Measuring the Evolution of Meta-models - A Case Study of Modelica and UML Meta-models

Maxime Jimenez[1], Darko Durisic[2] and Miroslaw Staron[3]
[1]*ENSICAEN Engineering School, Caen, France*
[2]*Volvo Car Group, Gothenburg, Sweden*
[3]*Chalmers | University of Gothenburg, Gothenburg, Sweden*

Keywords: Meta-models, Evolution, Measurement.

Abstract: The evolution of both general purpose and domain-specific meta-models and its impact on the existing models and modeling tools has been discussed extensively in the modeling research community. To assess the impact of domain-specific meta-model evolution on the modeling tools, a number of measures have been proposed by Durisic et al., *NoC* (Number of Changes) being the most prominent one. The proposed measures are evaluated on a case of AUTOSAR meta-model that specifies the language for designing automotive system architectures. In this paper, we assess the applicability of these measure and the underlying data-model for their calculation in a case study of Modelica and UML meta-models. Our preliminary results show that the proposed data-model and the measures can be applied to both analyzed meta-models as we were able to capture 68/77 changes on average per Modelica/UML release. However, only a subset of the data-model elements is applicable for analyzing the evolution of Modelica and also certain transformation of the data-model is required in case of UML. Despite these encouraging results, further studies are needed to assess the usefulness of the actual measures, e.g., *NoC*, in assessing the impact of Modelica/UML meta-model evolution on the modeling tools.

1 INTRODUCTION

Meta-models are nowadays widely used in industry to define languages for domain-specific models used in the design of large software systems. These models represent instances of domain-specific meta-models, that in turn could also be instances of existing general-purpose meta-models, e.g. UML. The relationship between domain-specific models, meta-models and general-purpose meta-models is commonly referred to as the meta-modeling hierarchy of MOF (Meta-Object Facility) (MOF, 2004).

In order to be able to use new features in the models, e.g., new modeling elements, domain-specific meta-models need to be updated first in order to define the new elements and their relationships. In industry, the impact of adapting new meta-model releases in the development projects can be time-consuming due to its impact on the existing models and modeling tools (Staron and Wohlin, 2006). Therefore, an early indicator of such impact can be beneficial for the software designers in order to make the right decision of adopting a new meta-model release or accepting the limitations of using the old one.

The study of Durisic et al. (Durisic et al., 2014) in the automotive industry aimed to provide such an indicator based on the defined measures of domain-specific meta-model change, *NoC* (Number of Changes) being the most prominent measure. These measures were developed for the AUTOSAR meta-model, that represents a domain-specific meta-model used in the design of the automotive software architectures, and they are based on the data-model specifically designed for this purpose. The AUTOSAR meta-model represents an instance of UML (Durisic et al., 2016).

In this paper, we aim to assess the applicability of the proposed measures for analyzing the evolution of the AUTOSAR meta-model on others meta-models, both domain-specific and general-purpose. We achieve this goal in a case study of two additional meta-models: Modelica (Modelica, 2014) and UML meta-model. Modelica meta-model is used to define the language for modeling complex systems containing mechanical, electrical, electronic, and other components. UML is a well known general-purpose meta-model that defines the language that can be used for modeling a variety of different domains.

Jimenez M., Durisic D. and Staron M.
Measuring the Evolution of Meta-models - A Case Study of Modelica and UML Meta-models.
DOI: 10.5220/0006218204960502
In *Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2017)*, pages 496-502
ISBN: 978-989-758-210-3

Our research question is defined as: *What is the level of applicability of the measures of domain-specific meta-model evolution and the underlying data-model defined in (Durisic et al., 2014) for monitoring the evolution of Modelica/UML meta-models?*

The results of our study show that the analyzed data-model used for the definition of *NoC* and other measures described in Section 2 can be applied to the meta-models of Modelica and UML. Nevertheless, certain data-model elements and attributes are not applicable in case of Modelica, in particular tagged values, stereotypes and UUIDs (unique identifiers), and also the definition of connectors in UML had to be reworked in order to fit the proposed data-model. We also concluded that the *NoC* measure together with other proposed measure can be used for visualizing the evolution of Modelica and UML meta-models.

