MBSE Practices in Telescope Modeling

Robert Karban, rkarban@eso.org; Tim Weilkiens, tim.weilkiens@oose.de, R. Hauber, rudolf.hauber@hood-group.com and R. Diekmann, rainer.diekmann@hamburg.de

Section I: Introduction

This article presents the results of model-based systems engineering using the OMG Systems Modeling Language (OMG SysML), drawing on experiences within the INCOSE MBSE Challenge project. It is a special version of the arcticle "MBSE in Telescope Modeling" (INSIGHT 12/2009) that has been written for the OCSMP certification. The first part of the article relates primarily to the second certification level Model Builder Fundamental. The second part is a reference for the third certification level Model Builder Intermediate.

In the framework of INCOSE's strategic initiative, the *Systems Engineering Vision 2020*, one of the main areas of focus is model-based systems engineering. In keeping with this emphasis, the European Southern Observatory (ESO; http://www.eso.org/) is collaborating with the German Chapter of INCOSE (http://www.gfse.de/) in the form of the "MBSE Challenge" team SE^2. The team's task is to demonstrate solutions to challenging problems using MBSE. The Active Phasing Experiment (APE; see Gonte et al. 2004), a European Union Framework Program 6 project, was chosen as the subject of the SE^2 Challenge Team (http://mbse.gfse.de/). Many technical products in the telescope domain show an increasing integration of mechanics with electronics, information processing, and also optics, and can therefore be rightly considered as optomechatronic systems.

The SysML models were created by reverse engineering from existing documentation and from interviews with systems engineers. We will make use of Ingmar Ogren's concept of a common project model (Ogren 2000) to establish a common understanding of the system.

You'll find a complete version of the model and other material like presentations on our website: http://mbse.gfse.de.

Project Description

Our system case study is the Active Phasing Experiment technology demonstrator for the future European Extremely Large Telescope, which is a high-tech, interdisciplinary optomechatronic system in operation at the Paranal observatory (see ESO 2009). The next generation of telescopes needs to collect significantly more light than older telescopes, therefore requiring bigger reflecting surfaces that consist of many individual mirror segments. Due to different disturbances (such as vibrations, wind, and gravity), the segments must be actively controlled to provide a continuous mirror surface with a phasing error of only a few nanometers over the main mirror's diameter of 42 m. The main challenge is to correctly detect the positioning errors of the segments via specific phasing sensors in order to create a continuous mirror surface.

APE was developed to evaluate those sensors, and was installed on one of the 8 m telescopes that constitutes part of the Very Large Telescope in Chile (VLT) for sky tests. APE can be seen as the black box in figure 1. For the installation at the telescope it had to comply with various mechanical, electrical, optical, and software interfaces. APE consists of about two hundred sensors and actuators such as wheels, translation stages, lenses, detectors, mirrors, light sources, an interferometer, and twelve computing nodes for control. Since APE had to be deployed in the test lab and in an already existing telescope, for each context it was necessary to model variants of function, interfaces, and structure. All of these characteristics made APE well suited to evaluate the potential of SysML in tackling similar issues.

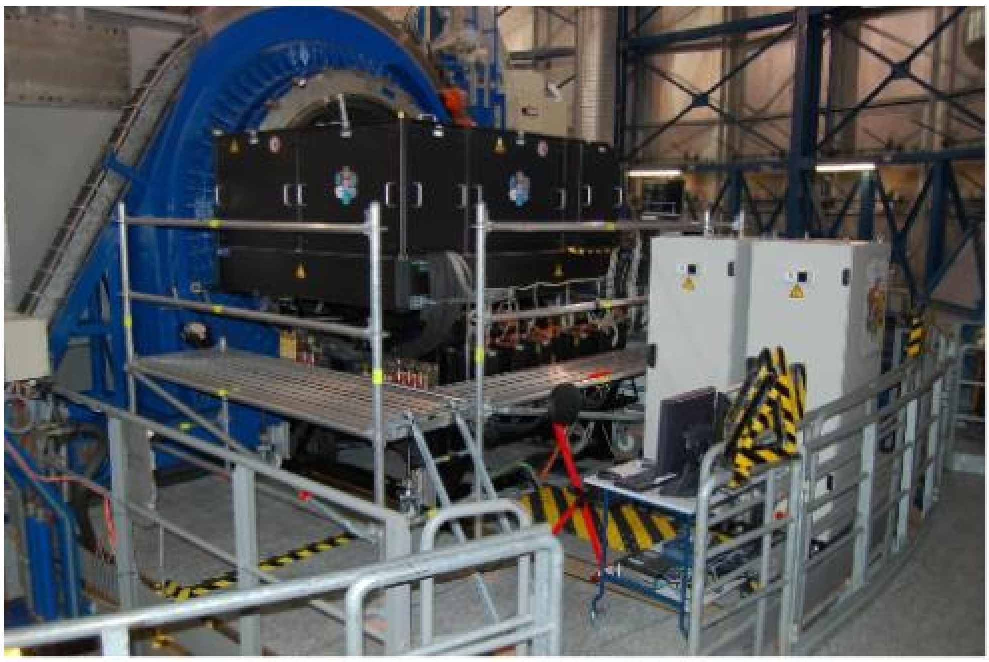


Figure 1: Active Phasing Experiment at the Very Large Telescope

MBSE Challenge Goals

SysML is a graphical language and defines a modeling elements with a notation, a formal syntax, and semantics. The SysML model is not merely a mental abstraction, but a collection of complex data structures that can be edited, augmented, queried, and reported on by means of a suitable tool, which is an indispensible pillar for MBSE. Like any language (formal or informal), it can be used in many different ways, including many wrong ways. The main goals of the SE^2 MBSE Challenge Team are to

- create modeling guidelines and conventions for all system aspects, hierarchy levels, and views;
- provide examples in SysML, solving common modeling problems;
- build a comprehensive model, which serves as the basis for providing different views to different engineering aspects and subsequent activities; and to demonstrate that SysML is an effective means to support systems engineering.

A SysML model, as described in the next section, illustrates the results of this modeling effort to date.

Methodology

SysML is a language and doesn't prescribe any methodology. For example, SysML allows one to model the "allocate" relationship between nearly any model elements. But where is this feature useful in a specific project? How can the relationship be determined from the model, e.g. for traceability? What are the consequences of having an "allocate" relationship between two elements? You don't find answers to these questions in the SysML specification.

We did not define a MBSE methodology. Our modeling work is based on different existing methodologies like SYSMOD (Weilkiens 2008), OOSEM (Friedenthal 2008) or Wymore (). Our MBSE Challenge Team has found some best practices and modeling guidelines to complement them. Last but not least, each project needs its own specific set of methods.

Section II: MBSE in Telescope Modeling for OCSMP Model Builder Fundamental

Model Structure

Modeling is all about abstraction (reducing the complexity of a system), improving communication and understanding of the system, and providing reusable system elements. However, capturing a lot of different aspects like requirements, structure, interfaces, behavior, and verification in a model of a complex system like APE leads to a large model. Therefore, the first challenge is to find a clear, intuitive structure for the model, since a well-structured model is crucial for controlling complexity.

In the APE model we applied two techniques to establish a good and easily understandable structure: recursive structure patterns and aspects. In SysML, packages are the structuring mechanism to group model elements, which were used by the authors for both techniques. Packages are a mechanism to group model elements in higher-level units, similar to the way folders organize files in a computer's file system.

APE Model Structure Pattern

The APE model uses a recursive structure pattern to model different system aspects on every level of decomposition of the APE. On each level we have packages for these aspects. See *Guidelines for organizing the model* below for a list of the packages.

