Three years of MBSE for a large scientific programme: Report from the Trenches of Telescope Modeling

Robert Karban, Michele Zamparelli, Bertrand Bauvir, Gianluca Chiozzi
European Southern Observatory
Karl-Schwarzschildstr. 2
85748 Garching b. München, Germany
Robert.Karban@eso.org

Abstract. The most ambitious projects of the European Southern Observatory's (ESO) is the construction of the European Extremely Large Telescope (E-ELT) which will be by far the world's largest optical and near-infrared telescope, and. will provide images 15 times sharper than those from the Hubble Space Telescope. Such a project poses continuous challenges to systems engineering due to its complexity in terms of requirements, operational modes, long operational lifetime, interfaces, and number of components. Since 2008 the Telescope Control System (TCS) team has adopted a number of Model Based Systems Engineering (MBSE) practices in order to cope with the various challenges ahead. This paper provides an overview of the different approaches we took during this time - which ones worked, and which did not.

About ESO

ESO is an prominent astronomy organization and carries out an ambitious programme focused on the design, construction and operation of observing facilities. ESO operates three observing sites in Chile: Paranal, La Silla and Chajnantor and has its Headquarters in Garching bei München, Germany. The two major projects of ESO during the last 15 years were the Very Large Telescope (VLT) and the Atacama Large Milimeter Array (ALMA).
The ALMA Project [10] is a global partnership between the scientific communities of East Asia, Europe and North America with Chile. It comprises an array of 66 12-metre and 7-metre diameter antennas observing at millimetre and submillimetre wavelengths. Its construction started in 1998 and in October 2011 Early Science Operations began with about 15 antennas (Figure 2) at the Chajnantor site at 5000 m.s.l. The completion of the array is expected in 2012.

Figure 1. VLT

Figure 2. ALMA

The VLT [4] (Figure 1) is a visible-light astronomical observatory and consists of an array of four telescopes, each with a main mirror of 8.2-m diameter, that can observe together or individually and several smaller telescopes dedicated to interferometry, making it the largest

facility of its kind. The construction of the VLT started in 1988 and it is fully operational at the Paranal Observatory at 2600 m.s.l. since the year 1999.
ESO is currently engaged in design studies for an extremely large optical/near-infrared telescope, the European Extremely Large Telescope (E-ELT) [1]. In Europe, extremely large telescopes are considered one of the highest priorities in ground-based astronomy. The E-ELT (Figure 3) with its diameter of 39.3 meters is poised to become the world's largest optical and near-infrared telescope with a novel adaptive 5-mirror design.

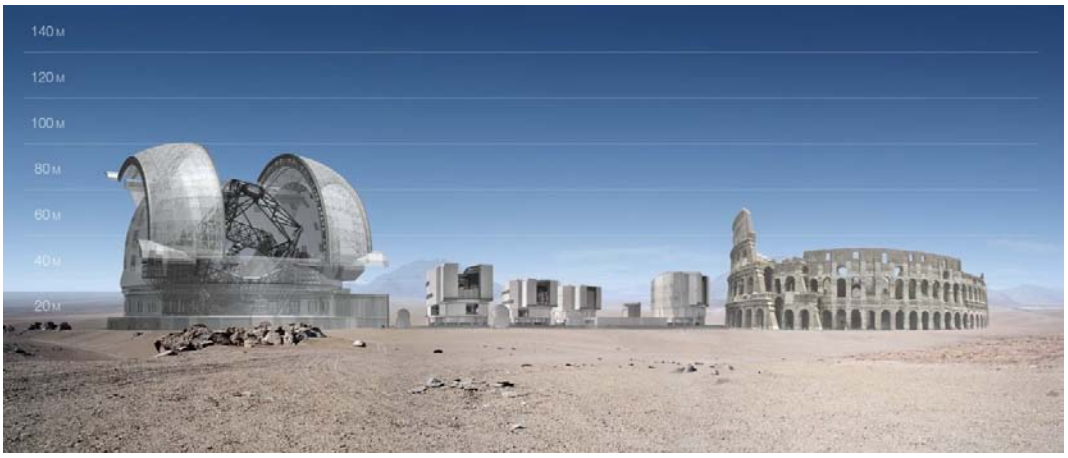


Figure 3. The European Extremely Large Telescope in Comparison

The telescope and its dome will have the size of a small football stadium being about 80m high and 80m wide.

Challenges for Control Systems

Control Systems for observing facilities (consisting of telescopes and instruments) are embedded in a data flow, where their central role is to execute observing blocks (which define targets and boundary conditions) and produce scientifically-relevant raw data. Within the E-ELT, the TCS will maintain wave front quality throughout the duration of the observation. The Instrument Control System (ICS) is responsible for acquiring the scientific data, using the TCS as a service provider.
The TCS includes all hardware, software, and communication infrastructure required to control the telescope (including the dome), and to interface to subsystems down to, but not including, actuators and sensors, as they are typically delivered together with the respective sub-system. The TCS provides access to the opto-mechanical components, manages and coordinates system resources (subsystems, sensors, actuators, etc…), and performs fault detection and recovery. The Control System basically produces a raw image which is then reduced to a scientific image. When building Control Systems for large Science Machines like Telescopes we have to face a number of challenges. First of all, Telescopes and their instruments are interdisciplinary, software intensive **scientific facilities** (mechatronic systems with a lot of glass), with long operational life-times between 10 and 50 years.
Those systems are one-of-a-kind experimental machines with many opto-mechanical components that had never been built before (e.g. nanometer accuracy position actuators, very low noise CCDs), and pushing state-of-the-art technology. The jump in technology between any two generations (typically 15 years apart) is very big and it is difficult foresee what will be possible to achieve and how the machines will be eventually used. Despite careful design and product assurance it can be safely expected that some components will not fully meet their requirements, their behavior is sometimes different than specified, or operational concepts were misunderstood or even wrong