In conclusion, despite the fact that different meta-models require slightly different data-models for the definition of measures of their evolution, it is generally possible to use the data-model and measures based on it proposed by (Durisic et al., 2014) for measuring the evolution of both domain-specific and general-purpose meta-models. However, validation that the results of the proposed measures, in particular *NoC*, can indeed be used as a preliminary indicator of impact of adapting new meta-model versions in the development projects is yet to be performed.

The rest of this paper is structured as follows: Section 2 presents the data-model and the measures proposed for measuring the evolution of the AUTOSAR meta-model; Section 3 describes the steps we followed in assessing the applicability of the proposed measures for measuring the evolution of Modelica and UML meta-models; Section 4 discusses the applicability of the data-model in case of Modelica and UML and shows the results of calculating the proposed measures; Section 5 briefly describes the validation of the measurements result and discusses general assumptions taken in this study and their consequences; finally, Section 7 concludes our studies and presents guidelines for future work in the field.

2 BACKGROUND

In order to identify different types of AUTOSAR meta-model changes, a data-model presented in Figure 1 has been defined in (Durisic et al., 2014).

*MetaModel*s consist of a number of *Package*s that are used to logically group meta-classes and meta-objects (referred to as *Element*s) based on their semantics. *Element*s may be connected using a number of *Connector*s, e.g., associations and inheritance

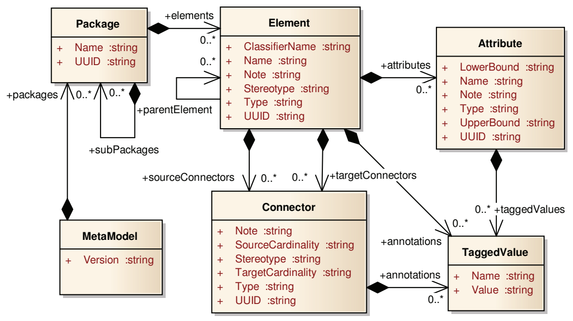


Figure 1: Data-model for AUTOSAR meta-model changes.

*Connector*s (in case of meta-classes). Additionally, *Element*s that represent meta-classes may contain arbitrary number of *Attribute*s. Finally, *Element*s, *Attribute*s and *Connector*s may have additional semantics captured in the *TaggedValue*s, e.g., note and origin of *Element*s.

Based on the defined data model, the following four main types of changes between different versions of the AUTOSAR meta-model are considered (changes to *Package*s are not considered):

1. Added, removed or modified *Element*.
2. Added, removed or modified *Attribute*.
3. Added, removed or modified *Connector*.
4. Added, removed or modified *TaggedValue*.

Based on these types of changes, the following measures of meta-model change are defined for monitoring the evolution of meta-models:

1. **NoE** (*Number of Elements*) in one meta-model release.
2. **NoME** (*Number of Modified Elements*) between two meta-model release.
3. **NoAE** (*Number of Added Elements*) between two meta-model release.
4. *NoRE* (*Number of Removed Elements*) between two meta-model release.
5. **NoC** (*Number of Changes*) between two meta-model release.

Measures calculating the total number of modified, added and removed *Attribute*s, *Connector*s and *TaggedValue*s were not considered critical for monitoring the evolution of domain-specific meta-models as *Element*s represent the main building blocks for supporting and using new features in the modeling tools and existing models. However, changes to *Attribute*s, *Connector*s and *TaggedValue*s are included in the *NoC* measure that captures all atomic changes to the elements and their sub-elements of the data-model (counted as one change), except for *Package*s.

3 RESEARCH METHOD

In order to provide the answer to our research question, we performed two case studies according to the protocol described by (Runeson et al., 2012) using Modelica and UML meta-models as a unit of analysis, and calculated the proposed measures of meta-model change. We performed the following steps:

Step 1: Data-model applicability assessment consists of a detailed study of the Modelica and UML meta-models using the available literature and manuals and comparison of these meta-models to the AUTOSAR meta-model, that served as bases for the definition of the data-model used for calculating *NoC* and other measures. We focused on identifying differences in the structure of the analyzed meta-models and how to transform the meta-models of Modelica and UML, without changing their semantics, so that they represent instances of the *NoC* data-model.