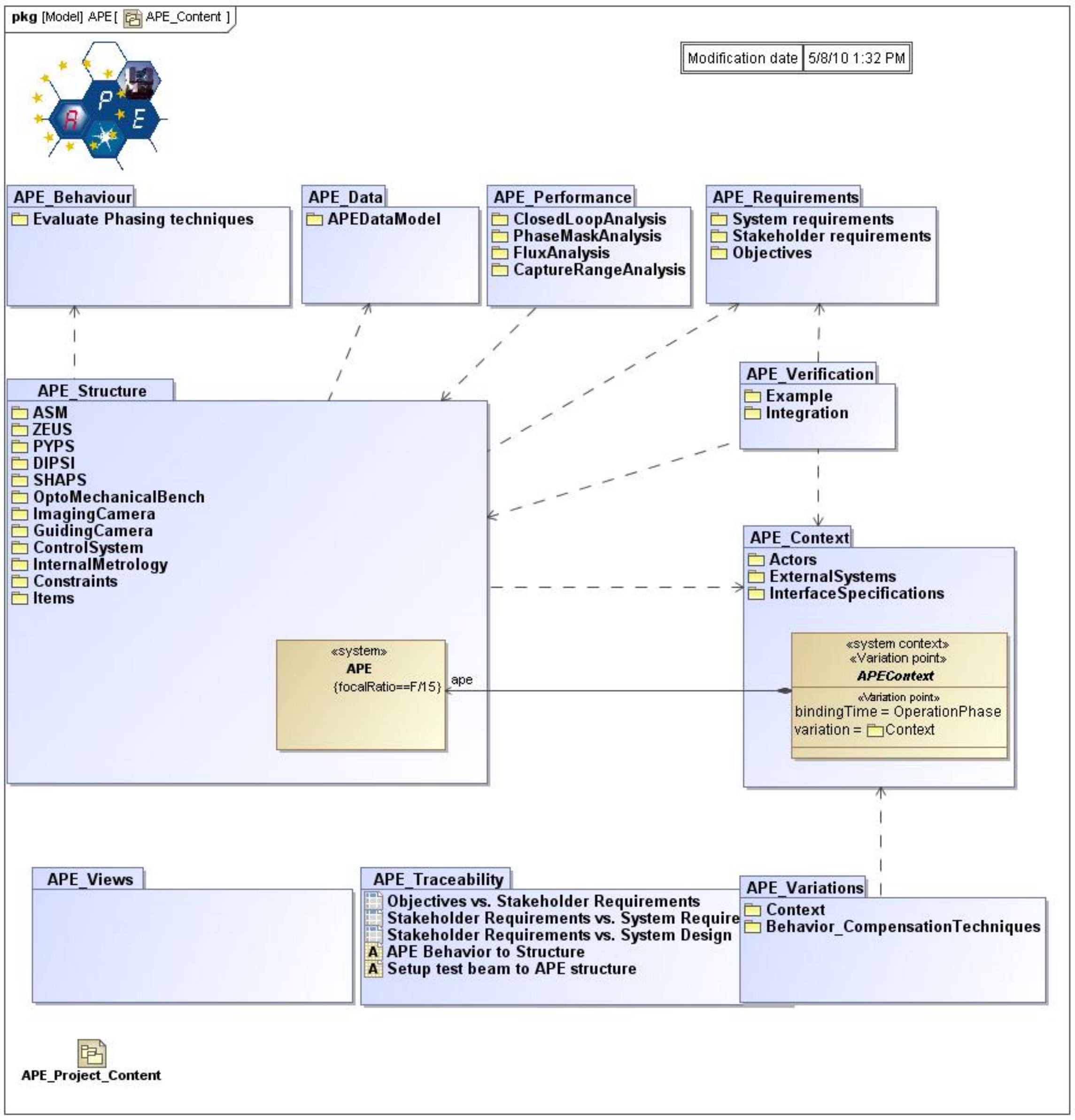


Figure 2: Top Level Content Diagram

The System Structure aspect ("APE_Structure" in figure 2) is used to decompose the system and provide the recursive modeling pattern within the subsystem package; for example, the "APE_Structure" sub-package "InternalMetrology" contains the same packages. The "InternalMetrology" sub-package "PhaseModulator" (figure 3) contains again the same packages, and so on. For every system decomposition element (like the nested structure for the "InternalMetrology" in figure 3), a package exists together with an overview diagram that shows the aspect packages of the respective element. The arrows between the packages show their dependencies. This recursive model structure provides an intuitive look-and-feel navigation capability within the APE model.

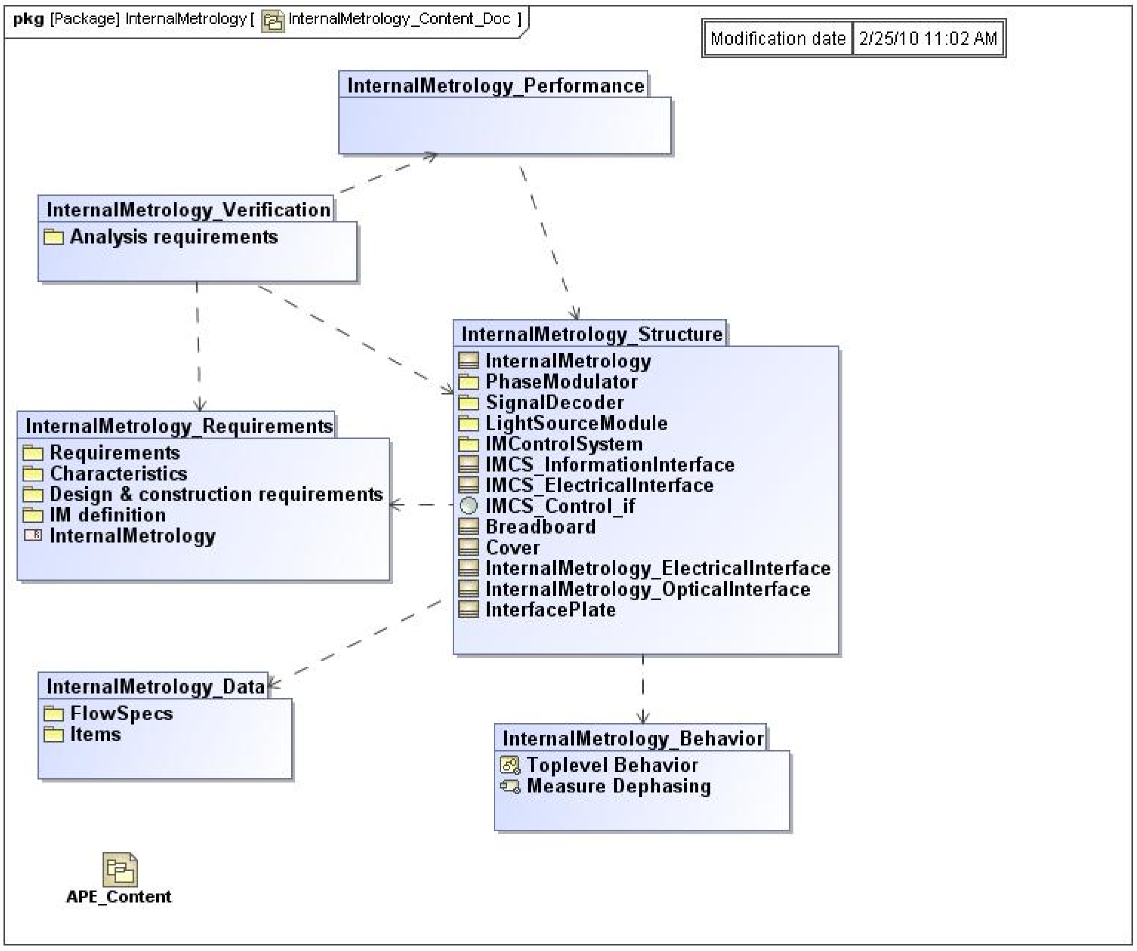

pkg [Package] InternalMetrology [InternalMetrology_Content_Doc]
Modification date
2/25/10 11:02 AM
InternalMetrology_Performance
InternalMetrology_Verification
Analysis requirements
InternalMetrology_Structure
InternalMetrology
PhaseModulator
SignalDecoder
LightSourceModule
IMControlSystem
IMCS_InformationInterface
IMCS_ElectricalInterface
IMCS_Control_if
Breadboard
Cover
InternalMetrology_ElectricalInterface
InternalMetrology_OpticalInterface
InterfacePlate
InternalMetrology_Requirements
Requirements
Characteristics
Design & construction requirements
IM definition
InternalMetrology
InternalMetrology_Data
FlowSpecs
Items
InternalMetrology_Behavior
Toplevel Behavior
Measure Dephasing
APE_Content

Figure 3. Subsystem content diagram

Aspects

Packages are also used to separate specific aspects of the system at each decomposition level. Each package provides one or more aspects on APE for a specific perspective, such as context, structure, data, and so on.