Therefore, despite the fact that those systems are at some point handed over to science operations, they are never frozen and **evolve sometimes substantially over their lifetime**, since unexpected scientific program tend to require new functions to be provided. During commissioning (i.e. making the system work to specifications) and operation we are often learning how the machine actually works. The control systems design needs to build-in the capability to cope with changes.
Most of the major **sub-systems (e.g. primary mirror, main structure) are contracted out** to industry together with their local, subsystem specific control system, which requires later on the integration of heterogeneous systems, as ESO is usually solely responsible for carrying out all system integration. It also means that acceptance of sub-system will be done on separate not-fully validated sub-systems
The control system interacts with a mixture of COTS and highly specialized, custom built actuators and sensors (e.g. wave front sensors, position actuators).
After handing over the system to operations there is a coexisting development and operational environment (due to required upgrades of software and hardware), with very **limited access to the systems for engineering tasks** during operations, and telescope subsystems and instruments running on different releases or even different versions of hardware and software infrastructure. This requires release and product line management.
In past decade the **border between hardware and software development responsibilities has become more and more blurred,** as software has infiltrated to a large extend into hardware development. Therefore the products cannot be designed and built independently as they influence each other mutually and an optimal balance is to be found. In particular interfaces cannot be conceived independently. Integrated and simultaneous engineering has to take place with the goal of designing an overall integrated system
With the increase in size and required performance for modern observatories there is also an increase in distributed control (i.e. different nodes, actuators and sensors geographically distributed across the telescope, higher performance), and an increase in high performance computing (e.g. for adaptive optics, complexity of algorithms). Last but not least, safety functions have to be implemented which protect humans and the system itself from hazardous situations.

Systems Engineering for Complex Systems: some issues

There are five main problem areas, as identified by NASA’s Jet Propulsion Laboratory (JPL) [17], which are common to many projects which build large complex systems:

1. Mission complexity is growing faster than our ability to manage it
   …increasing mission risk from inadequate specification & incomplete verification
2. System design emerges from the pieces, not from an architecture
   …resulting in systems which are brittle, difficult to test, and complex and expensive to operate.
3. Knowledge is lost at project lifecycle phase boundaries
   …increasing development cost and risk of late discovery of design problems.
4. Knowledge and investment are lost between projects
   …increasing cost and risk; damping the potential for true product lines
5. Technical and programmatic sides of projects are poorly coupled
   …hampering effective project decision-making; increasing development risk.

These problems are becoming evident in various parts of a system but we will consider only the control system in this paper.
The higher the essential complexity (the complexity associated with the underlying problem) of a system is, the wider impact the above mentioned problem areas have. It is imperative to

avoid adding any more complexity than necessary and the incidental (what extraneous complexity is added) complexity needs to be addressed and controlled.
The eventual goal is to have a correct-by-construction development process, where a model is used to reason about the proposed solution (formally prove properties of interest through inference e.g. checking design consistency by evaluating the model against constraints), ensuring that all required functionality will be delivered and the correct behavior exhibited, by formally defining and constraining the design space (e.g. like a software engineer bases code development on state machines only). Testing is still performed, but its role is to validate the correct-by-construction process rather than to find bugs.
System engineering's need for a system model for validating and verifying the system design before actual construction, is not unlike the need for CAD models and shop drawings for mechanical components, or for a detailed SW/ HW interface specification, exposing exactly the information needed for a component integration with the rest of the system.
A large part of the above can be enforced, supported, and checked with a Model, and what we consider to be **the pillars of MBSE**:

- Ontologies**,** Domain Specific Languages
- Methodology, System Modeling Language
- Model Transformation, Model Validation
- Modeling recipes, best practices, and patterns
- Tools

Modeling activities and a comprehensive system model

A system like the E-ELT requires a vast modeling effort (in particular dynamic modeling), with thousands of opto-mechanical components and very dynamic characteristics to be controlled in order to deliver the required functionality and performance, The static and dynamic performance has to be assessed with FE-models consisting of about 19000 nodes and 38000 elements. Different load cases must be analyzed like: gravity, buckling, wind, thermal gradients, and earthquake scenarios. The static behavior under wind-load at different angles of the elevation axis must be analyzed for maximum operation wind-speeds, and analogically the analysis of thermal loads is carried out.
During day-time the telescope must be kept in a climate controlled environment, the dome (which has also other functions like protection of the telescope from its environment), to compensate differences between day and night, and avoid temperature induced variations in the opto-mechanical components. There are models to analyze the air conditioning performance, to analyze wind buffeting and dome seeing (using CFD Models, wind tunnel tests, thermal models).
More models exist to analyze the propagation of smoke in case of fire, the structural behavior during an earthquake, or the displacement of actuators. Models are also required to carry out trade-off analyses for the control structure and the optimal location for actuators and sensors. Other models are required to define FDIR strategies, for example the impact of a failing sensor on the primary mirror. Those models are typically derived from FE-models and extended with models for the control system action. The example of broken sensor above is the result of the model for a control system function, modeled by combining fairly advanced models of the system under control with models of the control system and infer requirements, performance, etc.
Dynamic simulations are required to define the control strategies in order to deliver the required wave front quality. Control engineers create numerous models to make design trade-off analyses, analyze control performance, for requirements elicitation for actuators, sensors and control system, carry out failure and hazard analyses, and verify contracted design of control systems.

On top of all those model there are uncountable artifacts which represent system models in one way or another, like DOORS requirements data bases, Excel sheets, optical designs, project plans, and not the least thousands of emails with unstructured information. All this information is related to different aspects of the system: the functional-behavioral aspect, performance aspects, structural/component aspect, and analysis aspects