Step 2: Development of meta-model parser involves the development of a software tool for extracting the relevant information from the Modelica and UML meta-models specified in a proprietary format, to a universal format that could serve as input to another software tool that is used for calculating *NoC* and other measures for different meta-model versions. We chose XML format due to its wide acceptance in the modeling community. Therefore, all analyzed versions of the Modelica meta-model were parsed into our XML format. All analyzed versions of the UML meta-model can already be found in the XML format, however because they do not follow the same schema, we had to transform them into our XML format.

Step 3: Calculation of the measures involves the development of software tools similar to ARCA (Durisic et al., 2015), that is used for measuring the evolution of the AUTOSAR meta-model, for measuring the evolution of Modelica and UML meta-models. One difference between our tools and ARCA is that our tools use XML representation of the meta-model versions as input instead of Enterprise Architect models used by ARCA. The tools read two versions of Modelica/UML meta-model into instances of the data-model defined in the first step and calculate the proposed measures between them. We developed two separate tools for UML and Modelica because certain features of the tool were only applicable to one of them (e.g. meta-model extraction for Modelica).

The first tool supports Modelica meta-model releases *1.9.2* to *1.9.6* defined in the ”OMCompiler” folder (5 releases, also referred to as MetaModelica (Fritzson and Pop, 2011)). We were not able to determine the total coverage of Modelica meta-model releases as this historical information was not available due to format change. The second tool supports UML meta-model releases *2.1.1* to *2.4.1* (6 releases which represents 60% of UML's meta-models). As UML meta-model releases before 2.0 (i.e., 1.X) and release *2.5* use different identifiers and names for the same data-model elements, we decided to exclude them from the analysis as they would result in added and removed elements only. The UML meta-model version *2.0* in the XML format was unfortunately not available at the OMG website.

Based on the suggestion of Cook and Campbell (Cook and Campbell, 1979), we briefly discuss in this section possible threats to internal, external, construct and conclusion validity of our study:

Internal validity is concerned with the results of the analysis not being accidental. The internal validity of our study can be mostly threatened by possible defects in the implemented software tools for parsing Modelica/UML meta-models and calculating the change measures. To minimize this threat, we performed manual testing of the tools using small examples of the meta-models and compared the results with the results of the manually calculated measures.

External validity is concerned with the generalization of results. The purpose of this study was to assess the generalizability of the results of previous studies of meta-model evolution on a broader scope of meta-models. As this paper shows, the proposed measures of meta-model evolution could be calculated on two additional meta-models of Modelica and UML, using certain number of transformations to the underlying data-model or just using its subset. However, other meta-models may bring additional concept (e.g. primitive type variations) that may require extension of the data-model, thus reducing its generalizability.

Construct validity is concerned with the mismatch between theory and observations. In our study, this concerns the ability of the analyzed data-model to capture variations between different meta-model versions. As the purpose of our study was to assess the applicability of the already defined data-model, this part of the construct validity is implicitly achieved. However, the threat related to the necessary transformations of the UML meta-model to represent an instance of the *NoC* data-model still remains.

Conclusion validity is concerned with the degree to which the conclusions of the study are reasonable. This is considered high for our study due to the fact that we were able to calculate the measures of meta-model change on the evolution of Modelica and UML meta-models. However, the usefulness of the measures in monitoring the evolution of Modelica/UML is yet to be assessed in future work.

4 RESULTS

In the results section, we first show the data-models that we used for calculating *NoC* and other measures between different releases of Modelica and UML meta-models, and discuss their similarity and differences to the AUTOSAR data-model presented in Figure 1. We then show the results of calculating the measures for a set of Modelica and UML releases.

4.1 Modelica Data-model

For calculating the measures between different releases of the Modelica meta-model, we used a subset of the AUTOSAR data-model presented in Figure 2.

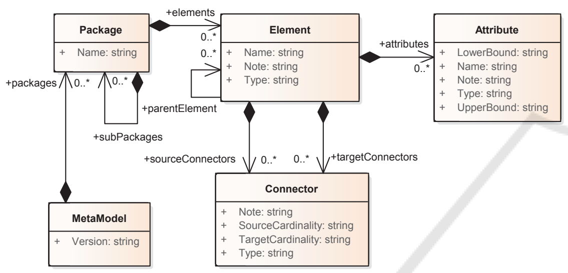


Figure 2: Modelica data-mode.

