TOWARDS MODEL RE-USABILITY FOR THE DEVELOPMENT OF TELESCOPE CONTROL SYSTEMS

R. Karban[#], L. Andolfato, M. Zamparelli, ESO, Munich, Germany

Abstract

During the life-cycle of a project, the development team often discovers similarities among system components within the same or previous projects that can be reused, with the potential for cost and time savings. The overall development effort can be dramatically reduced by reusing the different modelling artefacts like core requirements, functional design, and structural design. This paper illustrates an approach, using the Active Phasing Experiment (APE) project as a case-study, where modelling with SysML is used as a key technique to identify common system properties. APE is an opto-mechatronic technology demonstrator which was installed on the Very Large Telescope (VLT) at the Paranal observatory.

INTRODUCTION

There are many levels of model re-use. The following chapters concentrate on requirements, system constraints, and system interfaces. By applying a model based approach, requirements, and system interfaces become much more than simple text as they can be automatically validated and re-used. Design decisions and constraints at abstraction level N determine implicitly system properties at level N+1. Writing explicit requirements for the lower abstraction levels becomes superfluous as they are expressed implicitly in the model.

REUSABLE REQUIREMENTS

Domain Specific Language

Generic boilerplates for requirements [5] are adapted to fit the telescope domain and form a domain specific language (DSL) for requirements. They are customized from the standard SysML requirements by defining and applying stereotypes. The quantifiable parameters of the requirements are leveraged by a strongly typed customization as shown in Figure 1. The attributes are of type nm and Hz, as defined in the class `ClosedLoopWavefrontAttributes(1)`.
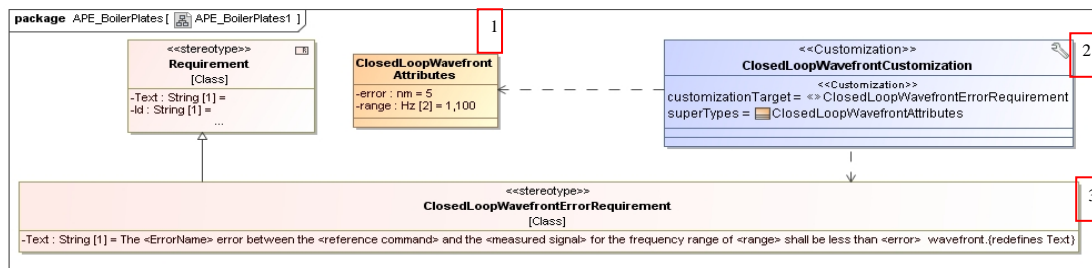

Figure 1: Definition of requirement boilerplates as a domain specific language.

Each boilerplate becomes an element of the DSL and can be reused in new systems (the `ClosedLoopWavefrontCustomization (2)` is merely a tool artefact to tie the derived stereotype to a default attribute values set).

Instantiation of Boilerplates

When a boilerplate is instantiated, the default text is copied automatically and the attributes defined in the DSL become the attributes of the concrete requirement with its own default values. The system requirements `IM Closed Loop (4)` and `PWFS Closed Loop (5)` in Figure 2 are instances of the boilerplate `ClosedLoopWavefrontErrorRequirement(3)` in Figure 1.
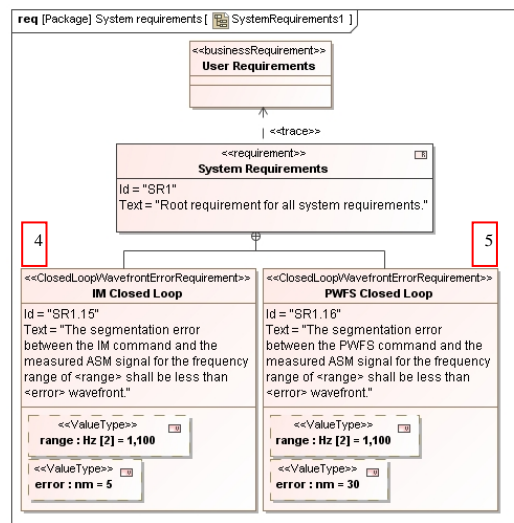

Figure 2: Instances of boilerplates.

_____

[#]rkarban@eso.org

Constrain the Design

The strongly typed parameters[*] of the requirement enable the modeller to bind them to properties of the system under design using the SysML «ConstraintBlock» in parametric diagrams. Together with the properties of the actuator (lag and bandwidth), the parameters of the `PWFS Closed Loop` requirement determine the sampling frequency. The model of their relationship is expressed as the «constraint» `ClosedLoopModel (6)`, as shown in Figure 3. This «constraint» represents in itself a reusable element and constrains the design at the system level. The sampling frequency is propagated down the system hierarchy, namely the control system, via another «constraint», the `MaxCorrectionTime (7)` which determines system properties further down the hierarchy. For example, the `meanTCCDAcquisitionTime (8)` is constrained by a `SensorSensitivityModel (9)` and associated requirements of the reference source.