Overview Diagrams

The third point to improve the understandability of the APE model is that we provide content diagrams at each level of the recursive model structure, which describe the system by showing all the different aspects captured by the model. The top-level overview diagram is a "project content" diagram (figure 2) and serves as an entry point and an anchor for navigation through the complex model. Content diagrams are SysML package diagrams.

Guidelines for organizing the model

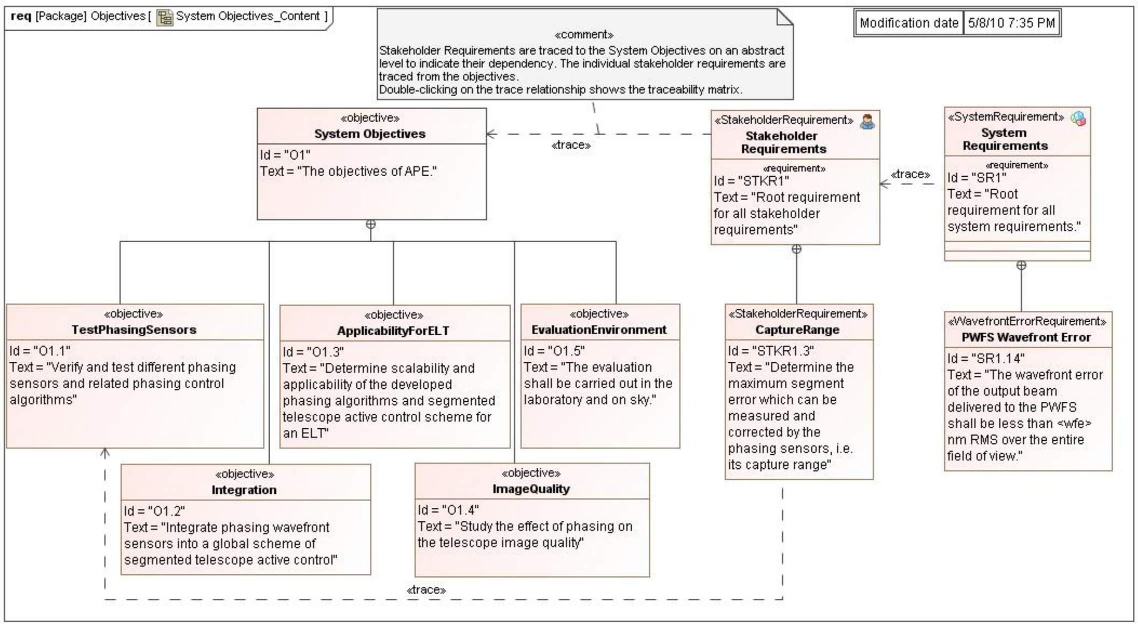
To increase readability of and support navigation within the model, the following guidelines for setting up model packages should be followed:

- Structures are organized in packages according to the system's product tree.
- The packages reflect the different concerns (each serving a distinct purpose) of the system. Having, conceptually, independent aspects allows:
  - Independent reorganisation, if needed
  - Easier integration (import) of an external requirements model (like Doors)
  - Each concern becomes simpler.

    - You can link in a more obvious way the model elements with traceability relationships (e.g. trace, satisfy)
- Use recursive structuring of packages: packages created at the highest level of the model shall be repeated for every lower level
- Create separate packages containing model information, references to external information (like Drawings, existing Requirements or Design documents) and information about the model. The latter are prefixed with _ (underscore) to have them at the end of the sort order to increase readability.
- The top level package is <projectname>_Project
- The package Context exists only at the top level because the lower level context are implicitely defined by the respective IBDs. Lower level context diagrams are only useful if the sub-model is given to an external contractor.
- The BDD of each substructure defines its elements and therefore its product tree.
- Comments, Errors and Issues are in separate packages to find them easily.
- Recurring packages in every model level are:
    - Data
    - Requirements
    - Structure
    - Performance
    - Variations
    - Verification
    - Views
    - Problems
    - Rationales
    - _Comments
    - _Drawings
    - _Errors
    - _External
    - _Glossary
    - _Issues
    - _References
- On the top level the Requirements package contains Objectives, User requirements and System requirements. Whereas on lower levels it contains only derived requirements of the sub structures. They are kept together with the design to have a consistent package, e.g. to give to a contractor.
- All packages related to a certain (sub) system have as a prefix the name of the (sub) system, i.e. its directly owning package to have a unique identification of the package for navigation. e.g. APE_Requirements, ASM_Requirements (but not APE_ASM_Requirements).

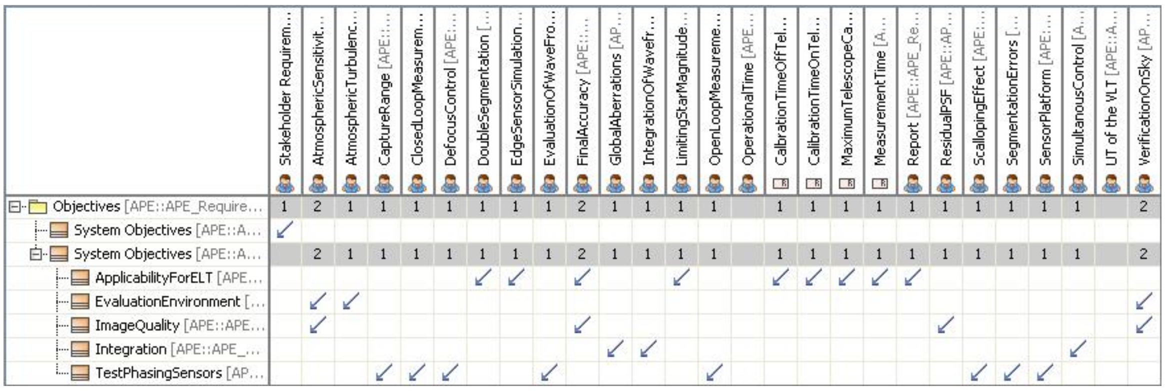
Objectives and Requirements

The next sections describe the APE modeling approach for the aspects mentioned above. APE, like any complex system, has a large number of functional, performance, physical, and interface requirements that have to be satisfied. This implies the need for formal requirements management during the project. APE has about 50 high-level system requirements. The control system has also about 50 requirements, refined by 150 use cases. We used the SysML requirements diagrams to show the main objectives of APE (figure 4).


req [Package] Objectives [System Objectives_Content]
Modification date 5/8/10 7:35 PM
«comment»
Stakeholder Requirements are traced to the System Objectives on an abstract level to indicate their dependency. The individual stakeholder requirements are traced from the objectives.
Double-clicking on the trace relationship shows the traceability matrix.
«objective» System Objectives
Id = "O1"
Text = "The objectives of APE."
«trace»
«StakeholderRequirement» Stakeholder Requirements
«requirement»
Id = "STKR1"
Text = "Root requirement for all stakeholder requirements"
«trace»
«SystemRequirement» System Requirements
«requirement»
Id = "SR1"
Text = "Root requirement for all system requirements."
«objective» TestPhasingSensors
Id = "O1.1"
Text = "Verify and test different phasing sensors and related phasing control algorithms"
«objective» ApplicabilityForELT
Id = "O1.3"
Text = "Determine scalability and applicability of the developed phasing algorithms and segmented telescope active control scheme for an ELT"
«objective» EvaluationEnvironment
Id = "O1.5"
Text = "The evaluation shall be carried out in the laboratory and on sky."
«StakeholderRequirement» CaptureRange
Id = "STKR1.3"
Text = "Determine the maximum segment error which can be measured and corrected by the phasing sensors, i.e. its capture range"
«WavefrontErrorRequirement» PWFS Wavefront Error
Id = "SR1.14"
Text = "The wavefront error of the output beam delivered to the PWFS shall be less than <wfe> nm RMS over the entire field of view."
«objective» Integration
Id = "O1.2"
Text = "Integrate phasing wavefront sensors into a global scheme of segmented telescope active control"
«objective» ImageQuality
Id = "O1.4"
Text = "Study the effect of phasing on the telescope image quality"
«trace»
Figure 4: APE objectives diagram

The following user defined types extend the SysML requirements modeling features to organize and trace objectives, stakeholder and system requirements of APE. In figure 4, *objective* is project-specific stereotypes specializing the SysML element requirement to capture the objectives for the projects (see figure 4); *stakeholder requirement* is a project-specific stereotype of a SysML requirement to capture the high-level requirements for the projects; and for the *system requirements*, SysML requirements are used to capture the detailed requirements for the projects.

