# Toward a Metamodel Quality Evaluation Framework: Requirements, Model, Measures, and Process

Taciana Novo Kudo*
Departmento de Computação
Universidade Federal de São Carlos
São Carlos, São Paulo, Brazil
taciana@ufscar.br

Renato F. Bulcão-Neto
Instituto de Informática
Universidade Federal de Goiás
Goiânia, Goiás, Brazil
rbulcao@ufg.br

Auri M. R. Vincenzi
Departmento de Computação
Universidade Federal de São Carlos
São Carlos, São Paulo, Brazil
auri@ufscar.br

## ABSTRACT

The quality of metamodel considerably affects the models and transformations that conform to it. Despite that, there is still little discussion about a comprehensive form to evaluate the quality of metamodels and its consequences in model-driven development processes. This paper proposes a metamodel quality evaluation framework called MQuaRE (Metamodel Quality Requirements and Evaluation). MQuaRE comprises metamodel quality requirements and measures, a quality model, and an evaluation process, with the evident influence of international standards for software product quality, such as ISO/IEC 25000 series. We present a simple use case of MQuaRE describing how requirements, measures, and the quality model should be used during the evaluation process of a metamodel for software patterns. Among other benefits, MQuaRE can help determine final metamodel quality, decide on the acceptance of a metamodel, and also assess the positive and negative aspects of a metamodel, contributing to its quality evolution.

## CCS CONCEPTS

• **Software and its engineering** → **Model-driven software engineering**; • **General and reference** → *Evaluation.*

## KEYWORDS

Model-driven software engineering, metamodeling, quality, requirement, model, measure, process, evaluation

**ACM Reference Format:**
Taciana Novo Kudo, Renato F. Bulcão-Neto, and Auri M. R. Vincenzi. 2020. Toward a Metamodel Quality Evaluation Framework: Requirements, Model, Measures, and Process. In *34th Brazilian Symposium on Software Engineering (SBES '20), October 21–23, 2020, Natal, Brazil.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3422392.3422461

*Currently, Mrs. Kudo is a Ph.D. candidate at the Universidade Federal de São Carlos and also works as an assistant professor at the Universidade Federal de Goiás.


*SBES '20, October 21–23, 2020, Natal, Brazil*

ACM ISBN 978-1-4503-8753-8/20/09…$15.00
https://doi.org/10.1145/3422392.3422461

## 1 INTRODUCTION

Models are the primary artifacts of model-driven software engineering (MDSD) [1], and a terminal model is a representation that conforms to a given software metamodel [10, 13]. As the quality of a software metamodel directly impacts the quality of terminal models, software metamodel quality is an essential aspect of MDSD.

However, the literature reports a few proposals for metamodel quality evaluation, but most lack a general solution for the quality issue. Some efforts focus on quality measures [8], a quality evaluation model [12], or a quality evaluation model with structural measures borrowed from OO design [9, 11, 14]. Thus, we support there is a need for a more thorough solution for metamodel quality evaluation, with potential benefits to MDSD in general.

In this paper, we propose a metamodel quality evaluation framework called MQuaRE (Metamodel Quality Requirements and Evaluation). MQuaRE is an integrated framework composed of metamodel quality requirements and measures, a metamodel quality model, and an evaluation process, with a great contribution of the ISO/IEC 25000 series [4] for software product quality evaluation.

MQuaRe points out the relevance of quality requirements for metamodel evaluation. The quality model includes quality characteristics and sub-characteristics of metamodels. MQuaRe also provides general measures for metamodel quality. Finally, MQuaRE's evaluation process arranges requirements, model, and measures with activities, tasks, input and output artifacts, and users' roles. We also present a simple use case of the evaluation of the quality of a metamodel for software patterns [6].

This paper is organized as follows: Section 2 outlines related work; Section 3 describes the MQuaRE framework; Section 4 presents a use case of MQuaRE; and Section 5 brings our conclusions and future directions.

## 2 RELATED WORK

This section presents related work on metamodel quality evaluation.

Ma et al. [8] define a method and quality measures to assess UML-based metamodels. The proposal can identify and characterize the stability and design quality of metamodels, influenced mainly by the OO paradigm.

Strahonja [12] defines a quality evaluation model for workflow metamodels based on nine categories: domain application, origins, concepts and constructs, modeling language and notation, cohesion, openness, usability, maintainability, and pragmatic aspects. The use of the proposed model relies on the individual capabilities and competencies of the metamodel evaluators. This subjective aspect

of Strahonja's work is mainly due to the non-existence of measures associated with the quality categories.

