure 9) and used to crosscheck compatibility with readily available Commercial Off The Shelf (COTS) components, like safety Programmable Logical Controllers (PLCs). The internal behavior of such parts best lends itself to a representation using state machines and activity diagrams. The former in particular highlights some specific behavior that the safety PLC has to comply with, according to general requirements (e.g. Integrity Check, Safe Run and Safe State). The latter are used to describe subsystem specific behavior.

It is important to notice that due to the overriding capability of the ILS, the Hazard Engineer effectively models two parallel and independent control systems in the same model, i.e. *altcs* and *altsplc* (Figure 7). This in turn results in more easily identifying dependencies and overlaps. At the end of this process all safety functions must have been allocated to safety components.

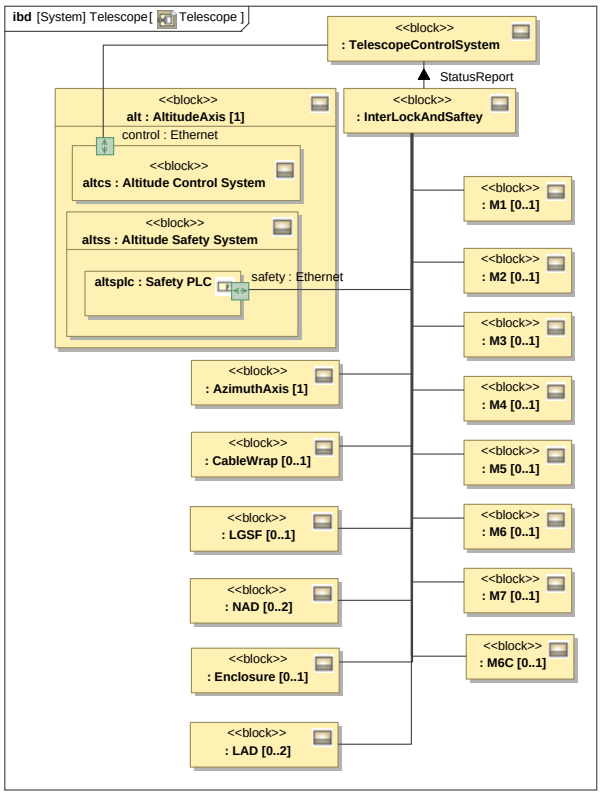

Figure 7 Internal structure of Telescope block

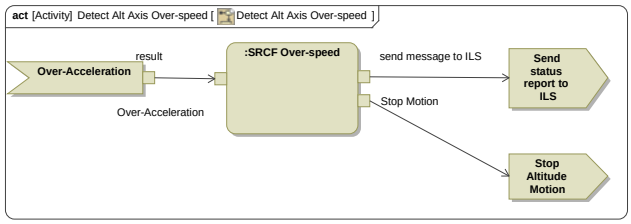

Figure 8 Behavior of altitude safety system

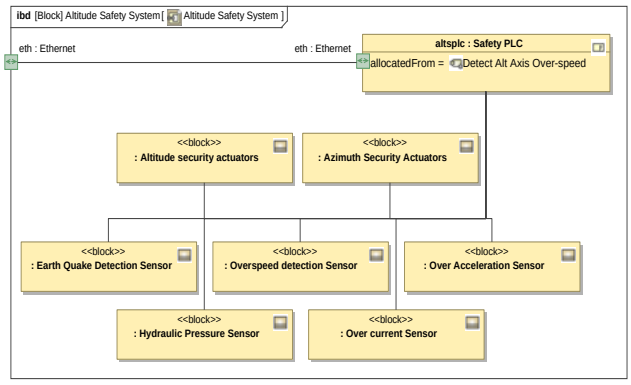

Figure 9 Allocation of behavior to structure

When the major properties of the system and its environment are known its performance can be analyzed with an appropriate model.

5.6 Determining Optical Error Budget Structure

The development of error budgets is a critical step in propagating the performance requirements derived from scientific needs to sub-system requirements. Error budgets are also used to assess the impact of unsatisfied requirements on the system performance and help define remedial actions. The elements contributing to the error budget are properties of the system (e.g. mirror polishing errors, control performance, etc.) and the environments (e.g. disturbance characteristics) and therefore part of the system descriptive model.

SysML parametric diagrams model a network on constraints on system properties to support engineering analyses and, as such, are a means to describe the structure of error budgets. E.g. blocks, value types and block definition diagrams are used to define the system and its properties; constraints (Figure 11) and parametric (Figure 10) diagrams describe their relationships.