Figure 5 shows the dependencies between the stakeholder requirements and the objectives in an automatically created traceability matrix.



	Stakeholder Requirem…	AtmosphericSensitivit…	AtmosphericTurbulenc…	CaptureRange [APE::…	ClosedLoopMeasurem…	DefocusControl [APE:…	DoubleSegmentation […	EdgeSensorSimulation…	EvaluationOfWaveFro…	FinalAccuracy [APE::…	GlobalAberrations [AP…	IntegrationOfWavefr…	LimitingStarMagnitude…	OpenLoopMeasureme…	OperationalTime [APE…	CalbrationTimeOffTel…	CalibrationTimeOnTel…	MaximumTelescopeCa…	MeasurementTime [A…	Report [APE::APE_Re…	ResidualPSF [APE::AP…	ScallopingEffect [APE:…	SegmentationErrors […	SensorPlatform [APE:…	SimultanousControl [A…	UT of the VLT [APE::A…	VerificationOnSky [AP…
Objectives [APE::APE_Require…	1	2	1	1	1	1	1	1	1	2	1	1	1	1		1	1	1	1	1	1	1	1	1	1		2
System Objectives [APE::A…	↙																										
System Objectives [APE::A…		2	1	1	1	1	1	1	1	2	1	1	1	1		1	1	1	1	1	1	1	1	1	1		2
ApplicabilityForELT [APE…							↙	↙		↙			↙			↙	↙	↙	↙	↙							
EvaluationEnvironment […		↙	↙																								↙
ImageQuality [APE::APE…		↙								↙											↙						↙
Integration [APE::APE_…											↙	↙													↙		
TestPhasingSensors [AP…				↙	↙	↙			↙					↙								↙	↙	↙			

Figure 5: Requirements/Objective traceability matrix

Guidelines for modeling the system requirements

- SysML requirements are not a replacement of requirements manangement (RM) tools but a visualization aid for architectural important requirements and a traceability bridge to the system analysis and architecture artifacts.
- Distinguish Objectives, User Requirements, System Requirements and Analysis elements (e.g. Use Cases)

- The objectives describe the major goals of the project. They are modeled with requirements elements, stereotyped objective.
- The user requirements describe the system from the perspective of the user, mostly technology independent. They are modeled with requirement elements, stereotyped as user requirements. User requirements are traced to objectives.
- Requirements shall be decomposed with the containment relationship (i.e. nesting) only on the same abstraction level.
- System requirements restate user requirements from a system (technical) perspective. Usually they assume a design decision which shall be justified by a rationale.
- Use the refine relationship to elaborate a requirement with another model element, like a use case refines a requirement.
- Do NOT use refine among requirements (where a requirement is a client and another requirement is a supplier).
- Use the derive relationship for (system) requirements that are discovered by some design or analysis effort, which do not restate a user requirement but rather apply to the next lower system level or next level of abstraction. They shall be justified by a rationale (e.g. Technical report) or trace relationship to an existing design
- Use the copy relationship to integrate existing common requirements, like interface requirements, international standards, policies etc. The common requirements are typically contained in a model library.
- Use the trace relationship to model the relationship between functional and non-functional requirements.
- Use the trace relationship between top-level containers of each level to indicate that there is some kind of relationship among their nested elements.

Context modeling

The system's context defines the system's boundaries and is modeled using SysML internal block diagrams. A SysML internal block diagram (IBD) shows an enclosing block, its parts, and its interfaces. For the system context our main focus is on system interfaces. The context itself is represented as a block in the model. The Observatory context owns the system under development (APE) and the connected elements like the telescope. The context element is defined in a BDD (see Figure 6). The interfaces and connections are shown in an internal block diagram of the Observatory context. This IBD is the context diagram of the system (Figure 7). The diagram header of the IBD shows the element type (stereotype «System context»), the enclosing block ObservatoryContext and the diagram name that mentions the aspect ObservatoryContext_Optical.

SysML uses ports to model the interfaces of a block. SysML provides a standard port for service-like interfaces and flow ports for item flows (often physical interfaces). However, there are different possibilities to use these ports for capturing system interfaces. You find more about the modeling of ports in Section III of this paper.

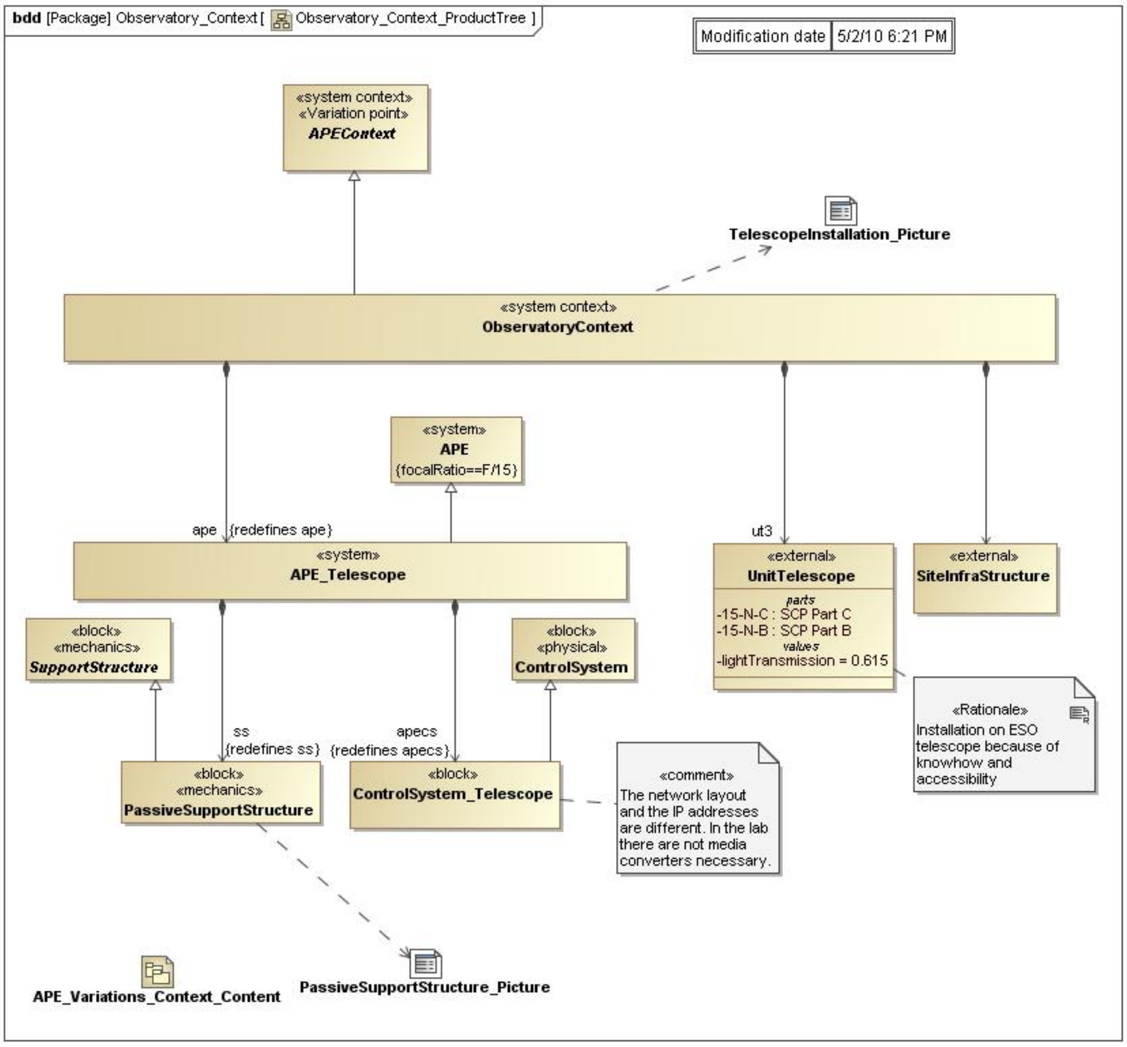

bdd [Package] Observatory_Context [Observatory_Context_ProductTree]
Modification date 5/2/10 6:21 PM
«system context» «Variation point» APEContext
TelescopeInstallation_Picture
«system context» ObservatoryContext
«system» APE {focalRatio==F/15}
ape {redefines ape}
ut3
«system» APE_Telescope
«external» UnitTelescope
parts
-15-N-C : SCP Part C
-15-N-B : SCP Part B
values
-lightTransmission = 0.615
«external» SiteInfraStructure
«block» «mechanics» SupportStructure
«block» «physical» ControlSystem
ss {redefines ss}
apecs {redefines apecs}
«block» «mechanics» PassiveSupportStructure
«block» ControlSystem_Telescope
«comment» The network layout and the IP addresses are different. In the lab there are not media converters necessary.
«Rationale» Installation on ESO telescope because of knowhow and accessibility
APE_Variations_Context_Content
PassiveSupportStructure_Picture