Williams et al. [14] propose measures based on class diagram size (e.g., no. of classes), whereas Rocco and others [11] correlate metamodel measures, such as the no. of classes and inheritance.

Ma et al. [9] define an evaluation model composed of quality attributes, quality properties characterizing each attribute, and measures assigning a value to each property. Their model includes five quality attributes, each one with its respective properties. Besides, measures are defined based on quality measures for OO models.

In turn, MQuaRE is a comprehensive metamodel evaluation framework comprised of a process that combines a quality model, requirements, and measures. To the best of our knowledge, none of the previous work congregates MQuaRE's features.

## 3 THE MQUARE FRAMEWORK

This section details how MQuaRE components are arranged toward metamodel quality evaluation, as seen in Figure 1.

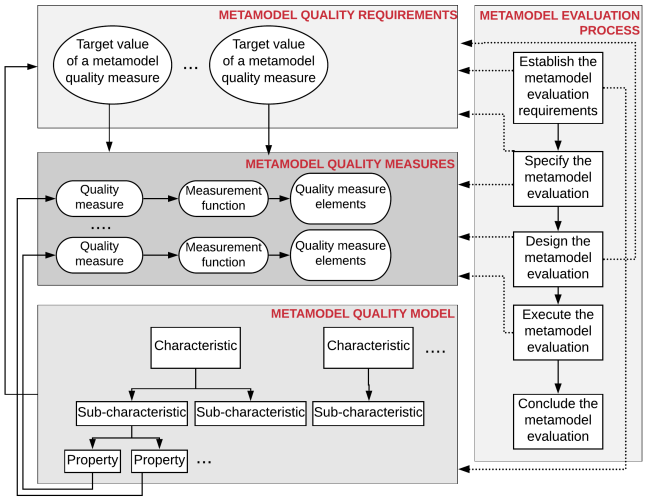


**Figure 1: An overview of MQuaRE: evaluation process and quality requirements, measures, and model.**

### 3.1 Metamodel Quality Requirements

Quality requirements specification plays a crucial role in the metamodel evaluation process. Should quality requirements are not stated clearly, the same metamodel may be interpreted and evaluated variously by different people. As a result, one achieves an inconsistent metamodel evaluation.

*Metamodel quality requirements* (MQR) may comprise multiples aspects of a metamodel, e.g., whether it is easy to use and maintain or compliant to specific standards, if applicable. MQR can be categorised through the MQuaRE's quality model we present in Section 3.2.

*3.1.1 Pre-conditions for Quality Requirements.* A quality model drives the documentation of metamodel quality requirements. Despite that, we recommend the following pre-conditions for MQR:

- MQR shall be uniquely identified and following the objective of the metamodel evaluation;
- MQR shall be associated with quality sub-characteristics, as defined in MQuaRE's quality model;
- MQR shall be specified in terms of a quality measure and a target value, which is the acceptable value for fulfilling a particular MQR;
- An acceptable tolerance value for the target value of a particular MQR shall be documented;
- Specific concepts and terms used in the metamodel should be used to avoid misunderstandings of the MQR;
- MQR shall be validated and approved by an evaluation requester.

The current version of MQuaRE provides 19 MQRs that meet those pre-conditions and can be reused by metamodel users.

*3.1.2 Quality Requirements Verification.* Defining the MQR is essential to avoid inconsistencies in the metamodel evaluation. Here is a list of recommendations to ensure the quality of MQR:

- MQR shall be verifiable, reviewed, and approved;
- Evaluation tools, techniques, or other resources (e.g., effort or time) required for verification shall be documented;
- Identified conflicts between MQR or between MQR and metamodel concepts shall be documented;
- The stakeholders' identities shall be documented.

Examples of MQR for the quality evaluation of a software pattern metamodel are shown in Section 4. Next, we present the MQuaRE's quality model, which works as guidance for the definition of MQR.

### 3.2 Metamodel Quality Model

The quality of a metamodel is the degree to which it provides value to a modeling activity. These stated needs are represented in the MQuaRE by a quality model that categorizes metamodel quality into *characteristics*, which in some cases, subdivide into *sub-characteristics*, as depicted in Figure 1. This hierarchical decomposition provides a convenient breakdown of metamodel quality.

