Modeling OMG SMM metrics using the Modelio modeling tool in the MEASURE project

Alessandra Bagnato and Andrey Sadovykh
Softeam Research & Development Division
Paris, France
FirstName.LastName @softeam.fr

Sarah Dahab, Stephane Maag and Ana R. Cavalli
Telecom SudParis, Université Paris-Saclay, CNRS UMR 5157, France
{sarah.dahab, stephane.maag, ana.cavalli}@telecom-sudparis.eu

Alin Stefanescu
University of Bucharest, Faculty of Mathematics and Computer Science, Bucharest, Romania
alin@fmi.unibuc.ro

Jerome Rocheteau
ICAM
jerome.rocheteau@icam.fr

Sana Mallouli and Wissam Mallouli
Montimage
FirstName.LastName @montimage.com

Key words: Model-Driven Development, Model repository centralization, Structured Metrics Meta-model

Abstract: Most software system properties can be quantified with the application of measurement processes. OMG's Structured Metrics Meta-Model (SMM) supports the meta-model agnostic method to define these measurements. This paper introduces the first building block of the MEASURE (Measuring Software Engineering) [5] project tool chain: the Modelio modeling tool together with the SMM Module developed based in Modelio's open source distribution. These tools permit the formal specification of metrics and a common interchange format enabling interoperability between the project's tools. This paper presents this new approach of metrics specification that simplifies their combination and their integration into compliant platforms and tools.

1. Introduction

In today's software systems, it becomes more and more necessary to define better metrics and develop methods and tools for automated, precise, and unbiased measurement of software engineering activities and artefacts. Measurements provide data for disciplined engineering such that engineers and managers rely on these comparable evaluations in assessing the static and operational qualities of systems [1].

The general goal of the ITEA3 MEASURE (Measuring Software Engineering) project [5][6] is to increase the quality and efficiency, and to reduce the costs and time-to-market of software engineering in Europe. By implementing a comprehensive toolset for automated and continuous measurement, this project allows future software engineering processes to properly measure and access their impact on final software costs. More importantly, it opens a new field for innovation related to advanced analytics of the measurement data enabled by any complex software development project. Within the MEASURE project [5][6], some tools [3][4] will integrate measures into software development environments and processes of the industrial partners, e.g., DCNS [5], and will iteratively improve them based on the gathered feedback.

In order to provide a standard way to model metrics and as a common interchange format to allow interoperability between existing tools and their respective models, the MEASURE partners have studied different metric specification languages and selected the OMG Structured Metrics Meta-Model (SMM) [1]. SMM has been designed for exchanging measures and data measurements.

The paper is organized as follows: In Section 2, we present the state of the art of metric specification languages and motivate our selection of SMM metamodel. The Section 3 introduces the Modelio modeling tool [2][3] and its SMM module developed in the open source distribution of Modelio. Section 4 concludes the paper.

2. Structured Metrics Meta-model (SMM)

2.1 Languages for Metrics and Measurements Specification

Measurement has become a fundamental aspect of software engineering. For example, accurate measurement is required to guarantee a system's quality (by highlighting problematic areas), and in the determination of better work practices. Software measurement is also a key element in initiatives such as SW-CMM (Capability Maturity Model for Software), ISO/IEC 15504 (SPICE, Software Process Improvement and Capability determination) and CMMI (Capability Maturity Model Integration). The ISO/IEC 90003:2004 standard also highlights the importance of measurement in managing and guaranteeing quality.

A description language that allows to represent the elements that must be considered in the measurement processes is important for decision making and in process improvement. Among these languages, we mention GMSL (Goanna Metric Specification Language) [14], Slammer (Specification LAnguage for Modelling MEasurements and Redesigns) [15], and SMML (Software Measurement Modeling Language) [16]. A summary of their features is provided in Table 1, per the following four criteria:

- Software lifecycle coverage: metrics should be related to different phases of the software lifecycle including design, implementation, test, and operation as well as metrics related to engineering processes.
- Metric complexity: metrics can be measured directly or can be derived from other metrics. The metric specification language should allow combinations of direct metrics to define derived ones.
- Tool: the metric specification language should be supported by existing tools and these tools should be available and maintained.
- Based on standards: the language should rely on an existing standard to allow its interoperability with compliant platforms and tools.