The application of SysML parametric diagrams exist only as a concept and today are only used for documentation purposes. The tasks involved in developing an error budget encompass: understanding the underlying physics, defining and evaluating predictive models and performing trade-off analyses, for which a descriptive model fall short of providing much support. [8], [9], [12] and [13] attempt to address the latter shortcomings by describing the means and methods of extending SysML models with large-scale computation capabilities.

The following figures show a simplified model where the Normalized Point Source Sensitivity (PSSn) has been used as an error metric for the image quality, which depends on parameters like wavelength, seeing and altitude angle. Note however, that the system structure is the same as for the hazard analysis but exposing a different view.

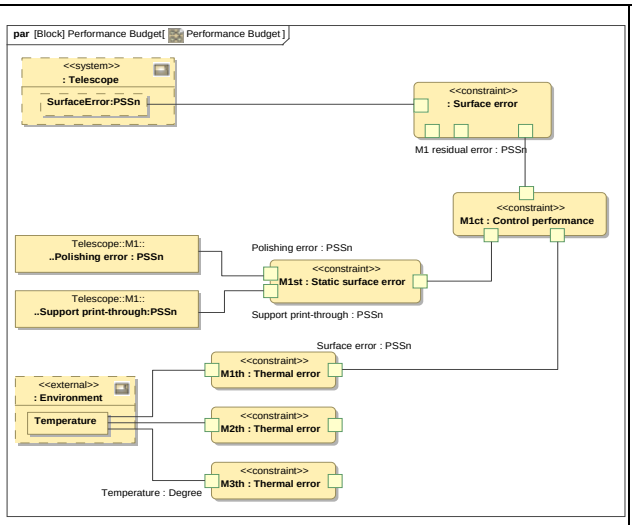

Figure 10 Parametric diagram, describing relationships

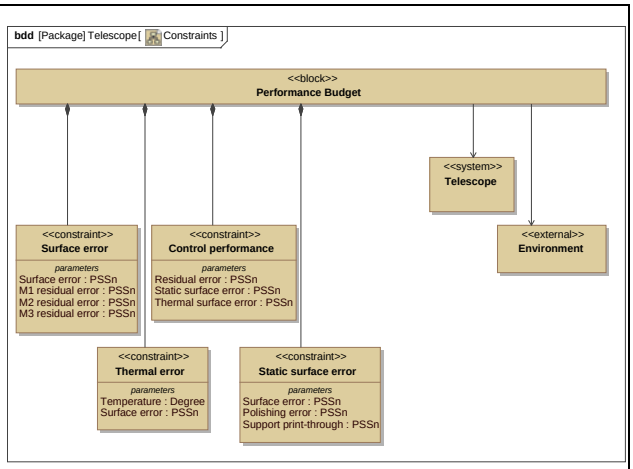

Figure 11 Constraints structure definition

The various aspects and views of the system need to be traced to each other to ensure consistency and increase the quality of the final system. Everything needs to be tested. A verification model is used to ensure that.

5.7 Test Case tracing - Verification

A properly built up model allows the tracing of many different decisions and dependencies. It ranges from tracing derived requirements, over satisfaction of requirements by design elements to verification of requirements by test cases.

Test cases are defined using the (test) use case steps which are a refinement of the requirements. The actors represent the roles involved in the test. For each use case there is one or more test cases which realize the use case (described by an activity diagram) and in fact, a single use case can verify multiple system requirements that are realized by multiple test cases.

Although the verification diagram (Figure 12) might look a bit cluttered with its arrows for all those dependencies traceability matrices can be automatically generated with the appropriate tool. The following figures show tracing of test cases, the related the verification of requirements and an automatically generated matrix (Figure 13).