Figure 6: APE Observatory Context definition

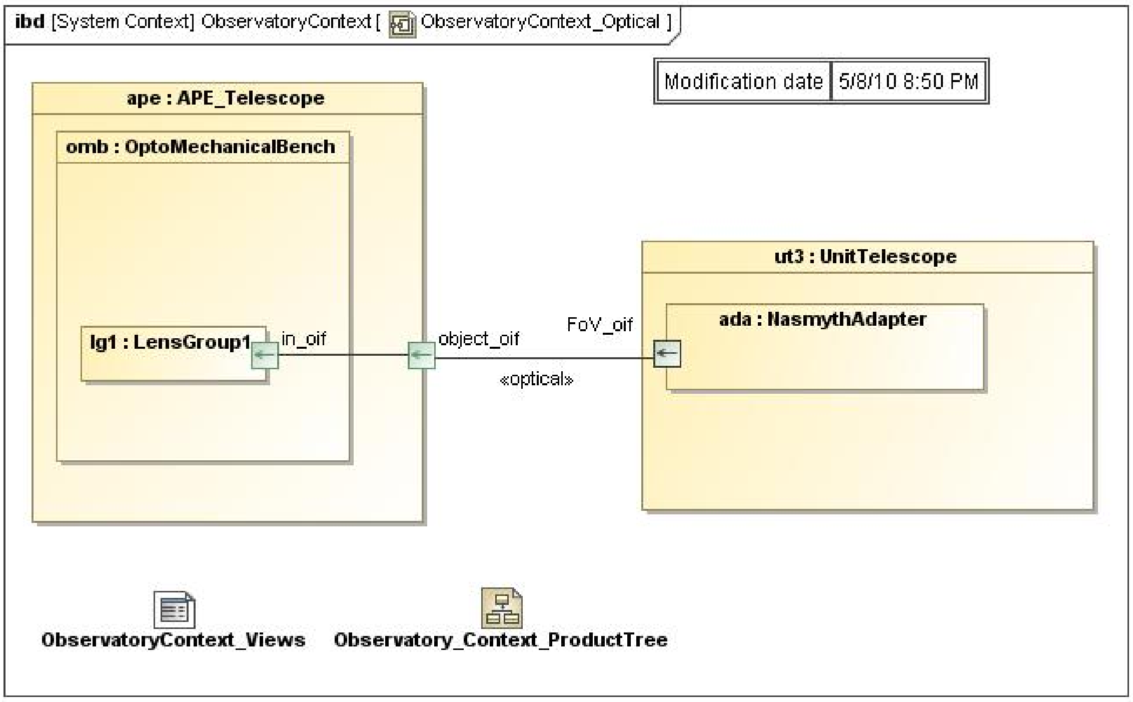

ibd [System Context] ObservatoryContext [ObservatoryContext_Optical]
Modification date 5/8/10 8:50 PM
ape : APE_Telescope
omb : OptoMechanicalBench
lg1 : LensGroup1
in_oif
object_oif
FoV_oif
«optical»
ut3 : UnitTelescope
ada : NasmythAdapter
ObservatoryContext_Views
Observatory_Context_ProductTree

Figure 7: APE Observatory Context diagram (optical aspect only)

System Structure

We have used SysML block definition diagrams (BDDs) to model the product tree and internal block diagrams (IBDs) to model the structure and interfaces of APE and its subsystem. The APE hierarchical breakdown is based on the product tree. It has several levels, going from the highest level into more and more details, using decomposition of its elements.

On the other hand, a complex system has much more than just one internal structure. There are multiple views showing electrical, optical, and mechanical elements that are interconnected, and therefore multiple structures exist. The same components can be connected in different views in different ways. We have used IBDs to show the electrical, optical, information and mechanical layout of APE and its components at different levels. Figure 8 shows the optical layout in an abstract manner. The connectors are stereotyped as *optical*, but they are elided for readability.