Table 1. Comparison matrix for metric specification languages

	Software lifecycle coverage (SLC)	**Metric complexity**	**Tool**	**Based on standard**
GMSL	(-) Only C/C++ is supported	(-) Only direct metrics	(+) Available	(-) No
SLAMMER	(+) Can cover all SLC	(-) Only direct metrics	(+) Available	(-) No
SMML	(+) Can cover all SLC	(+) Direct and complex metrics	(-) Not maintained	(+) Based on SMM
MEASURE APPROACH	**(+)** Can cover all SLC	**(+)** Direct and complex metrics	**(+)** Based on Modelio	**(+)** Based on SMM

On the last row of Table 1, we also added the "MEASURE Approach", that we developed in our project. From the analysis of the first three languages and state of art, the MEASURE partners have decided to rely on SMM, an OMG standard. The language allows targeting direct and complex metrics and can be supported later by the

MEASURE framework solution. Since Softeam is one of the MEASURE partners, it integrated the SMM standard as a module in its industrial Modelio tool.

2.2 The SMM features

The SMM specification defines a meta-model for representing measurement with an initial focus on software, its operation, and its design. It is an extensible meta-model for exchanging both measures and measurement information concerning artefacts contained or expressed by structured models, such as the OMG Meta Object Facility (MOF™) standard. The SMM is a specification for defining measures and representing their measurement results. The measure definitions make up the library of measures to establish the specification upon which all the measurements will be based. In this paper, we follow the SMM specification that defines:

- “Measure” as “a method assigning comparable numerical or symbolic values to entities to characterize an attribute of the entities”.
- “Measurement” as “a numerical or symbolic value assigned to an entity by a measure”.
- “Measurand” as “an entity quantified by a measurement”.
- “Unit of Measure” as “a quantity in terms of which the magnitudes of other quantities within the same total order can be stated”.
- “Measurement Scope” as “the domain (i.e., set of entities) to which a given measure may be applied”.