Furthermore, trade-offs among requirements, system properties, and sub-system properties can easily be evaluated by executing the parametric model of the system. Therefore, requirements can be automatically verified against the design and vice versa.
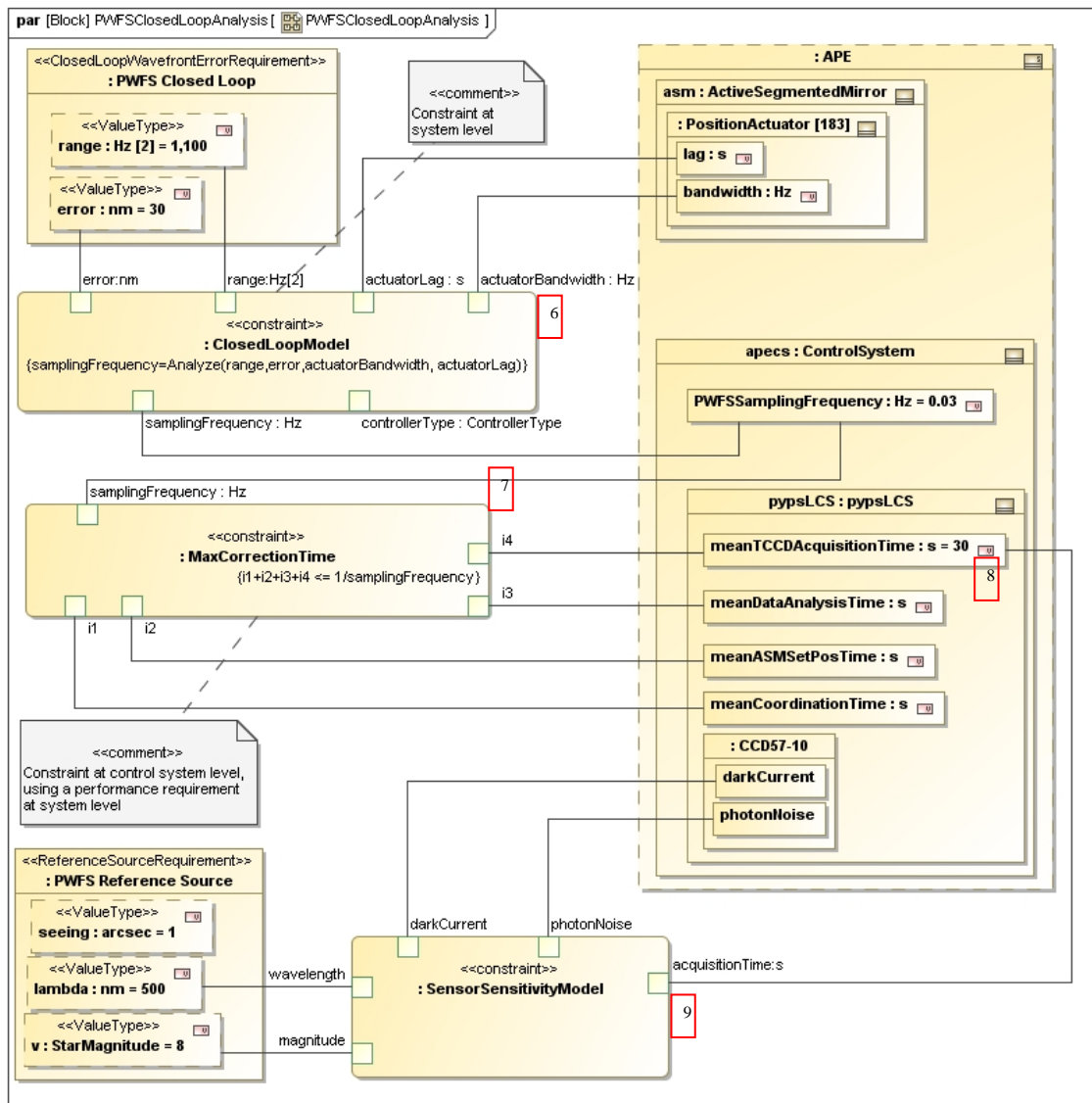
---

[*] While modeling tools already partially support parameters for requirements, this will be recommended for future revisions of the SysML specification

Figure 3: Constraining the design with requirements.

REUSABLE CONSTRAINTS

SysML constraints can also be used to define quantifiable system interfaces. Figure 4 shows a parametric description of the physical interface between a VLT Unit Telescope and an Instrument placed on one of the telescope's Nasmyth platforms. The system properties of the telescope (e.g. volume, mass) constrain the properties of the attached instrument; in the example the instrument is APE. The constraints of the «constraint» `NasmythInstrumentSpecification (10)` evaluate the properties of APE against the properties of the telescope. On either side, those properties become in turn requirements of the lower level system hierarchy and determine the properties of its parts. In the end, a mass roll-up of the complete APE system can be done to verify that it complies with the constraints given by the specification. On the telescope side, the allowed quantities are propagated down the system hierarchy to properly reflect the design decisions taken at the highest level.