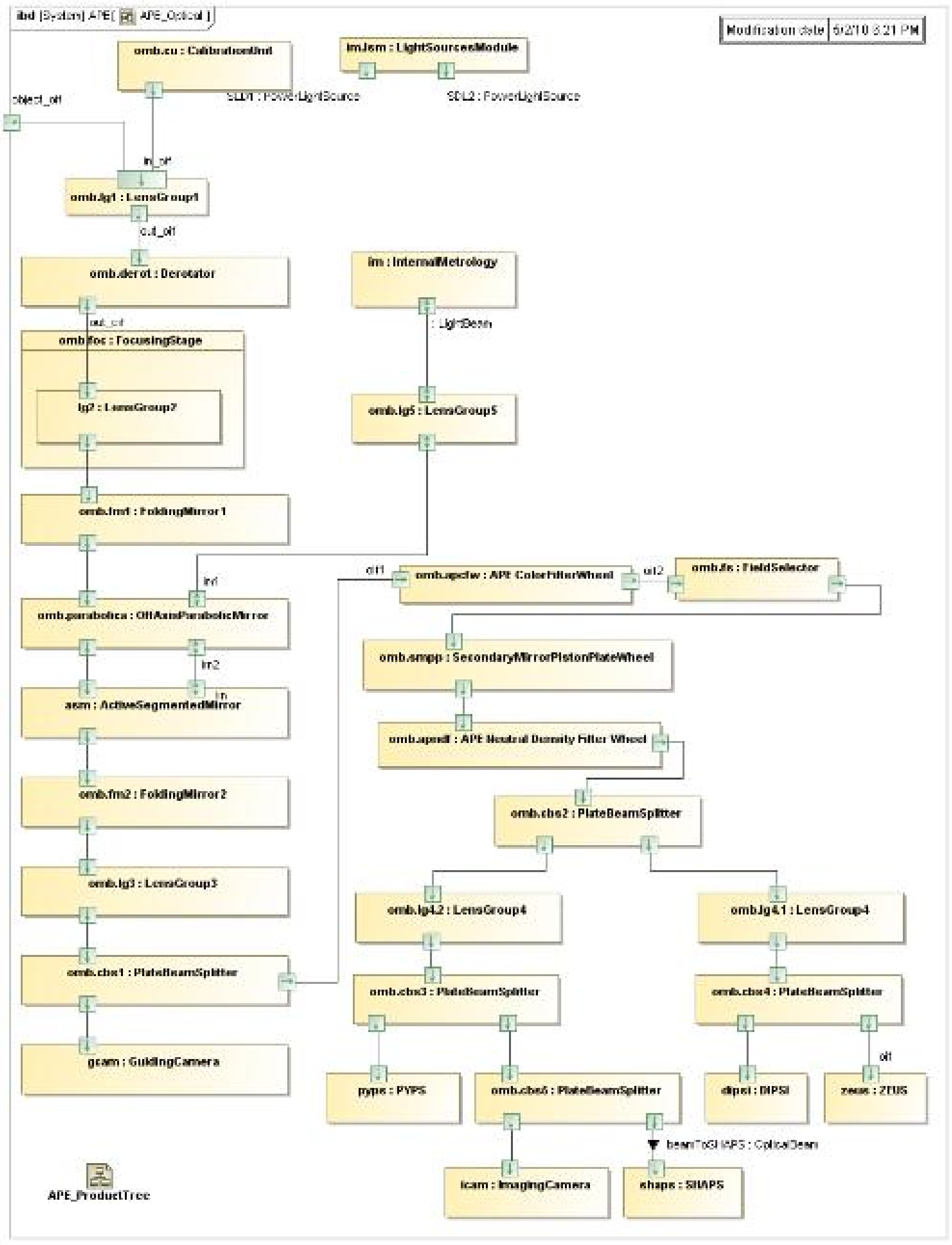
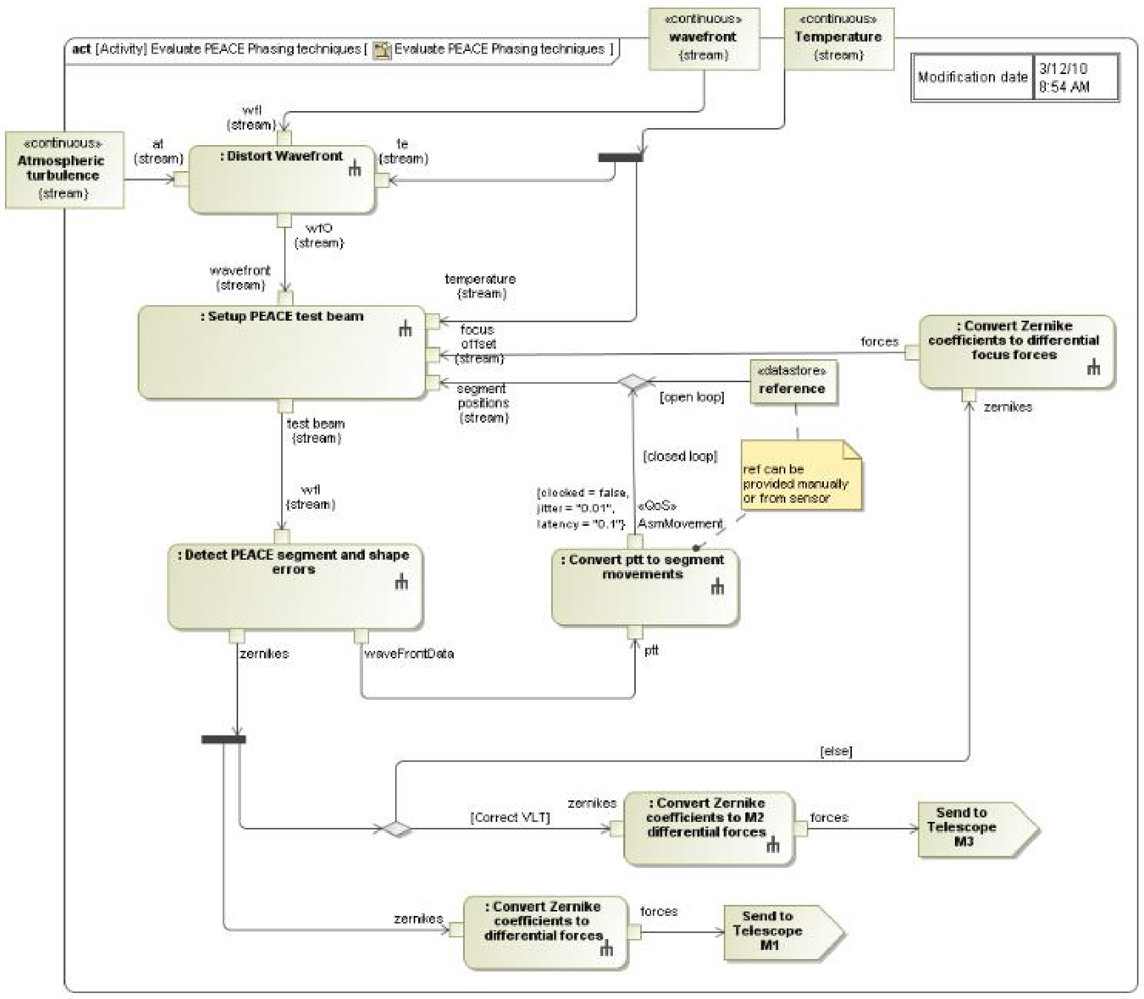


Figure 8: APE structure (optical aspect only)

Behavior

A complex system is much more than just the collection of its elements and their structural architecture, because its behavior derives from the collaboration of its parts. Therefore it is essential to capture the behavior of the system to be able to understand it. We have used activities to model the behavior of APE and its subsystems. SysML activity diagrams can be used to show the actions taken by the system and its data and control flow. Figure 8 shows the evaluation of the phasing technique of APE.



act [Activity] Evaluate PEACE Phasing techniques [Evaluate PEACE Phasing techniques]
«continuous» wavefront {stream}
«continuous» Temperature {stream}
Modification date 3/12/10 8:54 AM
«continuous» Atmospheric turbulence {stream}
at (stream)
wfI (stream)
te (stream)
: Distort Wavefront
wfO (stream)
wavefront (stream)
temperature (stream)
: Setup PEACE test beam
focus offset (stream)
segment positions (stream)
forces
: Convert Zernike coefficients to differential focus forces
zernikes
«datastore» reference
[open loop]
[closed loop]
ref can be provided manually or from sensor
test beam (stream)
wfI (stream)
{clocked = false, jitter = "0.01", latency = "0.1"}
«QoS» AsmMovement
: Detect PEACE segment and shape errors
: Convert ptt to segment movements
zernikes
waveFrontData
ptt
[else]
zernikes
[Correct VLT]
: Convert Zernike coefficients to M2 differential forces
forces
Send to Telescope M3
zernikes
: Convert Zernike coefficients to differential forces
forces
Send to Telescope M1

Figure 9: Activity diagram for APE

Data

Another aspect is the information or data handled by the system. SysML provides block definition diagrams for the definition of data, and it provides internal block diagrams, activities, and sequence diagrams for data usage and flow. APE uses SysML value types to define the data of APE and its subsystems. Figure 10 shows the composition structure of a measurement and its relation to movements of the segmented mirror ("AsmMovement"). Those data types are used to define ports in IBDs and objects in activity diagrams.