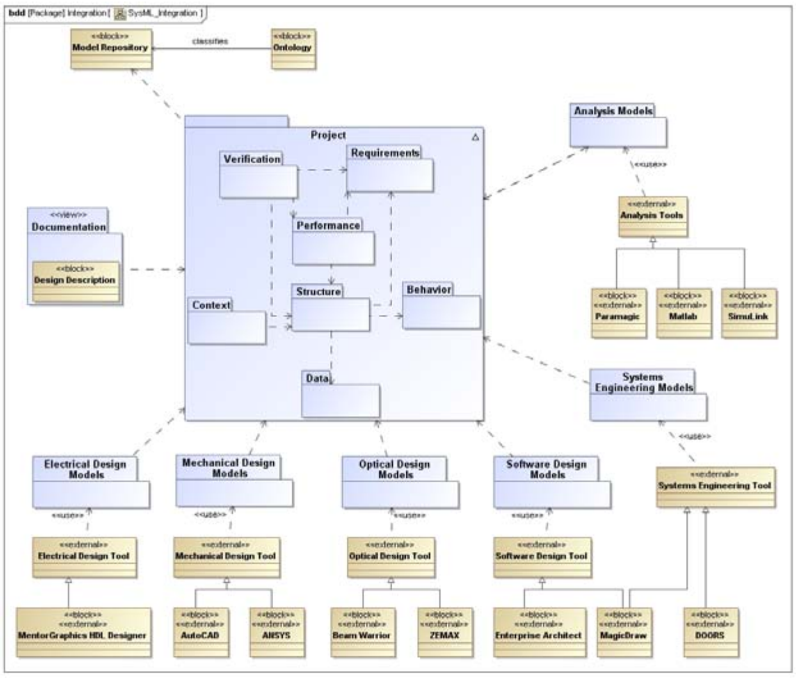

bdd [Package] Integration [SysML_Integration]
«block» Model Repository
classifies
«block» Ontology
Analysis Models
Project
Verification
Requirements
Performance
Structure
Behavior
Context
Data
«view» Documentation
«block» Design Description
«use»
«external» Analysis Tools
«block» «external» Paramagic
«block» «external» Matlab
«block» «external» SimuLink
Systems Engineering Models
«use»
«external» Systems Engineering Tool
Electrical Design Models
Mechanical Design Models
Optical Design Models
Software Design Models
«use»
«use»
«use»
«use»
«external» Electrical Design Tool
«external» Mechanical Design Tool
«external» Optical Design Tool
«external» Software Design Tool
«block» «external» MentorGraphics HDL Designer
«block» «external» AutoCAD
«block» «external» ANSYS
«block» «external» Beam Warrior
«block» «external» ZEMAX
«block» «external» Enterprise Architect
«block» «external» MagicDraw
«block» «external» DOORS

Figure 4 Integrated System Model

This information is related, has to be kept synchronized and needs to be validated and verified. Since models are already used in most engineering disciplines it is natural to request the same at system level and create an integrated system model which addresses the multiple aspects of a system. The individual engineering models do not cease to exist, but the elements which are relevant at system level, are represented in a system level model in order to specify and design their mutual dependencies. The system model is the centerpiece of MBSE (Figure 4) where integration takes place at two levels:

- First, the integration of the different disciplines at system level, where each system relevant (has an impact at system level across disciplines) component is represented in the system model in order to trace its relationship and impact to other system components.
- Second, the integration of different engineering specific models, and the exchange of information amongst those models (e.g. exchange information between CAD models, Optical Models, FEA models, and Matlab models which are relevant at system level).

There are a number of requirements for building interdisciplinary systems. Consistency and Correctness is required across disciplines, for analysis, design and the resulting documentation, for the integration of a multiplicity of system aspects, views and models for mechatronic systems, and for the reuse across projects. By formalizing the artifacts, we can reduce and control uncertainty, and maintain system integrity. Formalization in this context means to use defined semantics (e.g. with a language like SysML [14]) to capture the knowledge, which might have been created on a whiteboard or in an e-mail conversation. The construction of large systems usually involves different countries or even continents; therefore it is necessary to reduce ambiguities in particular in cross-cultural, cross-language projects (e.g. outsourcing activities), and natural language. For long lived operational system, which furthermore evolve over time, it is essential to provide to system maintenance teams the necessary understanding of how the parts of the system contribute to its mission.

The Telescope Modeling Challenge Team

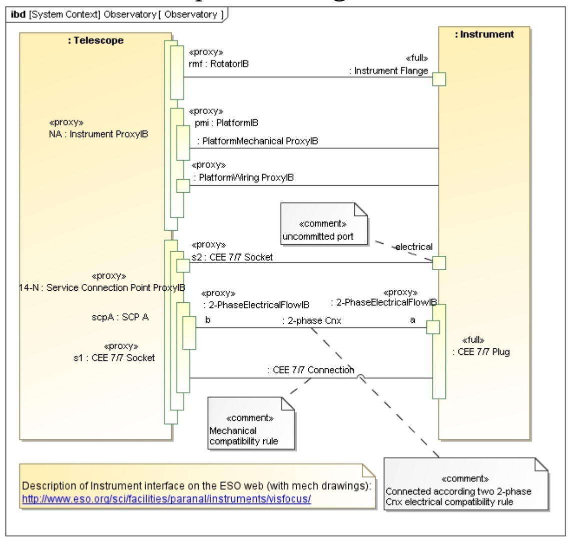
One of INCOSE's strategic initiatives addresses the area of MBSE [16]. It is considered one of the main means to tackle problems of current and future systems development. In this framework ESO is collaborating with the German Chapter of INCOSE (GfSE) forming a "MBSE Challenge Team". The task is to demonstrate solutions to "challenging", non-trivial, real-world problems using MBSE. The SE^2 Telescope Modeling challenge team was founded in 2007 [15] with the following goals:

- Provide examples of SysML, common modeling problems and approaches.
- Build a comprehensive model, which serves as the basis for providing different views for different engineering aspects and associated activities.

The system case study is a technology demonstrator for future Extremely Large Telescope (ELT), called the Active Phasing Experiment (APE), which is a high-tech interdisciplinary opto-mechatronical system. The activities included the creation of modeling guidelines and conventions for all system aspects, hierarchy levels, and views, and of the corresponding fully fledged SysML model, which can be also considered as an educational model because it shows the real system parts and the corresponding model elements. The main results are:

- APE model, guidelines (domain-specific and general, currently mixed) and best practices. Examples including naming conventions, application of appropriate model annotations, proper use of modeling constructs, and model reuse considerations.
- Model, Model libraries, and Profiles; Customization of the modeling tool
- Input for tool vendor and SysML Revision Task Force of the OMG

The following figures show cookbook examples (already based on SysML 1.3) of the telescope domain for interface modeling. Figure 5 shows some of the interfaces between the telescope and an instrument. The SCP (Service Connection Point) which provides services as power, network and liquid cooling are detailed in Figure 6, when looking inside the black box.