3. Structured Metrics Meta-model (SMM) in Modelio

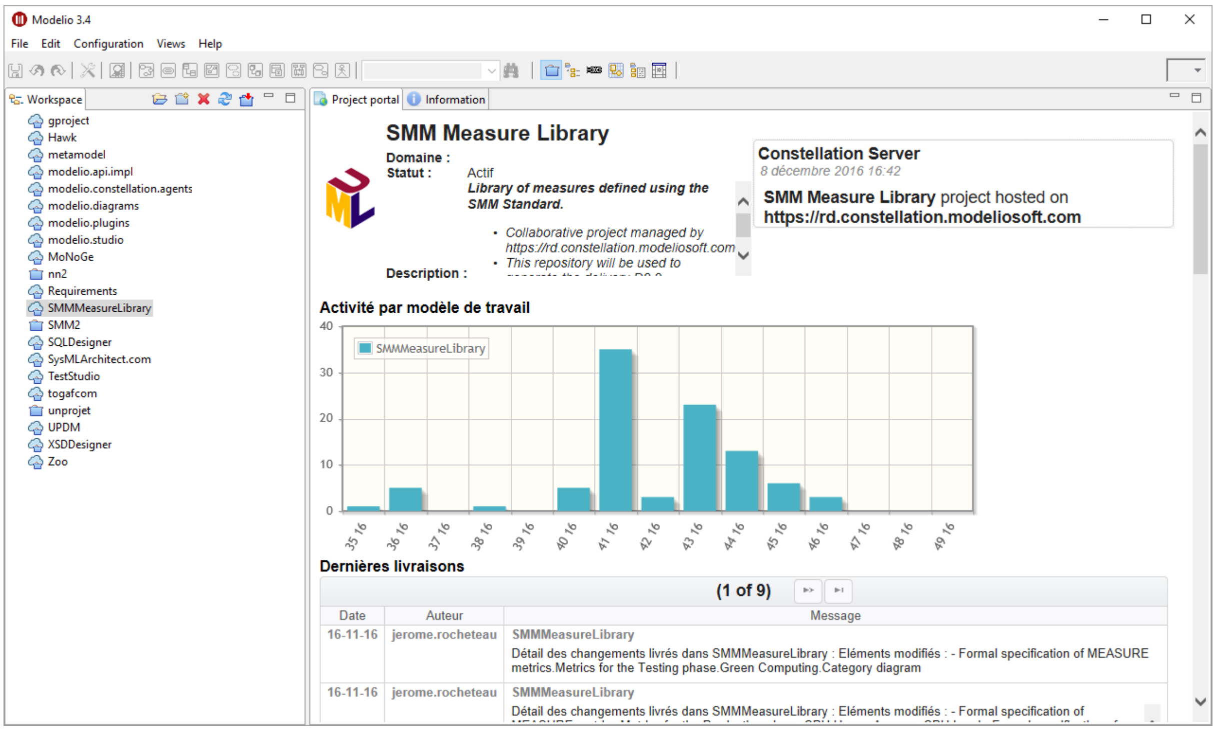
3.1 Modelio

Modelio [2] [3] is a comprehensive MDE workbench tool supporting the UML2.x standard. Modelio adds modern Eclipse-based ergonomics to the solid modeling and generation know-how obtained with the earlier Softeam MDE workbench, Objecteering, which has been on the market since 1991. Modelio provides a central repository for the local model, which allows various languages (UML profiles) to be combined in the same model, abstraction layers to be managed and traceability between different model elements to be established. Modelio proposes various extension modules, enabling the customization of this MDE environment for different purposes and stakeholders. Finally, Modelio is highly extendable and can be used as a platform for building new MDE features.

3.2 Specification of SMM metrics using Modelio

The Modelio tool [2] enables to use the UML2 Profiles and to combine them with a rich graphical user interface for dedicated diagrams, model element property editors and action command controls. One of them is our Structured Metrics Meta-model (SMM) [13] profile which will be released as open source under Apache 2 License to the modelling community.

The SMM specification allows the definition of measures and the representation of their measurement results. It includes elements representing the concepts needed to express a wide range of diversified measures. We modelled a shared library containing 143 metrics as in the MEASURE Library Project Portal in Figure1.



SMM Measure Library

Domaine :
Statut : Actif

Library of measures defined using the SMM Standard.

Description :

- *Collaborative project managed by https://rd.constellation.modeliosoft.com*
- *This repository will be used to*

Constellation Server
8 décembre 2016 16:42
SMM Measure Library project hosted on **https://rd.constellation.modeliosoft.com**

Activité par modèle de travail

Dernières livraisons

(1 of 9)

Date	Auteur	Message
16-11-16	jerome.rocheteau	SMMMeasureLibrary Détail des changements livrés dans SMMMeasureLibrary : Eléments modifiés : - Formal specification of MEASURE metrics.Metrics for the Testing phase.Green Computing.Category diagram
16-11-16	jerome.rocheteau	SMMMeasureLibrary Détail des changements livrés dans SMMMeasureLibrary : Eléments modifiés : - Formal specification of

Figure 1 SMM MEASURE Library Project Portal

In the following, we present seven metrics modeled in SMM using the Modelio tool. They all illustrate diverse concepts specified by the OMG standard and integrated within the tool. Thanks to that smooth integration, different types of software metrics have been successfully modeled, from simple ones (memory access count and I/O usage), to composite (computational energy cost) and *subjective* metrics (cognitive complexity and emotional usability).

3.2.1 Memory Access Count Metric

The Memory Access Count (MAC) measure computes the number of memory accesses to the data memory while the software is running. This simple metric targets the source code of the measured software (measurand) and returns a number as a result. As illustrated in Figure 2, each concept of a measurement can be modeled in SMM. The target of a measure is represented by the scope in SMM and by the target symbol in the figure. The unit of measure is also represented in SMM and seen in Figure 2 as the symbol "{…}" and the measure itself. Moreover, with SMM, we define the measure type. For instance, since this simple measure does not depend on another one, it is called DirectMeasure in SMM and represented by the microscope in the Figure 2*Figure 3*. The measure model is linked to each of its properties via the "ScopeLink" and "UnitLink".

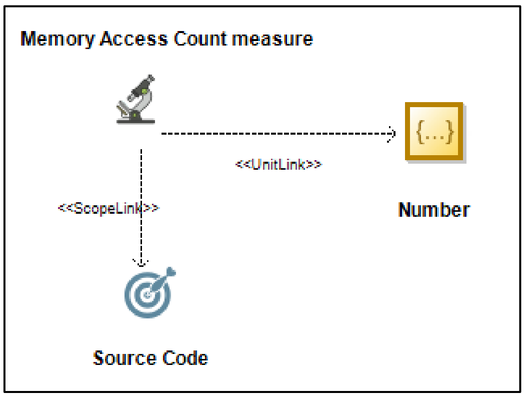


Figure 2: The Memory Access Count Metric modeled in SMM with the Modelio tool.