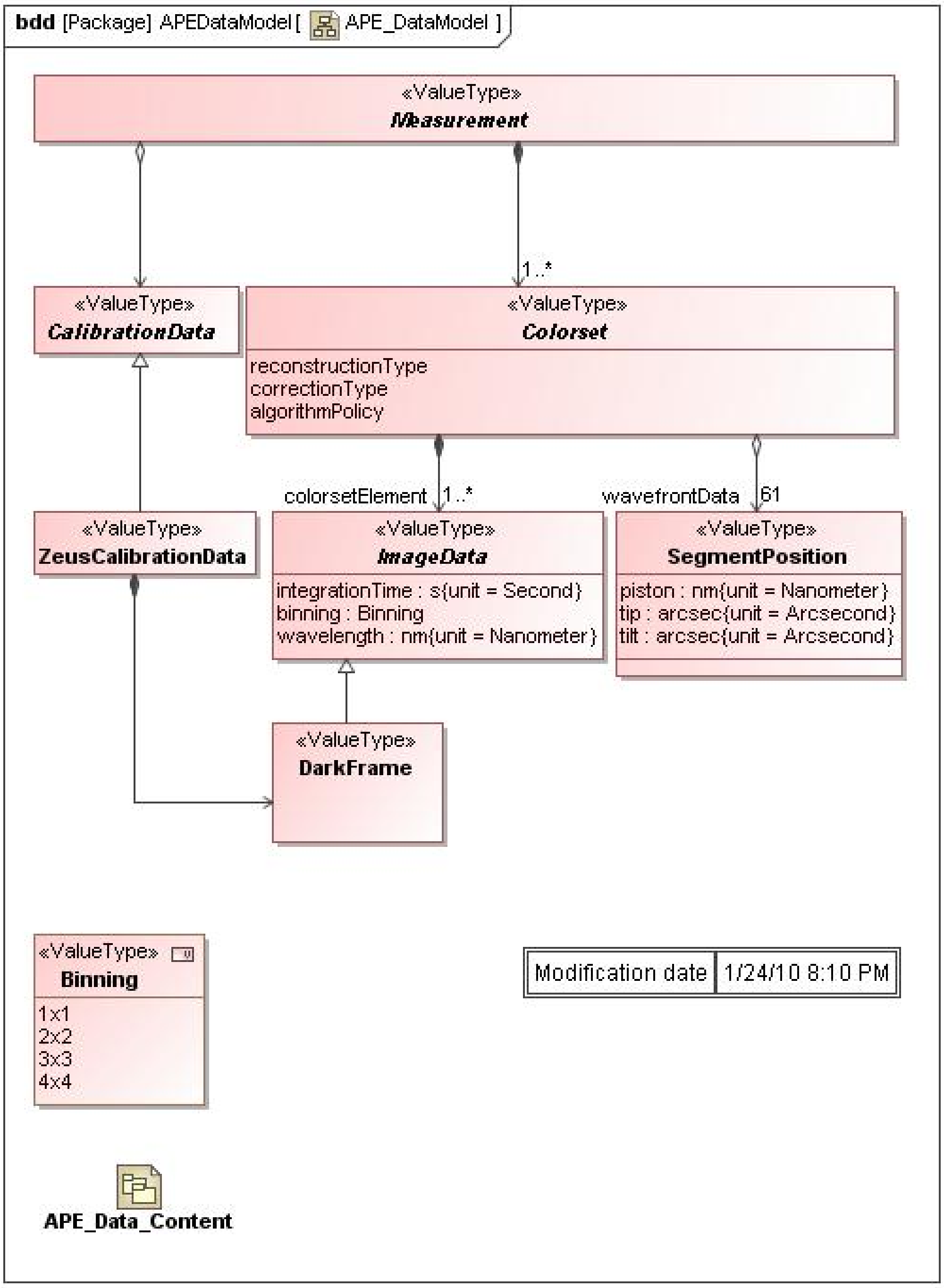


Figure 10: APE Data model

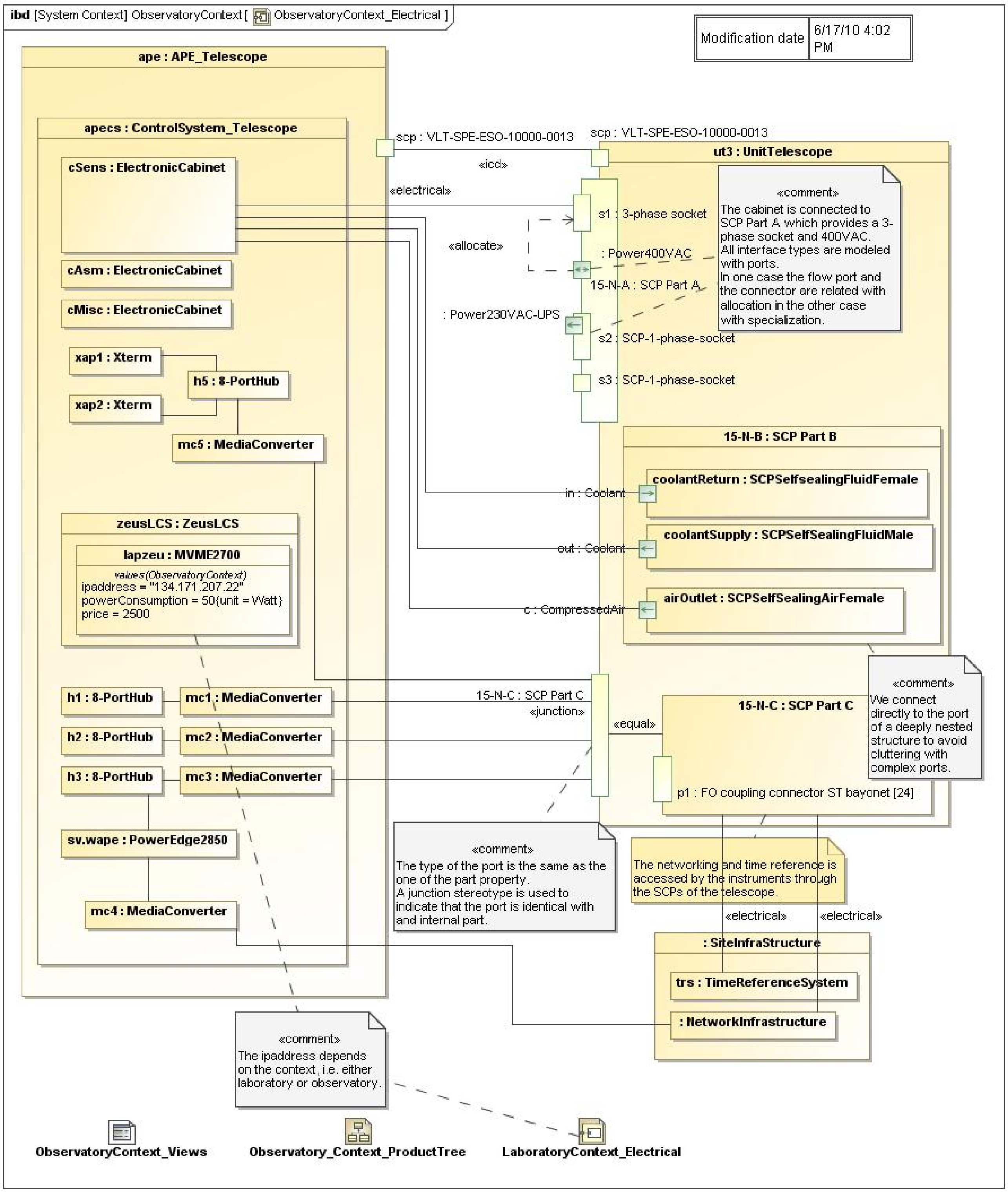
Section III: MBSE in Telescope Modeling for OCSMP Model Builder Intermediate

Context

Section II generally introduces the topic of context modeling. Now we focus on the interface modeling with ports. We use ports to specify the interaction points of the system with it's environment. Figure 11 shows the context of the telescope from an electrical viewpoint. On the left side is the APE system and on the right side the telescope as one of the main actors of the system.

There are different possibilities to use ports:

- Use standard ports to model abstract interfaces as representation of an interface control document (ICD) as shown for port "scp" at the top of the parts in figure 11. The type of the ports represents the ICD. To ensure consistency between the interface control document and the model, the latter serves as the basis for the former.
- Use standard ports to model mechanical interfaces as shown for port "s1" below the "scp" port at the telescope part in figure 11. The type of the port specifies the structure of the interface.
- Use standard ports for a combination of mechanical and flow ports as shown for port "15-N-A" just below the "scp" port at the telescope part in figure 11. The type of the port is a block that owns the mechanical and flow ports.
- Model mechanical and flow ports directly at the specific part, as shown for the ports "in" and "out" at the parts "coolantReturn" and "coolantSupply" in figure 11, crossing the border of the outer part, if you always show the internal parts.
- Use junction ports (standard port with stereotype «junction») as a proxy for internal parts as shown for port "15-N-C" at the bottom of the telescope part in figure 11. That allows to hide the internal parts without loosing the information about the connection to parts at the outside of the enclosing part.



«comment»
The cabinet is connected to SCP Part A which provides a 3-phase socket and 400VAC.
All interface types are modeled with ports.
In one case the flow port and the connector are related with allocation in the other case with specialization.

«comment»
We connect directly to the port of a deeply nested structure to avoid cluttering with complex ports.

«comment»
The type of the port is the same as the one of the part property.
A junction stereotype is used to indicate that the port is identical with and internal part.

The networking and time reference is accessed by the instruments through the SCPs of the telescope.

«comment»
The ipaddress depends on the context, i.e. either laboratory or observatory.

Figure 11. Internal block diagram of electrical context

Model Library and Systems-Engineering Profile

Besides the modeling of the different aspects, the APE modeling project provides a model library and an SE^2 profile. Data types and model elements that are frequently used are modeled in a model library to increase reuse. Abstract types are used as place holder for specific building blocks. They are classified in different catalogue packages (figure 12).