ibd [System Context] Observatory [Observatory]
: Telescope
: Instrument
«proxy» rmf : RotatorIB
«full» : Instrument Flange
«proxy» NA : Instrument ProxyIB
«proxy» pmi : PlatformIB
: PlatformMechanical ProxyIB
«proxy» : PlatformWiring ProxyIB
«comment» uncommitted port
electrical
«proxy» s2 : CEE 7/7 Socket
«proxy» 14-N : Service Connection Point ProxyIB
scpA : SCP A
«proxy» s1 : CEE 7/7 Socket
«proxy» : 2-PhaseElectricalFlowIB
«proxy» : 2-PhaseElectricalFlowIB
b
: 2-phase Cnx
a
«full» : CEE 7/7 Plug
: CEE 7/7 Connection
«comment» Mechanical compatibility rule
Description of Instrument interface on the ESO web (with mech drawings): http://www.eso.org/sci/facilities/paranal/instruments/visfocus/
«comment» Connected according two 2-phase Cnx electrical compatibility rule

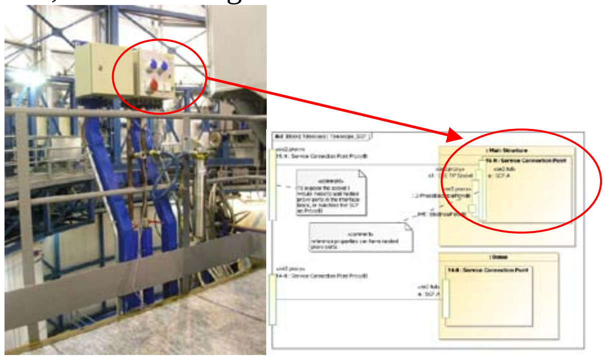


Figure 6 Telescope's Service Connection Point (SCP)

Figure 5 Black box view of telescope

Although, most practices collected in the cookbook can be considered domain independent, there is nevertheless a substantial amount which is domain specific and could be re-used in several of our modeling activities. All the work to write the cookbook was well invested because it enabled us to apply MBSE right from the start in the E-ELT. Most of the initial mistakes in modeling had already been made and could be avoided.

The results have been presented at various occasions and culminated in the publishing of the "**Cookbook for MBSE with SysML**" which provides modeling patterns, recipes, guidelines, and best practices for the application of SysML. Other relevant outcomes from the challenge team work are:

- The need to provide SysML profiles which focus on the guidance for domain-specific activities.
- The need to provide OWL2 Ontologies which focus on domain-specific semantic.
- The adherence to modeling guidelines and standards requires assessment by automatic validation

The lesson learned is that an organization should create its own, domain specific cookbook when introducing MBSE. Domain independent recipes and practices can be collected in a commonly shared cookbook.

Telescope design evolution: from VLT to E-ELT

The evolution of telescopes creates new, and domain specific challenges. When comparing the last big project of ESO, the VLT, with the next major project, the E-ELT the difference is size and complexity becomes apparent.
The VLT consists of about 600 tons of steel and glass, 200 actuators and 3 mirrors, 2000 I/O points, a small data volume, and some interacting, distributed control loops (0.01Hz up to 50Hz). The wave front control strategy (Figure 9) implemented by the control system is dominated by fixed client server relationships, basically 2 distributed control loops at moderate frequencies, and an easily-manageable number of components and system states
The E-ELT (Figure 7) consists of 10000 tons of steel and glass distributed over 15 subsystems and 9 focal stations, 20000 actuators and 1000 mirrors, 50000 I/O points, (the primary mirror has alone 15000), large engineering data volume (700Gflops/s, 17Gbyte/s). Hundreds of interacting, distributed control loops (0.01Hz up to kHz rates) among many loosely coupled interacting components, which cannot be forced into a traditional hierarchy with multiple layers, require a software intensive distributed control strategy. Since the overall performance and functional control requirements of the entire system are assigned to the TCS, and its development requires an interaction of Control Engineering, Software Engineering and Electrical Engineering – it becomes a truly interdisciplinary effort. The long lag between contract set-up and development adds additional complication.

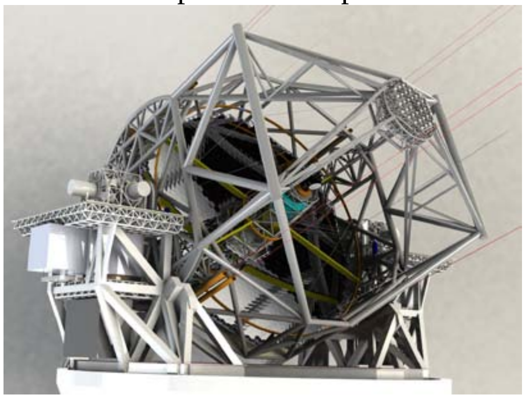
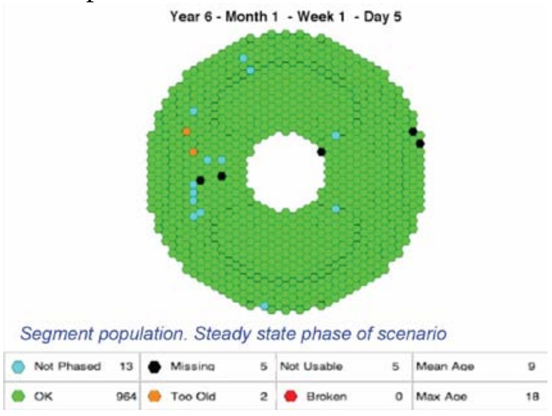


Figure 7 Rendering of the E-ELT structure



●	Not Phased	13	●	Missing	5	Not Usable	5	Mean Age	9
●	OK	964	●	Too Old	2	● Broken	0	Max Age	18

Figure 8 Thousands of possible configurations for the primary mirror

When we started to adopt MBSE practices for the E-ELT in 2008 [11], one of the first things we did was to model something we know well – it was the wave front control of the VLT. We learned what we wanted to model and how to model it, before we started modeling the E-ELT.

When comparing Figure 9 and Figure 10 (the content of the figures is not relevant to the reader), it becomes evident that the essential complexity of the system has increased. It is the outcome of model queries regarding the number of interactions, activities or interfaces which can more objectively confirm this impression.