The measurable quality-related properties of a metamodel are called *quality properties*. It is necessary to identify a collection of properties that cover characteristics or sub-characteristics, obtain quality measures for each, and combine them to achieve a derived quality measure corresponding to the quality characteristic or sub-characteristic. Thus, the quality model allows the categorization of metamodel quality requirements.

The quality model we propose revises ISO/IEC 25010:2011 [4], 9126-1:2001 [2] and related research [8, 9, 11, 12], and incorporates some quality characteristics and sub-characteristics with some amendments. Five characteristics form the MQuaRE's quality model as depicted in Figure 2: *Compliance*, *Conceptual Suitability*, *Usability*, *Maintainability*, and *Portability*. These characteristics may work as a checklist for ensuring comprehensive coverage of metamodel quality. Next, we overview the characteristics and sub-characteristics present in the MQuaRE's quality model.

*3.2.1 Compliance.* The degree to which the metamodel must comply with items such as widely accepted and sound theories, regulations, standards, and conventions. This characteristic includes conceptual compliance sub-characteristic.

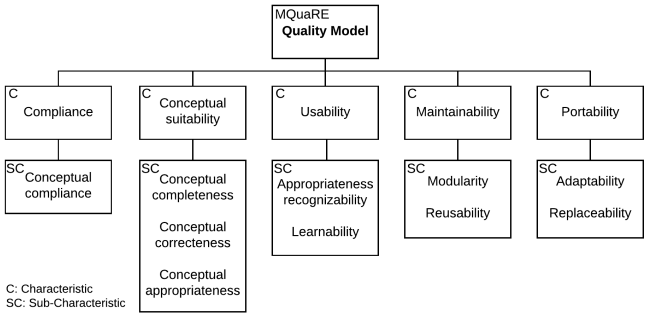


**Figure 2: The MQuaRE's quality model with quality characteristics and sub-characteristics.**

**Conceptual compliance:** the degree to which the metamodel to comply with such items as widely-accepted and sound theories, regulations, standards, and conventions concerning its conceptual foundation.

*3.2.2 Conceptual Suitability.* The degree to which the metamodel satisfies requirements when used under specified conditions. This characteristic subdivides into conceptual completeness, conceptual correctness, and conceptual appropriateness sub-characteristics.

**Conceptual completeness:** the degree to which the set of the metamodel concepts covers all the specified requirements.

**Conceptual correctness:** the degree to which the metamodel provides the correct modeling results with the needed degree of precision.

**Conceptual appropriateness:** the degree to which the metamodel facilitates the accomplishment of modeling tasks, and for determining their adequacy for performing these tasks.

*3.2.3 Usability.* The degree to which the metamodel can be used to achieve specified goals in a particular application domain. This characteristic includes the appropriateness recognizability and learnability sub-characteristics.

**Appropriateness recognizability:** the degree to which users can recognize whether the metamodel is appropriate for their needs or not.

**Learnability:** the degree to which the metamodel can be used by declared users to achieve specified goals of learning in a given context of use.

*3.2.4 Maintainability.* The degree of effectiveness and efficiency with which the metamodel can be modified by the intended maintainers. This characteristic includes modularity, reusability, and modifiability sub-characteristics.

**Modularity:** the degree to which the metamodel is composed of discrete concepts in such a way that a change of one concept has minimal impact on other concepts.

**Reusability:** the degree to which an asset can be used in more than one metamodel or in building other assets.

**Modifiability:** the degree to which the metamodel can be effectively and efficiently modified without introducing inconsistencies or degrading existing metamodel quality.

*3.2.5 Portability.* The degree of effectiveness and efficiency with which the metamodel can be transferred from one application domain to another. This characteristic includes adaptability and replaceability sub-characteristics.

**Adaptability:** the degree to which the metamodel can effectively and efficiently be adapted for different application domains.

**Replaceability:** the degree to which the metamodel can replace another specified metamodel for the same purpose in the same application domain.

The MQuaRE's quality model provides a basis for quantifying metamodel quality requirements through metamodel quality measures that we present next.

## 3.3 Metamodel Quality Measures

The quality characteristics and sub-characteristics can be quantified by applying *measurement functions*. A measurement function is an algorithm used to combine quality measure elements. The result of applying a measurement function is called a *quality measure*. In this way, quality measures are quantifications of the quality characteristics and sub-characteristics.