3.2.2 I/O Usage Metric

The I/O Usage measure is another simple metric. It computes the percentage of the bytes transferred by an application over a system component set during a second. This measure is an independent (or *direct measure*) measure and the scope is the application in execution and the unit of measure is the byte/second. As for the previous measure, those measurement properties are modeled in SMM in the Figure 3.

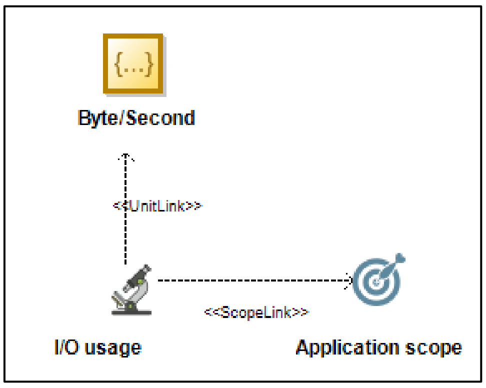


Figure 3: The I/O Usage Metric modeled in SMM with the Modelio tool.

3.2.3 Computational Energy Cost Metric

The computational energy cost metric is a composite metric, i.e., composed of several *unit* measures. This metric is composed of the CPU processing, the memory access number and the I/O operations. It corresponds to the sum of the energy costs of the CPU processing, the memory accesses and the I/O. When we identify the requirements through the SMM specification and the Modelio tool, we define the scope of this measure as the application during its execution. Its unit of measure is the Joule. Since this measure depends on three other measures (two unit ones: I/O usage, Memory Access Count and a composite one: CPU usage), a new SMM concept is needed in this model. Within SMM, this type of measure is called a Collective Measure and is represented in the model of the Figure 4 by the cube built by three colored layers. In addition to the above-mentioned advantages, it is easy to know the link between the different measures with SMM.

Also, note that Modelio easily avoids interleaved or crossed arrows whenever several relationships or links are needed between the SMM concepts. These straight lines clearly ease the modelling and facilitate the reading of the SMM models.

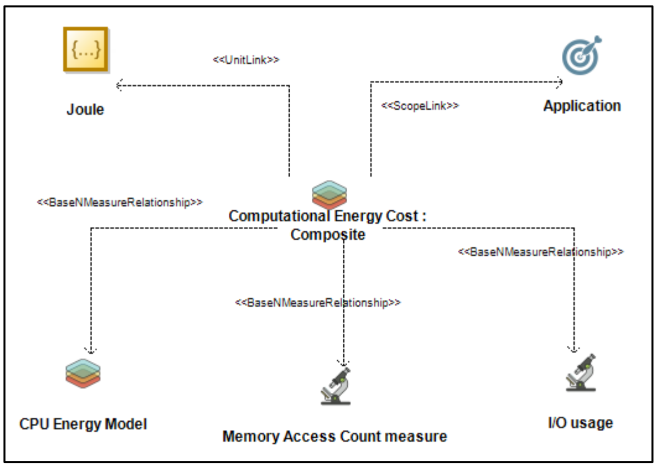

{...}
Joule
<<UnitLink>>
<<ScopeLink>>
Application
<<BaseNMeasureRelationship>>
Computational Energy Cost : Composite
<<BaseNMeasureRelationship>>
<<BaseNMeasureRelationship>>
CPU Energy Model
Memory Access Count measure
I/O usage

Figure 4: The Computational Energy Cost Metric modeled in SMM with the Modelio tool

3.2.4 Cognitive Complexity Metric

In this subsection, we tackle the specification of an interesting metric notified as *subjective*. The cognitive complexity has been first defined in [10] and [11] and is based on the cognitive weight that may be set on code design (classes, methods, attributes, etc.). The well-known metric called “Class Complexity due to Inheritance” (CCI) measures the complexity of object-oriented (OO) code design due to the inheritance and its maintainability. It returns a cognitive weight: a low value means a better design information.