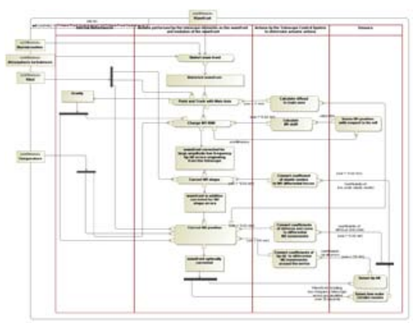


Figure 9 VLT wave front control strategy

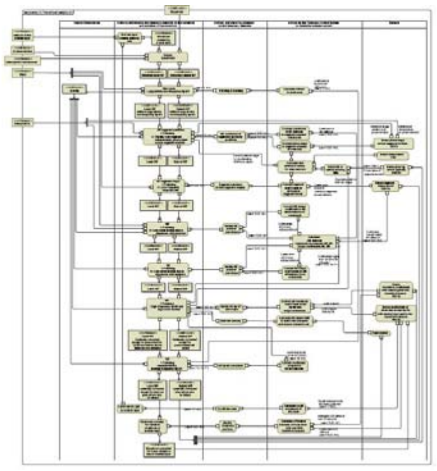


Figure 10 One of many E-ELT control strategies

This effort was based on results that had been achieved by the SE2 Telescope Modeling Challenge Team [15]. This initial effort, which can be considered significant, provided sufficient confidence that MBSE application from scratch to a real project was possible.
The rendering of the current E-ELT design (Figure 7) shows many of its components. It is important to realize that its secondary mirror is almost as big as the primary of the current largest telescopes.
The primary mirror of the E-ELT (M1) is one of the most demanding subsystems. M1 consists of about 800 hexagonally shaped segments because such a large mirror cannot be built in a single piece. The position of the segments must be coordinated to deliver a continuous surface with an error below 100nm RMS across 40m. 2400 actuators and 5000 sensors must work in a several hundred Hertz closed loop to meet this requirement. Moreover, 10000 actuators (12 motors per segment, the warping harness) are responsible for deforming each individual segment in order to correct aberrations.

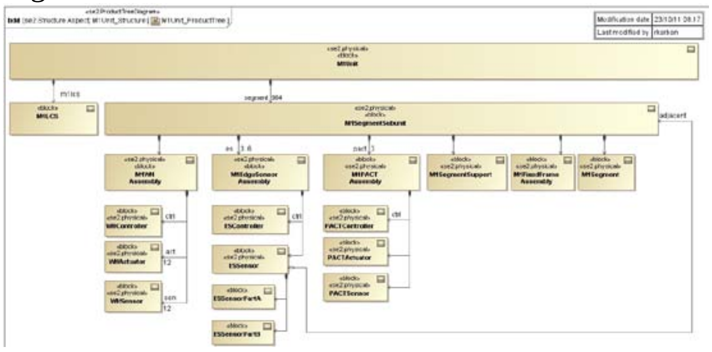


Figure 11 Product Tree of M1 in SysML

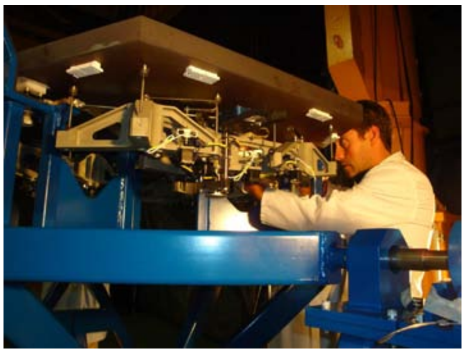


Figure 12 Prototype Segment

Given the amount of components, the control strategy must be flexible and deliver its function while accommodating component failure. Due to the sheer number there is, statistically, always one or more failing, which may cause the mirror to lose its shape within seconds if not caught and recovered and also the loss of the corresponding observation.
The segments will be constantly replaced (because of re-coating), and there will be always segments not working, broken, or not available (Figure 8). The control system has to cope with

all those different mirror configurations. Figure 12 shows a prototype of a single segment with its warping harness and the corresponding a Block Definition Diagram of M1 (Figure 11) specified with the System Modeling Language (SysML).
The main functions provided by the TCS are related to Control and Interlock (an automatic and immediate stop of a unit in case of a safety critical situation) handling, and fault management.

The Upgrade of the VLT

As part of validating the technological decisions for the E-ELT, control systems at the VLT are being refurbished using technologies intended for use in the E-ELT control systems. The upgrades serve several purposes: field testing technologies and methods in an operational environment (outside the lab), providing input into the E-ELT technology decisions, addressing obsolescence in the VLT telescope control systems, and preparing observatory technical staff for the construction of the E-ELT [8].
As the first step in the upgrade and field test program, the Enclosure (dome) Control System (ECS) of one Unit Telescope (UT) was refurbished in 2010, using the State Analysis methodology (see below) and architecture. It is now successfully operating at the Paranal observatory.
During 2011 and 2012, the project will be working to upgrade the control system of the main axes. The main axes upgrade applies more rigorously the State Analysis methodology than the enclosure upgrade, together with other MBSE concepts, as they are introduced gradually and not in a “Big Bang” The overall system model is built according to principles of the Object Oriented Systems Engineering Method.

Early Adopters of MBSE at ESO

In the year 2008 we had two years of preliminary design phase in front of us, with many tasks to be carried out, and a number of artifacts and deliverables to be produced for the TCS, which consisted of:

- Define infrastructure (e.g. network infrastructure)
- Derive power budget and cost estimate from equipment catalogue
- Enforce common standards through catalogue and design guidelines
- Define requirements and ICDs for contracted subsystems (e.g. data rates, data volume, latency)
- Build a consistent information model of TCS properties to manage its size
- Provide a (architectural) design which satisfies telescope functions (e.g. wave front control strategies)
- Produce a consistent design and Correct-by-Construction requirement
- Generate Documentation, Software, System Configuration, and Deployment

The defined tasks in the above scenario seemed to provide the ideal test bed for MBSE to prove its virtue, and deliver the required items in time and with high quality. At the same time those points define the scope of what we want to achieve by system modeling upfront. It allowed determining when modeling is complete.
For the construction proposal delivered at the end of 2010, there were several deliverables based on information from a common project model (more details can be found in [18]):

- Integration of Wave front Control Strategies into the overall control strategy
- Telescope Control System Design Description (Architecture)
- Design Conventions (control system meta-architectures)
- Local Control System Reference Design Description for Suppliers
- Interface Control Documents

- Cost Estimate (e.g. capturing complexity in activity diagrams)

The main dogma of our modeling efforts was to strive for a pragmatic and practical approach, and avoid ending up modeling for the model's sake. One important decision to take was to define what you want to get as a result out of the modeling thus providing termination criteria to modeling activities.