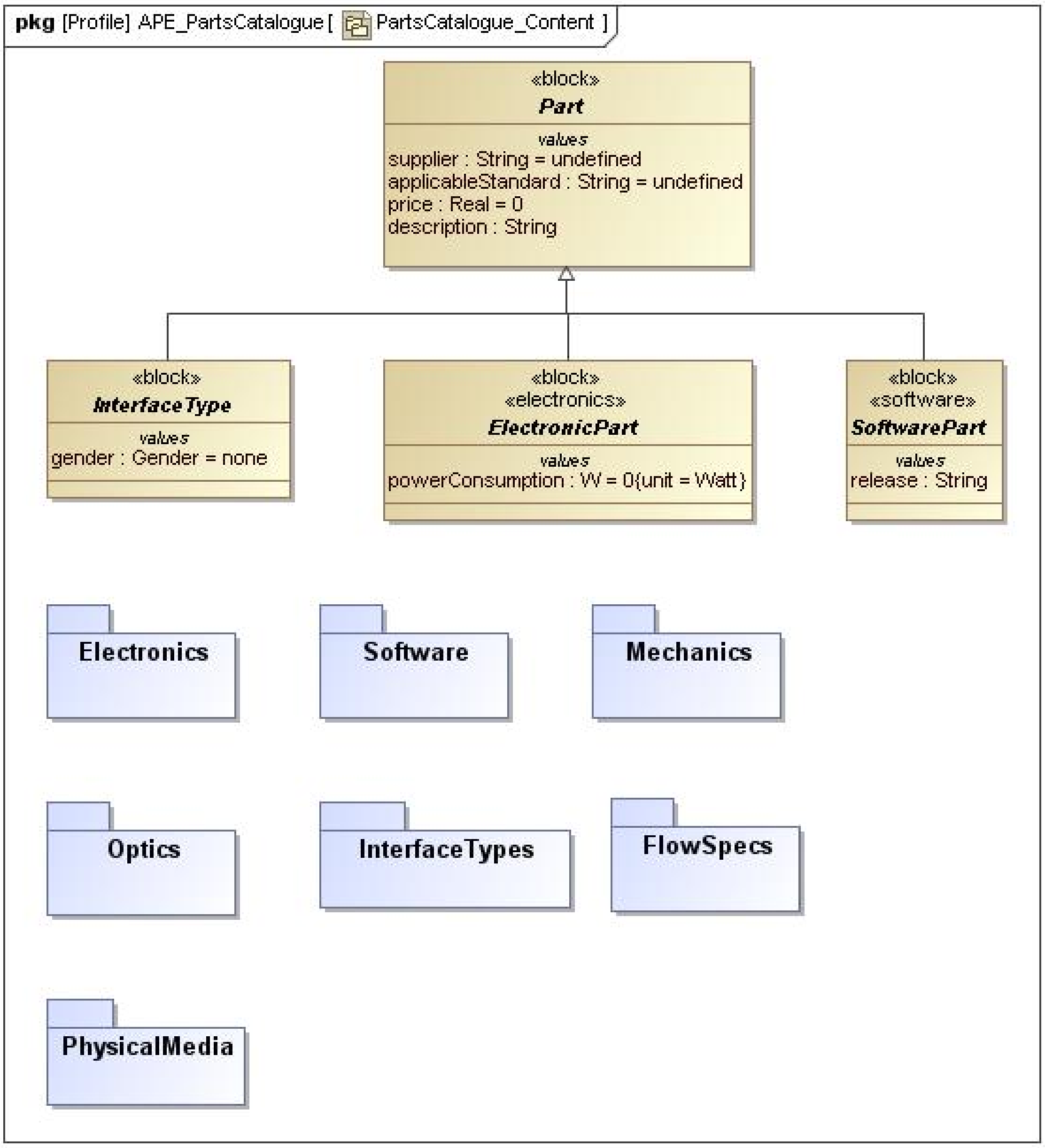


Figure 12. APE abstract types

Catalogues can be easily extended by using inheritance. Furthermore, the preliminary design of a system can initially work with an abstract type (when the detailed requirements are yet unknown) and decide later which specific type to use for the implementation. A generic connector gets a different context-specific pin assignment by inheritance. For each specific assignment a separate specialization is needed. The SE^2 profile provides the project-specific extensions to SysML, i.e. stereotypes like «objective», «junction» or «mechanical».

Modeling Challenges

SysML is a relatively new language. This creates two inherent challenges: Is SysML sufficiently mature for real projects, and is it accepted by a wide range of systems engineers? Especially the fact that SysML is based on UML sheds a special light on these challenges. Could a modeling language that was initially defined for software development be used to model systems, and will systems engineers accept a language with origins in the software discipline? An overall result of our project is that this question can be answered yes.

The APE project is a pretty good challenge for SysML. It is complex and interdisciplinary without a special focus on software; it is a real system and not the simplified coffee machine so often used as demonstration project. Although we found that SysML is practicable to model complex systems, we have made a list of the language's shortcomings. The most significant ones are these:

- Variant modeling
- Connection of nested blocks
- Grouping of interfaces with nested ports
- Logical vs. physical decomposition
- Functional multilayer abstraction
- Reuse of blocks, allocation, and instances
- Structural multilayer allocation
- Defining quality of service
- Transition to UML for software
- Configuration and quality control
- Navigability

There are four aspects related to these:

- **Notation**: It is a real challenge for a modeling language to provide an interdisciplinary notation for complex systems. It must be easy to understand and be capable of modeling details unambiguously.
- **Model**: Behind the notation is the real model, i.e., the data structure and semantics of the information.
- **Tool**: The implementation of the SysML specification is a challenge for tool vendors.
- **Methodology**: SysML is a language without any methodology. You need a methodology or at least some best practices for good modeling.

Structured Model Queries

Models provide a good deal more than just a set figures. Models are stored in repositories with a defined data structure. Like databases the model repository makes it possible to query the model for specific information, e.g. an impact analysis when changing a requirement. SysML doesn't define a query language. Most modeling tools allow to write a script to query the model. We've implemented queries to compute the price and the power consumption of the system. We plan to implement more queries like "are all actions allocated to parts?", "are all requirements satisfied?", and so on.

Verification

For every large system, verification is an essential part of the system acceptance in order to prove that the system meets its requirements. SysML supports the modeling of test cases with a specific model element, "testCase." The "testCase" element can be used at different levels for component integration, software integration, and system integration. Furthermore, APE has two different test contexts: verification in the lab and in the sky.

Tool

SysML has some modeling areas where creating and editing model information in a diagram is cumbersome. For example for requirements, which are often entered in a bulk, it is easier to use a table than a diagram to enter or modify the data. The same is true for relationship modeling like the allocation of behavioral elements to structural system elements. It is easier to assign the relationships in a matrix than in a diagram with lots of arrows. SysML already defines table and matrix formats. Most tools provide them as limited read-only views on the model. What we need are table and matrix views with the ability to create and modify model elements.

Configuration and Quality 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 corrupt the model. This can only be mitigated by establishing rigorous configuration-management practices and using tools that allow rollbacks.

Acknowledgements

Many thanks to Sanford Friedenthal for reviewing this article and giving us valuable feedback.

References

ESO (European Southern Observatory). 2009. Very large telescope information. *ESO* (Web site). http://www.eso.org/public/astronomy/teles-instr/paranal.html (accessed 3 Nov. 2009).

Friedenthal, S., A. Moore, and R. Steiner. 2008. *A Practical Guide to SysML: The Systems Modeling Language.* Burlington, MA: Elsevier/Morgan Kaufmann.

Gonte, F. Y. J., N. Yaitskova, P. Dierickx, R. Karban, A. Courteville, A. Schumacher, N. Devaney et al. 2004. APE: A breadboard to evaluate new phasing technologies for a future European Giant Optical Telescope. In *Ground-based Telescopes*, ed. Jacobus M. Oschmann, Jr. (vol. 5489 of *Proceedings of SPIE* [The International Society for Optical Engineering]), 1184–1191. Bellingham, WA: SPIE.

Ogren, I. 2000. On principles for model-based systems engineering. *Systems Engineering* 3 (1): 38–49.

OMG (Object Management Group). 2008. *OMG Systems Modeling Language (OMG SysML™): version 1.1.* OMG document no. formal/08-11-02. http://www.omg.org/spec/SysML/1.1/ (accessed 3 Nov. 2009).

Weilkiens, T. 2008. *Systems engineering with SysML/UML: Modeling, analysis, design.* Amsterdam: Morgan Kaufmann OMG Press/Elsevier.

Wymore, A. Wayne 1993, *Model-Based Systems Engineering*, CRC Press.