The current version of MQuaRE includes 23 quality measures bound to the quality model as follows: *Compliance* (2), *Conceptual Suitability* (4), *Usability* (5), *Maintainability* (8), and *Portability* (4). Every quality measure is described by an identification code, the measure name, a description of the information provided by the measure, and a measurement function.

Our quality measures are the result of an analysis of related work [8, 9, 11, 12] and the ISO/IEC 25023:2016 [5] and ISO/IEC 9126-3:2003 [3] standards. It is also noteworthy that the metamodel quality measures depend on the purpose of the evaluation, the selected quality characteristics, and the possibility to apply the measurements.

## 3.4 Quality Evaluation Process

The MQuaRE's process model assumes that the evaluation founds on the MQuaRE's requirements, making clear the objectives and criteria of assessment. Besides, the MQuaRE's quality model and measures should also be considered in the evaluation process.

Figure 3 depicts a BPMN-based representation for the MQuaRE's evaluation process with activities, user roles, and input and output artifacts. Activities and the respective tasks are detailed next.

*3.4.1 Establish the evaluation requirements.* The metamodel quality evaluation requirements must be identified, taking into account the evaluation purpose. The aim is to meet all quality requirements of the metamodel and considering restrictions, such as the evaluation purpose and target date. The following should be input for this activity: (i) metamodel quality evaluation needs; (ii) metamodel to be evaluated, including its specifications; and (iii) applicable evaluation tools and methodology. This activity consists of the following tasks:

(1) **Establish the objective of evaluating the metamodel**: the purpose of the metamodel quality evaluation shall be documented as a basis for further evaluation activities and

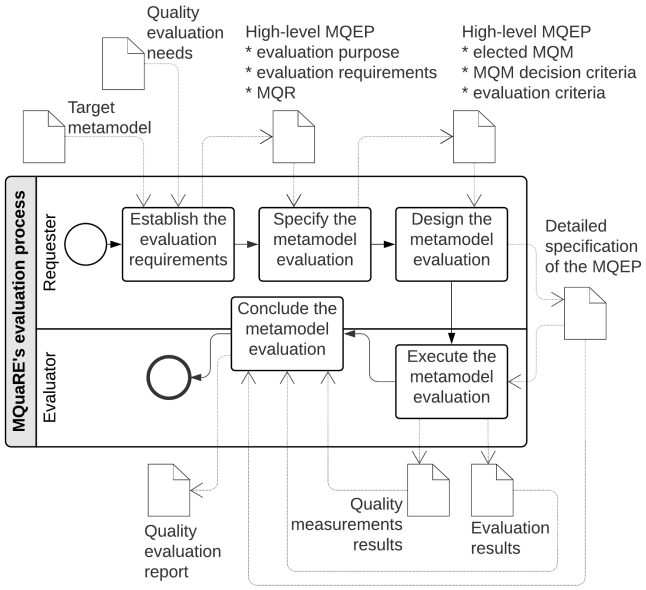


**Figure 3: The MQuaRE's Metamodel Evaluation Process.**

tasks. The targeted metamodel may be an intermediate version or a final version. Multiple evaluation purposes may take place such as deciding on the acceptance or assuring the quality of an intermediate metamodel version, the comparison between distinct metamodels for a same domain, assessing the positive and negative effects of a final metamodel version, among others.

(2) **Define the metamodel quality requirements**: these shall be specified using the MQuaRE's quality characteristics and sub-characteristics. Should a specification of MQR is available, it may be reused, reviewed, and refined.
(3) **Identify the artifacts to be used in the evaluation**: every metamodel-related artifact required for the evaluation shall be identified and registered. Examples of metamodel artifacts include metamodel specifications (e.g., requirements- and design-oriented), metamodel implementation, metamodel user documentation, just to name a few.

As illustrated in Figure 3, the main output artifact of this activity is a *high-level metamodel quality evaluation plan* (MQEP), which should contain the specification of purposes and requirements for metamodel quality evaluation as well as the specification of metamodel quality requirements.

*3.4.2 Specify the metamodel evaluation.* This activity consumes a high-level MQEP as input and consists of the following tasks:

(1) **Select the metamodel quality measures**: the requester shall select MQM's to cover all metamodel quality requirements. Measurement procedures should be measured with sufficient accuracy to allow criteria to be set and comparisons to be made.
(2) **Define decision criteria for metamodel quality measures**: decision criteria are numerical thresholds or targets used to determine the need for action or further investigation or to describe the level of confidence in a given result. These shall be defined for the selected individual measures often concerning quality requirements.
(3) **Establish decision criteria for metamodel evaluation**: the requester should prepare a procedure for further summarization, with separate criteria for different quality characteristics, each of which may be in terms of individual quality sub-characteristics and measures. The summarization results should be used as a basis for the metamodel quality assessment. Therefore, the evaluation results of the different characteristics shall be summarized to assess the quality of the metamodel.