However, this has turned out to be more difficult than thought. In 2007, the main problem was that there was basically no material on the application of SysML available, apart from the specification. Meanwhile there are several books on the market and the also the Cookbook [15] is widely available, but it took until 2009. Although there are several methodologies available [7], there are only few which are genuinely targeted to systems engineering, whereas some have been derived from existing software development processes. In both cases, the available documentation, and in particular examples of their application are still scarce, in particular with respect to large systems where scalability becomes a major issue. A significant effort was spent in finding appropriate means to organize large models, defining the necessary views, and scaling the selected methodologies. Omitting this step would have likely led to an excessive risk of an inadequate model right in the middle of the project, possibly with an intractable maze of model elements. Since MBSE is supposed to be formal, it also turned out that a better defined semantic of SysML is required. This has been achieved by defining domain specific semantics, and by providing feedback to the specification of SysML (in particular related to interface modeling) which resulted in SysML 1.3. During all this work, the definition of appropriate Ontologies has become extremely important, in order to define formally, what you really mean when modeling something. Needless to say, also the technical infrastructure (like tool chains) revealed itself of utmost importance to support MBSE in an organization.

In order to leverage more on system modeling, documents are auto-generated from the model, using a Model Based Document Generation (MBDG) technique, in which the document is modeled in the same model as the system, using SysML elements. A subset of the DocBook [6] markup language is mapped to stereotypes, and applied to UML/SysML model elements, e.g. chapter to package, paragraph to comment (the implementation is still currently being brought from prototype to production quality). This allows a consistent integration of system model and system documentation, and direct linking to model elements (also diagrams) from the document, among other things like tool independency. Some documentation artifacts for the VLT upgrade were produced according to this technique.

There has also been an effort to integrate text requirements more formally into the system model to have better verification of them [12], and enable re-use of typical requirement patterns. However, it has not been used in practice because the gain has not outweighed the effort, in particular because requirements are managed in a separate requirements management tool which is not well integrated with the modeling tool.

MBSE Methodology

In the beginning the system model focused almost only on the physical and behavioral aspects. The model was constructed according to guidelines [15] developed by the Telescope Modeling challenge team, which concentrated mostly on properly using SysML and organizing a large model without following a particular method. The resulting model served its purpose for the construction proposal but it became very clear that a systematic approach was required to manage different phases of development, functional and physical models, and in particular support the design of control system for complex dynamic systems. The guidelines needed to be augmented with two methods of proven worthiness:

The **State Analysis** (SA) methodology [2] [3] is targeted to the control related domain, and focuses on behavior. It has been developed by NASA's Jet Propulsion Laboratory (JPL) since 1990. The methodology, which is founded on a state-based architecture and goal-based

operation, defines a process for identifying and modeling the states of the physical system and their relationships. State Analysis provides also a uniform, methodical, and rigorous approach for developing control system architecture. However, SA does not support SysML, and is not integrated into an overall system modeling process. Therefore, the first goal has been to develop, a SysML profile [9] for SA in collaboration with the JPL.
The **Object Oriented Systems Engineering Method** (OOSEM) [5] integrates top-down functional decomposition with a model-based approach that uses SysML to support the specification, analysis, design, and verification of systems. OOSEM is intended to ease integration with object-oriented software development, hardware development, and test. It encourages use of OO models to capture system and component behavioral, performance, and physical characteristics that provide the basis for integrating other specific engineering models. From both methodologies we apply the practices which we think provide most of the gain. The methods (in particular OOSEM) have been slightly adapted to our domain specific needs to be most effective [18]. The lesson learned is that a method shall be applied if and only if it provides structured guidance in the maze of MBSE. It is important to realize the necessity of different levels of abstraction, their relation and how to effectively use them.
The main benefit is seen in managing properly the evolution of systems at functional and physical level (Figure 13) with several generations of physical implementations and functional architectures.

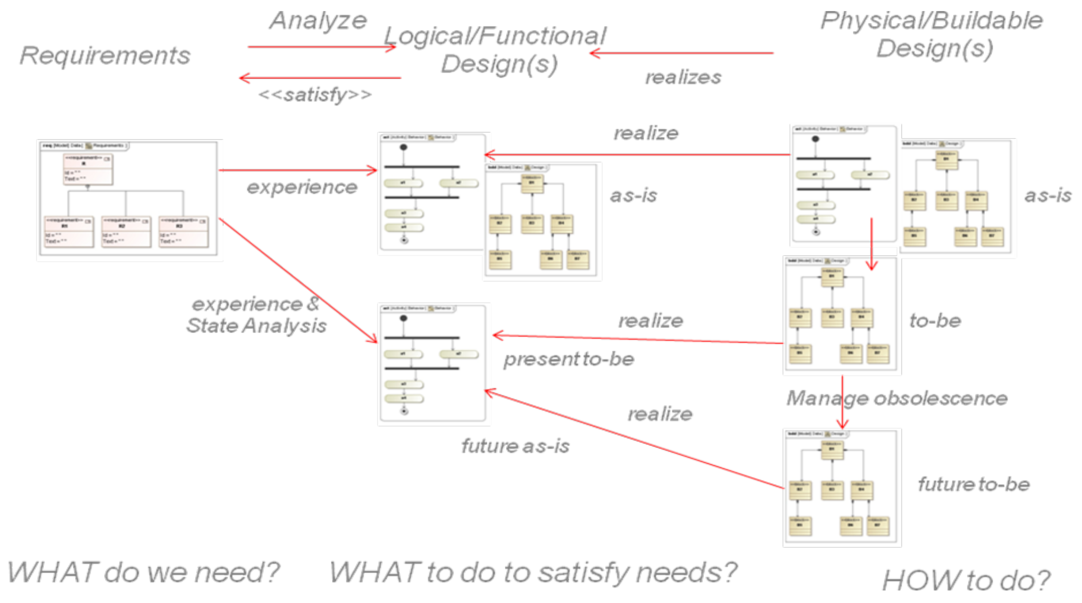


Figure 13 Relation of Functional vs Physical Models

In the past, the functional architecture was heavily based on previous experience and implemented without systematic approach to identify logical components and their mapping to technology, resulting sometimes in a tight coupling between function and implementation. This becomes in particular a problem in obsolescence management when the same functional architecture is implemented with different technologies, and the functional architecture shall be maintained. When doing such an upgrade of technology, it is important to define the transition of the current system to the future system, and how to integrate legacy components – at logical and physical level.