This *unit* metric computes the cognitive weight of a class by computing the weight of each method and adds the complexity weights of the directly inherited classes. Therefore, in order to compute the full code weight, CCI adds the weight of all classes if they are at the same level or multiply it if there is a derivation link. As illustrated in the Figure 5, the scope is an object-oriented code, the result type is a weight. Modelio allows to specify the SMM concept of measure characteristics. It is represented by the symbol of panels. The SMM characteristic element gives information about what is measured giving then more precisions about the measurement. For example, in the figure, it is clearly defined that this measure computes the inheritance complexity of a code via the characteristic element. The advantage of this option is to make possible the differentiation between two measures which may have an identical specification but with a different semantic. Also, this allows to know what is the feature of the measure.

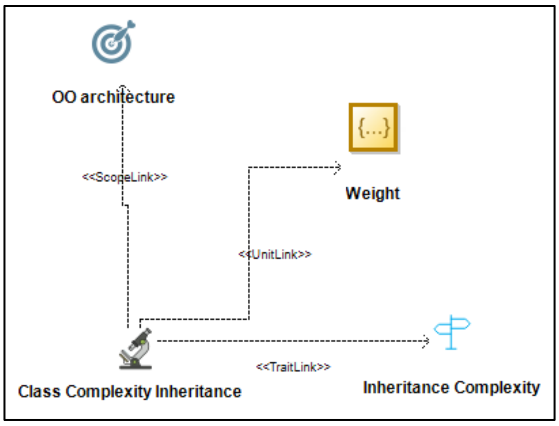


Figure 5: The Class Complexity Inheritance Metric modeled in SMM with the Modelio tool

Through this example, we illustrate that SMM, and its implementation in the Modelio tool, enables to specify a wide variety of different kinds of metrics, from unit, composite, technical but also *subjective* ones. In the next subsection, another interesting cognitive metric is modeled derived from two characteristic elements.

3.2.5 Emotional Usability Metric

This type of metric is based on the user’s emotions, how we may measure data and analyze them and it is dedicated to the understanding of emotional usability when using a software, a tool, etc. [12] Compared to the above measures, those metrics are specifically based on that subjective concept.

It is commonly accepted that there are two methods for collecting the emotions of a user: one based on the user verbal expression and one based on the physical analysis of the experimented user (e.g., expression of the face, etc.) [12]. Those two methods can be used together.

The Figure 6 and Figure *7* illustrate their SMM modeling using the concepts presented below, that is, the scope (verbal emotion for the first one and non-verbal emotion for the second), the type of expected result (an Index) and what information they give on the measurand (the usability of an application). The type of dependence of the measures shows they are independent (DirectMeasure).

3.2.6 PAD Scale

The PAD Scale (Pleasure, Arousal, Dominance) measures the usability of a program during the manipulation of its UI. It measures the verbal emotion of users and is based on adjective word pairings defining different levels of three main emotions: the pleasure, the arousal and the dominance.

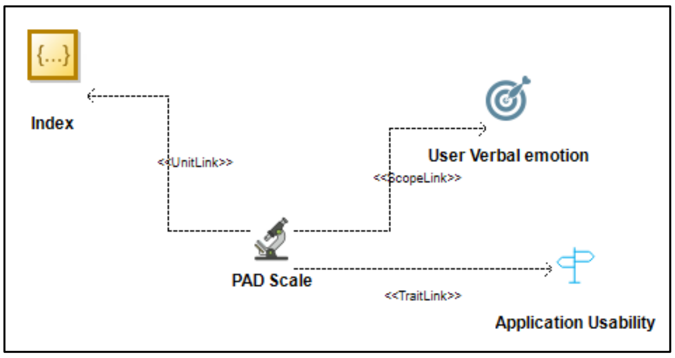


Figure 6: The PAD Scale Metric modeled in SMM with the Modelio tool

3.2.7 Emocard

The Emocard measures the usability of a program during the manipulation of its UI using non-verbal emotion. This metric uses the facial expression of users to measure their emotions. It is based on the user's face expression defining different levels of the three above-mentioned emotions.