The primary output artifact of this activity is a *revised high-level MQEP*, containing the chosen quality measures as well as decision criteria for metamodel quality measures and assessment.

*3.4.3 Design the metamodel evaluation.* The revised high-level MQEP previously presented is input data for this activity, which includes the following task:

(1) **Plan metamodel evaluation activities**: in this task, those metamodel quality evaluation activities identified shall be scheduled, taking into account the availability of resources such as personnel, evaluation tools, and examples of metamodel application domains. The evaluation plan should detail the following elements: the purpose of the metamodel quality evaluation, quality evaluation requirements including metamodel artifacts and related resources (e.g., personnel, tools, budget, and deadlines), MQR and metamodel quality measures, decision criteria for metamodel evaluation and metamodel quality measures, and an evaluation schedule.

As the evaluation activities evolve, the evaluation plan shall be revised until a thorough level plan. The outcome of this activity is a *detailed specification of the MQEP*.

*3.4.4 Execute the metamodel evaluation.* A thorough specification of the MQEP represents the input artifact for this activity that consists of the following tasks:

(1) **Compute metamodel quality measurements**: the selected metamodel quality measures shall be applied to the metamodel following the evaluation plan. As a result, values on the measurement scales are computed and assigned to quality measures.
(2) **Apply decision criteria for metamodel quality measures**: the decision criteria for the metamodel quality measures shall be applied to the measured values.
(3) **Apply decision criteria for metamodel quality assessment**: the set of decision criteria shall be summarized into quality characteristics and sub-characteristics. A statement of the extent to which the metamodel meets quality requirements describes the assessment results, which should:
- establish an appropriate degree of confidence that the metamodel can meet the evaluation requirements;
- identify any specific deficiencies concerning the evaluation requirements and any additional evaluations needed to determine the scope of those deficiencies;

- identify any particular limitations or conditions placed on the use of the metamodel;
- identify any weaknesses or omissions in the evaluation and any additional evaluation that is needed.

The following should be outcomes of this activity: *metamodel quality measurements results* and *quality evaluation results*.

*3.4.5 Conclude the metamodel evaluation.* This activity requires as input the detailed specification of MQEP, metamodel quality measurements results, and quality evaluation results. It consists of the following tasks:

(1) **Review metamodel evaluation results**: the evaluator and the evaluation requester shall carry out a joint review of the evaluation results.
(2) **Conclude the metamodel evaluation process**: depending on how the evaluation report is to be used, it should include the following items, among others: the MQEP, computed measurements results, performed analyses, intermediate results or interpretation decisions, the evaluators' profiles, the final result of the metamodel quality evaluation, and any necessary information to be able to repeat or reproduce the assessment.

The final outcome is a *metamodel quality evaluation report*.

## 4 USE CASE

This section presents a plain use case of the MQuaRE framework. As depicted in Figure 3, the evaluation process begins with the definition of the targeted metamodel and the primary reason for evaluation. In this case, we aimed to estimate the final quality of a metamodel called SoPaMM (Software Pattern MetaModel) [6], which defines how software patterns, in general, can be organized, classified, related, and represented.

A non-detailed MQEP was then created, including the SoPaMM artifacts and related resources both required for the evaluation. For this purpose, we developed a suite of software artifacts documenting the SoPaMM's development: requirements specification, design specification, metamodel implementation, and user documentation. Human resources for the evaluation included two evaluators; in this case, the SoPaMM's and the MQuaRE's original creators.

Next, we describe an excerpt of the list of MQR chosen for the SoPaMM metamodel:

**MQR2.** The metamodel must cover the concepts[1] found in its specifications.
**MQR7.** The evaluator must be able to recognize whether a metamodel is appropriate for their needs through evident concepts found in the specifications.
**MQR8.** The evaluator must be able to recognize whether the metamodel concepts' purpose is correctly interpreted through the specifications.
**MQR13.** The evaluator must be able to recognize modifications in the metamodel based on documented changes in the specifications.