Domain Specific Modeling Languages (DSML)

In order to have a more consistent, validated design across the complete system, and follow common guidelines and standards, several domain-specific modeling concepts have been

semi-formalized and defined. DSMLs serve in constraining the solution space for system design by expressing a pre-defined architecture. The architectural concepts are formally defined in an ontology and mapped to SysML stereotypes and concrete system architectures are built according to the DSML.
A DSML was created for the TCS (which has many subsystems) and for Instruments (of which many will be built). Common SysML profiles and parts catalogs for telescope and instrument help the modeler in carrying out the concrete design. Common design elements are grouped in a parts catalog accessible inside/outside of ESO which contains the public interface of each reusable element for Hardware and Software components, avoiding the creation and maintenance costs of identical design elements.

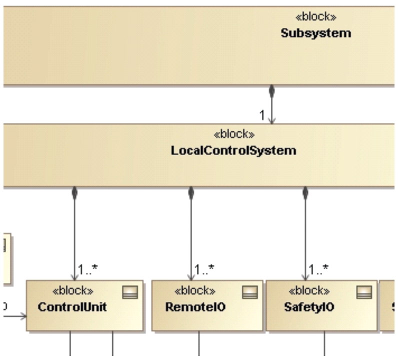


Figure 14 Telescope Control System DSML

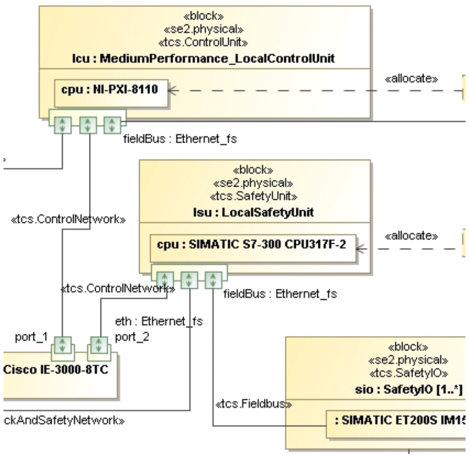


Figure 15 Concrete TCS Design

Reference models are created which describe the design of an example system using the DSML and the parts catalog, reducing the initial learning curve, and represent the main test case of the parts catalog and the DSML.
Figure 14 shows a subset of the TCS DSML while Figure 15 shows the reference model for a concrete design. Each part of this design is typed with a stereotype defined in the ontology of the TCS to clearly describe its compliance with the meta-architecture.

Model Driven Development

In order to maximize the benefits we strive to create several related development artifacts (documents, code, test cases, simulators, etc.), using model transformation, from the same model and thus move gradually to a model based engineering approach.
Model transformations (languages) play a very important role while model simulation is used for rapid prototyping and analysis of the dynamic behavior of groups of applications. Model checkers allow the formal verification of properties of the system and the design of test cases using the traces of the model checker. An important step has been to start generating documentation from the model which ensures consistent documents (no need to cross-check all the time deliverables as ICDs, requirements and design documents) and building up an infrastructure for model execution and code generation [13] with the final goal to have an integrated system model which allows us to specify structure, behavior, generate code, and deployment configuration.

MBSE Status at ESO

At the end of 2011, the status can be summarized as follows. The control system team of the E-ELT delivered end of 2010 the construction proposal which consists of a completed preliminary design. The associated review was passed. The goals of enforcing systematic architecture rules correct-by- construction and a consistent verifiable system design are gradually achieved by adopting a formal system modeling language, and adopting a common system model. A significant step was achieved by base-lining JPL's State Analysis Method and Architecture, and the continuous collaboration with JPL on a SysML profile for State Analysis. The adoption of OOSEM concepts is helping in a more systematic approach to properly use SysML, capture the required information, and organize it properly. The field testing at the VLT successfully started in 2010 and continued in 2011 by deploying and validating E-ELT technological choices, and applying State Analysis and OOSEM in an operational environment, refurbishing at the same time parts of the control system of the VLT.

Conclusions. This paper reports on the way we have come from adopting just a systems modeling language in the beginning until discovering that there are more pillars required to build the foundations for applying MBSE practices to a real-world project. The paper points out that there is an enormous amount of modeling done for such a project. Yet, those modeling activities live often separate lives, despite the fact that it would be easier to handle this enormous amount of information in a more structured way by integrating it. It is very important to realize that there is more to modeling than just drawing diagrams. The model adds value because it can be validated, queried, reasoned about, and used to create documentation and operational artifacts like software. It is easy to fall into the trap of putting semantics into diagrams: the semantics must be instead fully defined by the model behind in order to create more artifacts from the same source in an automated way: this requires an expensive initial effort which pays back many more times later on.
The application of appropriate MBSE methodologies for the problem domain (like SA and OOSEM) is valuable in defining common concepts and procedures, and keeping the resulting artifacts consistent in a common project model. More time should be spent on properly defining how to best use existing methodologies, which therefore appear to cost much more time and resources than necessary. Time must be set aside to build up an MBSE-friendly infrastructure; otherwise MBSE practices will also be considered a burden, because they are not well defined.
The confidence to be able to cope with the challenges that building control systems for ground based astronomical facilities involve, in particular for such systems like the E-ELT, has risen by adopting methods like JPL's State Analysis, and combining it with principles of OOSEM, for the design and specification of an entire hardware/software system. State Analysis has added substantial value to the analysis, specifying behavior, and architectural design of the control system. Nevertheless, there is still a long way to go to efficiently use those technologies in a day to day engineering life. It requires a significant investment in supporting infrastructure, to get out of an experimental phase and move into a well defined engineering era. It is important though to keep the focus on the product and to distinguish efforts to advance the state of practice and state of the art, where you push the limits and can get distracted by finding problems in the tools, language specifications, and documentation. System modeling with SysML relies on voluntary discipline of the modeler in a context where complete validation of the modeling artifacts is made impossible by the weak semantics of the OMG specifications. This semantic weakness is overcome by tool vendors for their own modeling languages and products, but comes at the price of de-facto vendor lock-in.
The OMG specifications are not perfect and we have to accept that they evolve and this requires engagement from the modeling community. There is a lack of tool conformance to specifications and the possibility of exchanging models. Additionally, sound ontologies (general and domains specific) must be created for better semantics to support model transformation, checking, and validation.