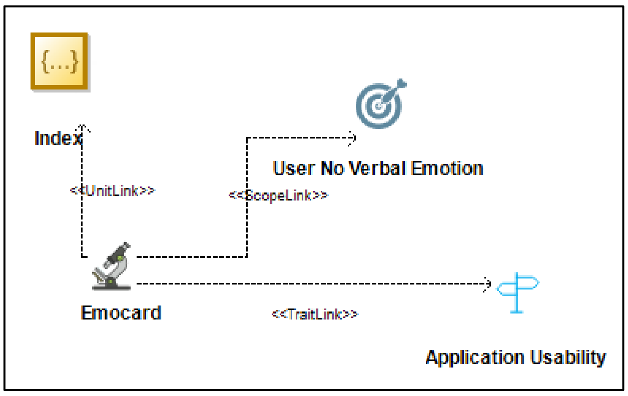


Figure 7: The Emocard Metric modeled in SMM with the Modelio tool

4. Conclusions and future work

In this paper, we motivate the use of the OMG SMM standard for specifying software metrics by comparing with other formal languages. Its smooth integration has been performed through a module developed into the modelling tool Modelio produced by the French company Softeam Cadextan [8]. Based on diverse usage examples, we herein demonstrate its usability. Our paper shows that this tool implementation allows, thanks to the SMM specification as a common interchange format, to specify simple, complex but also composite and even subjective metrics (e.g., cognitive complexity).

In future works, several perspectives may be raised. First, we plan to execute the specified metrics through the different industrial use cases provided by the MEASURE project. Besides, the dedicated and inherent modelling processes are expected to be properly and seamlessly integrated into the project platform. Furthermore, we will define consistent measure plans with important generated measured data. By these huge data sets, the scalability and sustainability of our approach will be then evaluated.

Acknowledgements.

The research leading to these results was partially funded by the ITEA3 project 14009, MEASURE [5] [6].

References

[1] Structured Metrics Meta-model (SMM), Version 1.1.1 OMG Document Number: formal/2016-04-04 Standard Document URL: http://www.omg.org/spec/SMM/ Last visit: March 2017
[2] Modelio (Softeam) https://www.modeliosoft.com/ . Last visit: March 2017
[3] Modelio Open Source Distribution (Softeam) www.modelio.org . Last visit: March 2017
[4] Montimage Monitoring Tool (MMT) http://www.montimage.com/products.html . Last visit: March 2017
[5] The MEASURE project: Measuring Software Engineering, http://measure.softeam-rd.eu/home. Last visit: March 2017
[6] Project 14009, MEASURE, Measuring Software Engineering, https://itea3.org/project/measure.html. Last visit: March 2017
[7] Object Management Group, Inc. (OMG) http://www.omg.org/. Last visit: March 2017
[8] Softeam Cadextan, http://www.softeam.com. Last visit: March 2017
[9] Monperrus, M., Jézéquel, J.-M., Baudry, B., Champeau, J., Hoeltzener, B.: Model-driven generative development of measurement software. Software and System Modeling 10(4), pp. 537-552, 2011
[10] Rilling, J., Klemola, T.: Identifying comprehension bottlenecks using program slicing and cognitive complexity metrics. In 11th IEEE International Workshop on Program Comprehension, pp. 115-124, 2003.
[11] Andrews, G., Halford, G.S.: A cognitive complexity metric applied to cognitive development. Cognitive psychology, 45(2), pp. 153-219, 2002.
[12] Agarwal, A., Meyer, A.: Beyond usability: evaluating emotional response as an integral part of the user experience. In the Proc. of the ACM CHI Conference on Human Factors in Computing Systems. ACM, pp. 2919-2930, 2009.
[13] Structured Metrics Meta-model Modelio Forge https://forge.modelio.org/projects/smm Last visit: March 2017
[14] Andreas Vogelsang, Ansgar Fehnker, Ralf Huuck, Wolfgang Reif: "Software Metrics in Static Program Analysis", in the proceedings of the 12th International Conference on Formal Engineering Methods, ICFEM, pp. 485-500, Shanghai, China, 2010.
[15] Esther Guerra, Juan de Lara, Paloma Díaz: "Visual specification of measurements and redesigns for domain specific visual languages", Journal of Visual Languages & Computing, V.19, I. 3, pp. 399-425, 2008.
[16] Beatriz Mora, Mario Piattini, Francisco Ruiz, Felix Garcia: "Smml: Software measurement modeling language", in the proceedings of the 8th Workshop on Domain-Specific Modeling (DSM'2008), 2008.