[1]Concepts are foundation elements of a metamodel, e.g., a concept may be a class, relationship, or attribute in a UML metamodel. As we propose MQuARE as a generic-purpose framework, concepts may vary according to the specification language of the metamodel under evaluation.

Next, we refined MQEP with MQuaRE's quality measures required for addressing the SoPaMM's quality requirements (i.e., 2, 7, 8, and 13). Given that every measure is associated with a sub-characteristic in the MQuaRE's quality model, we show in Table 1 the correspondence among MQR, measure, and quality sub-characteristic and characteristic. In sequence, Table 2 details each measure required for the evaluation of the SoPaMM. Also, we assigned a target value and an acceptable tolerance value to each quality measure in the evaluation of the SoPaMM.

As we chose multiple quality characteristics for evaluation, we defined a quality formula as the weighted mean of measures grades per characteristic. Equation 1 presents a final quality (FQ) formula with the measures grades described in Table 2:

$$FQ = 10 * ((CCp1 + 3 * ((UAp3 + UAp4)/2) + MMd1)/5) \quad (1)$$

, i.e., in this hypothetical case, usability (UAp-3 and UAp-4) is three times more relevant than conceptual suitability (CCp1) and maintainability (MMd1).

From this point, we elaborated a detailed specification of the MQEP, including information about MQR, quality measures, measurement functions, the correct interpretation of measurement results, and the SoPaMM metamodel's specifications. All this body of information was organized into a simple schedule with the resources available (i.e., evaluators and tool support).

Table 3 shows the calculus of each measure based on the SoPaMM's documentation and a non-subjective interpretation of each measurement value. Based on this information, SoPaMM's final quality score is 9.2, which was carefully revised by the evaluators later.

## 5 CONCLUSIONS AND FUTURE WORK

We target a general solution for metamodel quality evaluation, most influenced by general standards. The idea is to establish a general quality approach which later, based on our experimentation, can also be instantiated for a more specific context. We think having a general quality evaluation model can make even different metamodels comparable through some general quality metrics.

It is noteworthy that the MQuaRE requirements, model, and measures are flexible and, thus, can be adapted to the specific contexts of a metamodel. That is, new characteristics and sub-characteristics can be included in the MQuaRE's quality model as well as corresponding measures and requirements. The MQuaRE framework should serve as a guide for metamodel quality evaluation, but it should not hamper execution.

We have already finished the development of the whole framework and several supporting documents to facilitate MQuaRE usage in practice. For this, we make available the full documentation of MQuaRE [7], including a detailed description of the 19 quality requirements and 23 quality measures, and the specific contributions from related work and international quality standards.

The validation of the MQuaRE framework is a work in progress so that we comprehend how much consistent MQuaRE is regarding its requirements. Soon, MQuaRE will also be evaluated through a survey with experts in metamodeling or metamodel quality. A tool to support the MQuaRE framework is also under development.

**Table 1: Evaluation of the SoPaMM metamodel: the correspondence among MQR, measures, and quality model.**

| Requirement | Measure | Sub-characteristic | Characteristic |
|---|---|---|---|
| MQR2 | Conceptual coverage | Conceptual completeness | Conceptual Suitability |
| MQR7 | Evident concepts | Appropriateness recognizability | Usability |
| MQR8 | Concept understandability | Appropriateness recognizability | Usability |
| MQR13 | Conceptual stability | Modifiability | Maintainability |

**Table 2: Evaluation of the SoPaMM metamodel: quality measures' id, name, description, and measurement function.**

| ID | Measure | Measure description | Measurement function |
|---|---|---|---|
| CCp-1 | Conceptual coverage | What proportion of the specified concepts has been modeled? | X = 1 - A / B<br>A = No. of missing concepts<br>B = No. of concepts described in the specification<br>0 <= X <= 1 (the closer to 1, the more complete) |
| UAp-3 | Evident concepts | What proportion of metamodel concepts are evident to the user? | X = A / B<br>A = No. of concepts evident to the user<br>B = No. of concepts described in the specification<br>0 <= X <= 1 (the closer to 1, the better) |
| UAp-4 | Concept understandability | What proportion of metamodel concepts are correctly understood by a first-time user without prior study or training or seeking external assistance? | X = A / B<br>A = No. of concepts correctly understood by the evaluator<br>B = No. of concepts described in the specification<br>0 <= X <= 1 (the closer to 1, the better) |
| MMd-1 | Conceptual stability | How stable is the metamodel specification during the development life cycle? | X = 1 - A / B<br>A = No. of concepts changed during development life cycle<br>B = No. of concepts described in the specification<br>0 <= X <= 1 (the closer to 1, the more stable) |