Making a case for MBSE's adoption real benefits has been very difficult, and engineers' general tendency to favor short term gains with tangible artifacts, to more methodological, structured, and constrained approaches, has not made it easier.
Control System and Software engineers have a quasi natural inclination to think in abstract terms, and are keen on seeing the added benefits of a model based approach. Beyond the TCS boundaries this becomes more challenging and time consuming, hence its restriction to the TCS. Additional benefits (like simulation, validation model transformation, reuse of design elements, and generation of multiple artifacts) must be demonstrated, as nice consistent diagrams for communications are not sufficient. In particular, model transformation allows using different capabilities of different tools, starting from the same model.
Future work will mainly concentrate on cutting a vertical slice from system level analysis and design to code generation for control systems. In particular, on model based simulation of State Analysis models defined with SysML, their transformation to production code, and deploying it in current and future operational systems. In general, more working products based on models have to be delivered to better demonstrate gains in correctness, consistency, time, money.

Acknowledgements

Special thanks go to Chi Lin, Chris Delp, Daniel Dvorak, Mitch Ingham, David Wagner, Nick Rouquette, and Steven Jenkins from JPL who made it possible to collaborate at a technical and programmatic level, and enabled us to adopt the State Analysis Method for the E-ELT.
Many thanks to Sandy Friedenthal, who has always been available for discussions, questions and guidance, in particular related to the application of OOSEM. Nerijus Jankevicius, from NoMagic, helped substantially in promptly solving problems at tool level, related our modeling tool Magic Draw.
And last but not least, many thanks to the Telescope Modeling Challenge Team, consisting of Tim Weilkiens, Rudi Hauber, Andreas Peukert, Rainer Diekmann, and Andreas Hein, who made it possible to publish the "Cookbook for Modeling with SysML".

References

[1] Gilmozzi R., Spyromilio J., 2008, "The 42m European ELT: status", Proc. SPIE 7012, 701219
[2] Choi J., Coleman A., Dvorak D., Hutcherson J., Ingham M., Lee C.Y., Wolgast P., Feb 2008, "Goal-Based Operations of an Antenna Array for Deep Space Communication", iSAIRAS. Los Angeles, CA.
[3] Ingham M., Rasmussen R., Bennett M., Moncada A., Dec. 2005, "Engineering Complex Embedded Systems with State Analysis and the Mission Data System", AIAA Journal of Aerospace Computing, Information and Communication, Vol. 2, No. 12, pp-507-536
[4] Wirenstrand K., 2003, "VLT telescope control software: status, development, and lessons learned", Proc. SPIE 2003, vol. 4837, p. 965
[5] Friedenthal S, Moore A., Steiner R., 2009, "A Practical Guide to SysML", Morgan Kaufmann OMG Press,
[6] Docbook 5 standard, http://www.docbook.org/
[7] Estefan J., 2008, "Survey of Model-Based Systems Engineering (MBSE) Methodologies," Rev. B, INCOSE Technical Publication, INCOSE-TD-2007-003-012008
[8] Karban, R., Kornweibel, N., Dvorak, D., Ingham, M., Wagner, D., 2011, "Towards a State Based Control Architecture for Large Telescopes: Laying a Foundation at the VLT", 13th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPS), Grenoble, France

[9] Wagner D., Bennett M., Karban R., 2012, “An Ontology For State Analysis: Formalizing the Mapping to SysML”, IEEE Aerospace Conference, Big Sky, MT., USA
[10] ESO, 2007, “ALMA, Exploring the Universe at Milimetre Wavelengths”, http://www.eso.org/sci/facilities/alma/publications/Brochure-2007.pdf
[11] Karban R., Zamparelli M., Bauvir B., Koehler B., Noethe L., Balestra A., 2008, “Exploring Model Based Engineering for Large Telescopes – Getting started with descriptive models”, Proc. SPIE 7017, 70171I (2008)
[12] Karban R., Andolfato L., Zamparelli M., 2009, “Towards Model Re-usability for the Development of Telescope Control Systems”, 12th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPS), Kobe, Japan
[13] Andolfato L., Chiozzi G., Migliorini N., Morales C., 2011, “A Platform Independent Framework for Statecharts Code Generation”, 13th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPS), Grenoble, France
[14] OMG Systems Modeling Language (OMG SysML), v1.2, formal/2010-06-01, http://www.omgsysml.org, OMG (2010).
[15] SE^2 Telescope Modeling Challenge Team, “Cookbook for MBSE with SysML”, http://mbse.gfse.de
[16] INCOSE’s MBSE Initiative, http://www.omgwiki.org/MBSE/
[17] Integrated Model-Centric Engineering (IMCE) Workshop for JEO, Jan. 2011
[18] Karban R., Zamparelli M. Bauvir B., Chiozzi G., 2012, “Practical Application of MBSE Methodologies for Large Telescopes”, INCOSE IS12

Biographies

Robert Karban is a Software Engineer at ESO, developing distributed real-time control systems for telescopes since 1996. His current main task is the modeling and development of the control system of the E-ELT, as its system architect. Robert Karban received his M.S. in computer science in 1990 from the Technical University of Vienna, Austria.	**Michele Zamparelli** has worked since 1998 at the European Southern Observatory as software integrator for the VLT and ALMA projects. His tasks include configuration control, Quality Control, automated inspection, build systems, and tool support. He has been involved with standardization and process improvement activities.
Bertrand Bauvir has been working on control system for large medical and scientific facilities since 1998. He joined ESO in 2001 to participate to the design and deployment of the VLT Interferometer in Chile; and later to participate to the E-ELT, leading the Control System design effort. He received his M.Sc. in Electrical Engineering from the department of Electrical Engineering and Computer Science of the University of Liège, Belgium.	**Gianluca Chiozzi** works at the European Southern Observatory since 1994 as a Software Engineer. For the last 15 years he has been heavily involved in the design and implementation of the Telescope Control Software for the VLT and ALMA projects and now for the E-ELT project. He is head of the Control and Instrumentation Software Department. Before ESO he worked at the IBM Technical and Scientific Research Center in Milan.