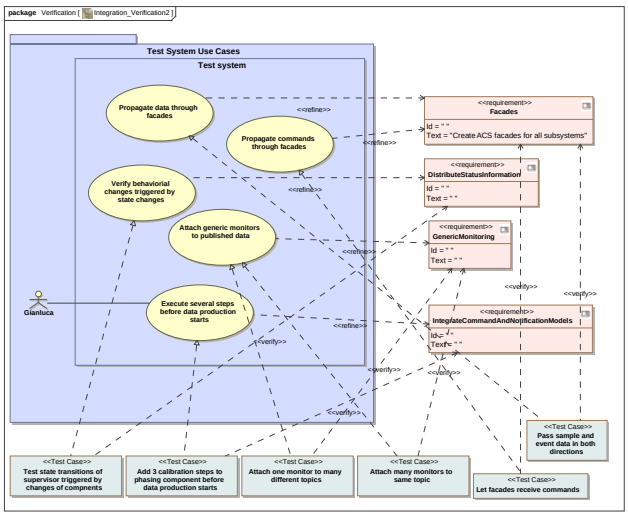

Figure 12 Test Case tracing

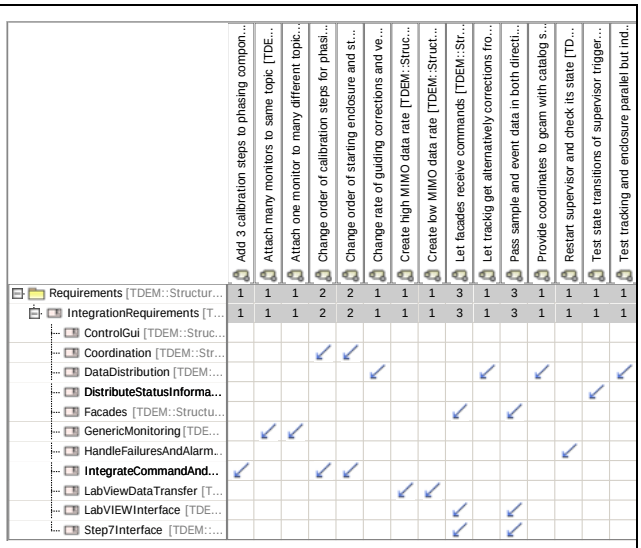

Figure 13 Traceability Matrix

6. CRITICAL AREAS

6.1 Guidelines and mentoring

Sets of guidelines illustrated by real-world examples are a necessary condition for successful modeling. Without them one can easily waste time by inventing modeling recipes. The risk is high that the model will eventually become expensive to maintain.

When starting up, modelers are often facing problems which cannot be immediately attributed to unsuitable modeling recipes, wrongly applied formal language or misusage of a tool.

Even more time is wasted when a model grows without being based on sound foundations. Some experienced person should accompany modeling activities in order to be efficient. This is a must when starting with MBSE.

6.2 Tool-chain and integration

As a start, diagrams can simply be created using SysML with a general drawing tool, instead of a specialized modeling tool. The difficulty will then be to structure and organize them and to keep them consistent as they change along the system's life.

It is most effective to create a model and create different diagrams according to the information that needs to be communicated. If it is built on a well thought structure it can be easily extended and maintained (see chapter 4.2). The more formal is the tool, the better, because it helps in applying properly SysML. With permissive tools (not enforcing SysML standards) the resulting model will most probably not be compliant with the language specification and therefore the added communication value will disappear.

For long lasting projects the tools will not remain the same over their lifetime. Provisions need to be taken to handle the change, like selecting tools whose models can be stored in an interchangeable format.

As stated in [5], it is not possible to overview the system and at the same time keep an understanding of all its details. A collection of diagrams alone (created by a simple drawing tool) will unlikely be sufficient to keep the balance between different aspects: a model as such fits more this purpose.

For document-centric organizations a smooth integration of the artifacts created with a modeling tool (“the diagrams”) and a printable document must be ensured. It is almost unthinkable that an organization changes its culture. Therefore the creation of human readable documents is a very important issue.

The experience shows that a SysML modeling tool and SysML requirements diagrams in particular are not sufficient to manage requirements (see 5.2). It has to be decided where requirements are primarily managed (i.e. where the master is and modeling tools normally have limitations in properly maintaining a database of requirements. A strategy is to manage all requirements in a requirements management tool and then export them to the SysML model, but current tools manufacturers seem to be falling short of this goal.

6.3 Configuration control

As soon as a common project model is created and more than one person uses it, configuration control becomes a fundamental requirement. In particular consistent linking among model elements must be ensured. Individual changes must be traceable as well as creating visual differences to follow in detail what has changed where. Due to the extensive linking, side effects (introduced by changes) can go unnoticed and make the model incorrect. This can only be mitigated by establishing rigorous configuration management practices and using tools which allow roll-backs.

7. CONCLUSION

In the models which have been partially shown throughout this paper we have observed substantial information overlap which could not be ascribed to the specifics purposes of the models (e.g. the basic system structure was created several times). While it is obvious that a Common Project Model (CPM) would address this issue, it is to be expected that an all-out CPM approach would be premature if not counterproductive in the early stages of individual modeling attempts, by introducing dependencies which would stifle creativity.

Nevertheless common modeling practices, guidelines and lessons learnt not only enhance productivity but also facilitate communication across the disciplines within a project. Even in a non CPM setup, they help building homogeneous models and are indispensible if a transition to CPM is considered worthwhile at a later stage.

One of the fundamental questions that anyone should ask to himself when modeling is: “what information do I want to convey?” A pragmatic view suggests that modeling should stop when the added value to the user is smaller than the effort, i.e. when all the necessary information has been captured to unambiguously state the problem and/or the solution.