**Table 3: Evaluation of the SoPaMM metamodel: the respective results for each quality measure.**

| Measure | Measurement | Results interpretation |
|---|---|---|
| Conceptual coverage | A = 0; B = 20; **X = 1** | The SoPaMM's implementation covers all specified concepts |
| Evident concepts | A = 18; B = 20; **X = 0.9** | 90% of the SoPaMM's concepts are evident to the user |
| Concept understandability | A = B = 20; **X = 1** | A user with no prior help correctly understands all of SoPaMM's concepts |
| Conceptual stability | A = 5; B = 20; **X = 0.75** | 75% of the SoPaMM's concepts remain stable after changes |

## ACKNOWLEDGMENTS

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. The authors thank reviewers for their valuable comments.

## REFERENCES

[1] Markus Herrmannsdorfer and Guido Wachsmuth. 2014. *Evolving Software Systems.* Springer, Berlin, Heidelberg, Chapter Coupled Evolution of Software Metamodels and Models, 33–63.

[2] ISO/IEC. 2001. ISO/IEC 9126-1:2001 Software engineering — Product quality — Part 1: Quality model. *ISO/IEC 9126-1:2001* 1 (2001), 1–25.

[3] ISO/IEC. 2003. ISO/IEC TR 9126-3:2003 Software engineering — Product quality — Part 3: Internal metrics. *ISO/IEC 9126-3:2003* 2 (2003), 1–62.

[4] ISO/IEC. 2014. ISO/IEC 25000:2014 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE. *ISO/IEC 25000:2014* 2 (2014), 1–27.

[5] ISO/IEC. 2016. ISO/IEC 25023:2016 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of system and software product quality. *ISO/IEC 25023:2016* 1 (2016), 1–45.

[6] Taciana Novo Kudo, Renato F. Bulcão Neto, and Auri M. R. Vincenzi. 2019. A Conceptual Metamodel to Bridging Requirement Patterns to Test Patterns. In *Proceedings of the XXXIII Brazilian Symposium on Software Engineering.* ACM, New York, NY, USA, 155–160. https://doi.org/10.1145/3350768.3351300

[7] Taciana Novo Kudo, Renato F. Bulcão-Neto, and Auri M. R. Vincenzi. 2020. Metamodel Quality Requirements and Evaluation (MQuaRE). arXiv:2008.09459

[8] Haohai Ma, Weizhong Shao, Lu Zhang, Zhiyi Ma, and Yanbing Jiang. 2004. Applying OO Metrics to Assess UML Meta-models. In *UML 2004 — The Unified Modeling Language. Modeling Languages and Applications*, Thomas Baar, Alfred Strohmeier, Ana Moreira, and Stephen J. Mellor (Eds.). Springer, Berlin, Heidelberg, 12–26.

[9] Zhiyi Ma, Xiao He, and Chao Liu. 2013. Assessing the quality of metamodels. *Frontiers of Computer Science* 7, 4, Article 558 (2013), 12 pages.

[10] OMG. 2002. Meta Object Facility (MOF) Specification, Version 1.4. *Object Management Group, Inc.* (2002).

[11] Juri Rocco, Davide Di Ruscio, Ludovico Iovino, and Alfonso Pierantonio. 2014. Mining metrics for understanding metamodel characteristics. In *Proceedings of the 6th International Workshop on Modeling in Software Engineering (MiSE 2014).* ACM, New York, NY, USA, 55–60.

[12] Vjeran Strahonja. 2007. The Evaluation Criteria of Workflow Metamodels. In *29th International Conference on Information Technology Interfaces.* IEEE, New York, NY, USA, 553–558.

[13] Éric Vépa, Jean Bézivin, Hugo Bruneliere, and Frédéric Jouault. 2006. Measuring model repositories. In *Model Size Metrics Workshop - a MODELS 2006 Satellite Event.* Springer, Genoa, Italy, 1–5.

[14] James Williams, Athanasios Zolotas, Nicholas Matragkas, Louis Rose, Dimitios Kolovos, Richard Paige, and Fiona Polack. 2013. What do metamodels really look like?. In *Proceedings of the 3rd International Workshop on Experiences and Empirical Studies in Software Modeling.* CEUR-WS, Miami, USA, 55–60.