The `UnitTelescope` owns a part property of type `NasmythPlatform (11)` which is at the same time part of the interface for an attached instrument. This fact is also expressed by a SysML standard port of the same type (see `NA:NasmythPlatform (12)` in Figure 4).
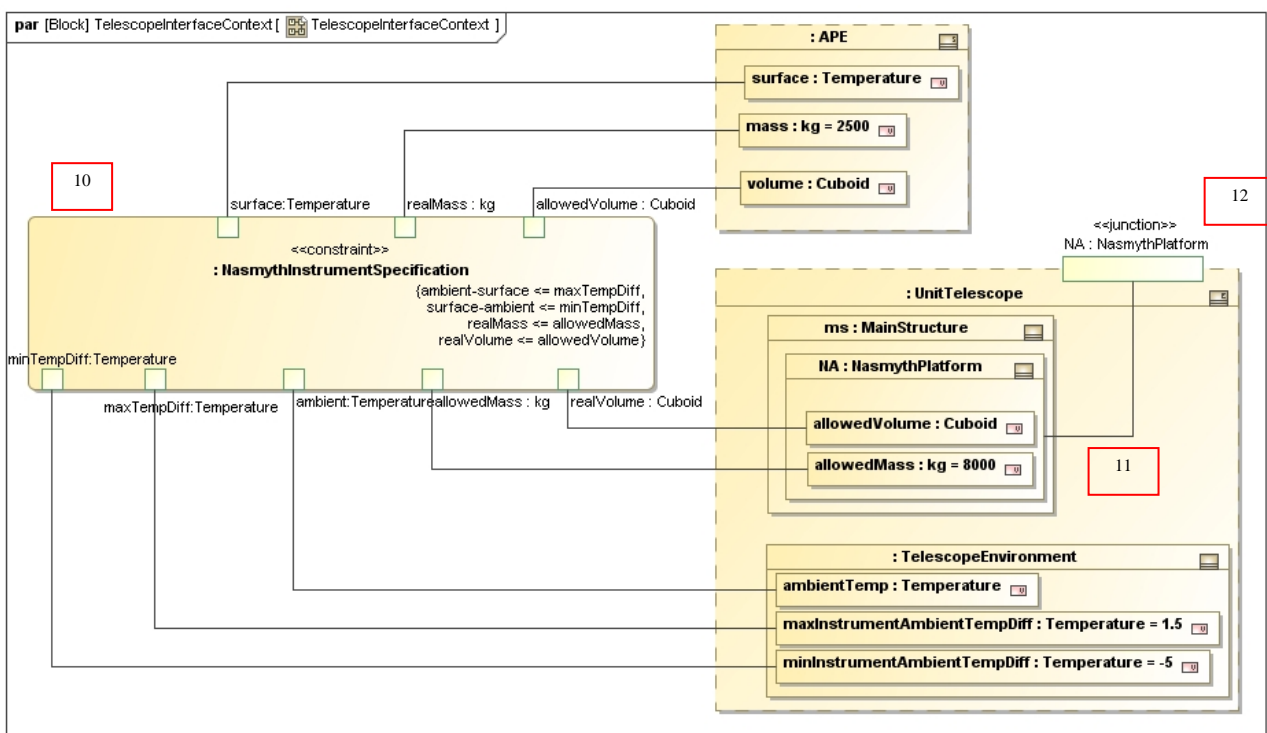

Figure 4: Defining interfaces with constraints.

CONCLUSION

System modelling offers a wide variety of re-use. We have shown how text only requirements can be parameterized and their properties formally typed. The SysML provides the means to bind those properties to constraint parameters to constrain the system design. Requirements can be re-used in a formal way and create a consistent requirements specification across projects. Relations among system properties are described with «ConstraintBlock»s which allow a formal re-use of specifications and constrain the design of systems of systems. In a model based approach, explicit modelling of requirements does not add information and can be automated. We observed an increase in re-usability with increasing abstraction level requiring different approaches for capturing commonalities. A promising approach consist in creating a repository of reusable modelling artifacts.

REFERENCES

[1] OMG Systems Modeling Language (OMG SysML), v1.1, formal/ 08-11-02, http://www.omgsysml.org, OMG (2008).

[2] S. Friedenthal and A. Moore and R. Steiner, "A Practical Guide to SysML", September 2008, Morgan Kaufmann, OMG PressA. Name and D. Person, Phys. Rev. Lett. 25 (1997) 56.

[3] C. Delp, NASA/JPL, private communications 2009

[4] N. Jankevicius, NoMagic, private communications 2009.

[5] E. Hull, K. Jackson, J. Disk, "Requirement Engineering", Springer, 2005.