Experience showed that proper tool training and modeling mentoring provide the necessary conditions for successful MBSE introduction in an organization with a consolidated, document-centric culture.

MBSE practices come at a cost, most notably mentoring, and their adoption requires tailoring to each particular domain. Nevertheless in all disciplines and tasks where MBSE was applied, we have experienced a remarkable increase in the depth and effectiveness of discussions aimed at understanding the problem and exploring the solutions space. The resulting communication exchange was more timely, precise and widespread than if MBSE had not been used.

REFERENCES

[1] Wymore, A. W., [Model-Based Systems Engineering], CRC Press, London New York Washington D.C. (1993)

[2] Kossiakoff, A., Sweet N. W., [Systems Engineering Principles and Practice], Wiley, New Jersey (2003)

[3] Maier, W. M., Rechtin, E., [The Art of Systems Architecting], CRC Press, London New York Washington D.C. (2002)

[4] Isermann, R., [Mechatronic Systems], Springer, London (2005)

[5] Ogren, I., "On principles for model based systems engineering", Systems Engineering Journal (2000)

[6] Hause, M., "The SysML Modeling Language", Fifth European Systems Engineering Conference (2006)

[7] Estefan, A. J., "Survey of Model-Based Systems Engineering (MBSE) Methodologies", INCOSE MBSE Focus Group (2007)

[8] Johnson, T., Paredis, J.J.C., Burkhart R., "Integrating Models and Simulations of Continuous Dynamics into SysML", Deere & Company (2007)

[9] Johnson, T., Paredis, J.J.C., Burkhart R., Jobe M. J., "MODELING CONTINUOUS SYSTEM DYNAMICS IN SYSML", Proceedings of the IMECE (2007)
[10] Barnhart, A. E., "Why Use SysML?", Presentation
[11] Friedenthal, S., Moore, A., Steiner A., "OMG Systems Modeling Language (OMG SysML) Tutorial", INCOSE (2007)
[12] Peak R., Friedenthal, S., Burkhart, R., Wilson, M., Bajai, M., "Simulation-Based Design Using SysML Part 1: A Parametrics Primer", INCOSE Intl. Symposium, San Diego (2007)
[13] Peak, R., Friedenthal, S., Burkhart, R., Wilson, M., Bajai, M., "Simulation-Based Design Using SysML Part 2: Celebrating Diversity by Example", INCOSE Intl. Symposium, San Diego (2007)
[14] Bock, C., "SysML and UML 2 Support for Activity Modeling", Wiley InterScience (2005)
[15] Friedenthal, S, “private communications”, Lockheed Martin, (2007-2008)
[16] Kelly, D., “private communications”, NoMagic, (2007-2008)
[17] Friedenthal, S., Sampson, M., Griego, R., "INCOSE Model Based Systems Engineering (MBSE) Workshop" (2008)
[18] SE^2 MBSE Challenge Team, "Telescope Systems Modeling", http://www.myway.de/mbse, INCOSE International Symposium (2008)
[19] Gonte, F., Y. J., Yaitskova, N., Dierickx, P., “APE: a breadboard to evaluate new phasing technologies for a future European Giant Optical Telescope”, Proc. SPIE 5489 (2004)
[20] OMG Systems Modeling Language (OMG SysML), v1.0, formal/07-09-01, http://www.omgsysml.org, OMG (2007)
[21] MBSE Challenge Team (Griego R., Dee M.), “Enterprise Modeling”, private communications (2008)
[22] International Council On Systems Engineering (INCOSE), http://www.incose.org
[23] IEEE, http://www.ieee.org
[24] ISO, http://www.iso.org

GLOSSARY

Definition of a system

A system is a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results. The results include system level qualities, properties, characteristics, functions, behavior and performance. The value added by the system as a whole, beyond that contributed independently by the parts, is primarily created by the relationship among the parts; that is, how they are interconnected ([3]).

Definition of Systems Engineering

Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem: Operations, Cost & Schedule, Performance, Training & Support, Test, Disposal and Manufacturing ([22]).

Definition of Complexity

Complexity is a measure of the number and types of interrelationships among system elements. Generally speaking, the more complex a system, the more difficult it is to design, build, and use ([3]).

ACKNOWLEDGMENTS

Special thanks go to Sandy Friedenthal from Lockheed Martin who has always been available for discussions, questions and guidance and provided motivation to invest time into MBSE. Darren Kelly from NoMagic was of invaluable help in properly using MagicDraw as a tool as well as giving concise and comprehensive modeling recommendations.