The relationship between one *Package* and its *subPackages* has been kept in case of future use, but it has not been used for the measurements because the Modelica meta-model does not contain any subpackages. Additionally, the *Element*s, *Attributes* and *Connector*s in Modelica do not contain any additional semantics captured in a form of *TaggedValues*, nor do they have *Stereotype*s and *UUID*s (unique identifiers). Therefore, we did not use these elements from the AUTOSAR data-model for the measurement.

4.2 UML Data-model

For calculating the measures between different releases of the UML meta-model, we used a subset of the AUTOSAR data-model presented in Figure 3.

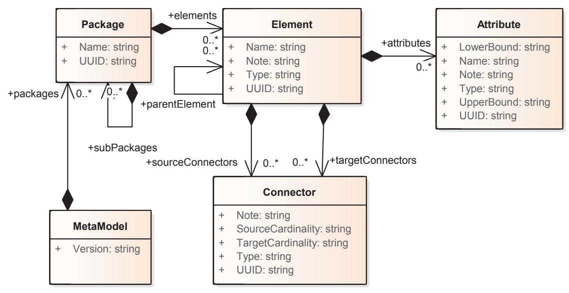


Figure 3: UML data-mode.

After the initial inspection of the UML meta-model, we identified a number of similarities but also a few differences in comparison to the AUTOSAR meta-model. The biggest difference is related to the definition of connectors (referred to as *Association*s in UML) that are owned by *Package*s instead of *Element*s. However, the definition of *Associations* is also present in the *Element*s in a form or *Property*s (*attributes*) that are referred to by two *associationEnds*.

The compliance of the UML and AUTOSAR meta-models can be achieved with one transformation of the UML meta-model. The transformation implies aggregating the UML *Association* twice by the two *Element*s connected by this *Association*, as shown in Figure 4. As the AUTOSAR meta-model represents an instance of UML (UML model), this transformation was already performed in case of AUTOSAR by the UML modeling tool used in the development of the AUTOSAR meta-model (Enterprise Architect), i.e., UML *Association* aggregated by *Package*s are instead aggregated by the connected *Element*s.

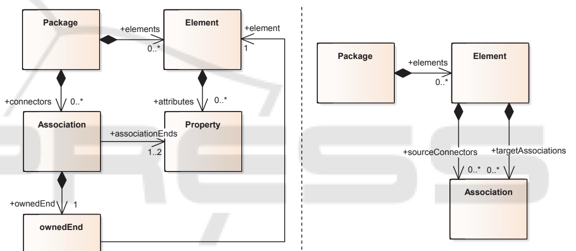


Figure 4: UML (left) and AUTOSAR (right) meta-model.

In addition to this, *Tagged Value*s and *Stereotype*s of *Elements*, *Attribute*s and *Connector*s are not applicable in case of UML meta-model as it defines them for the instantiating UML models such as AUTOSAR meta-model. Despite these differences, the similarity between the AUTOSAR and UML meta-models is higher than between the AUTOSAR and Modelica meta-models. This is expected as the AUTOSAR meta-model represents an instance of UML.

4.3 Modelica Measurements

In order to visualize the size of Modelica meta-model and its evolution, we calculated the Number of Elements (*NoE*) in the analyzed Modelica releases. The results are presented in Figure 5.

We can see that the *NoE* slightly increases between releases *1.9.2* and *1.9.4*, while it remains stable between releases *1.9.4* and *1.9.6*. The increase in the number of elements in releases *1.9.3* and *1.9.4* is related to the addition of new packages in the meta-model, e.g. *Dump* and *StateMachineFlatten*.

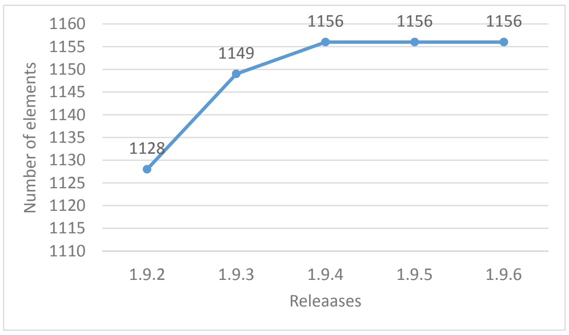
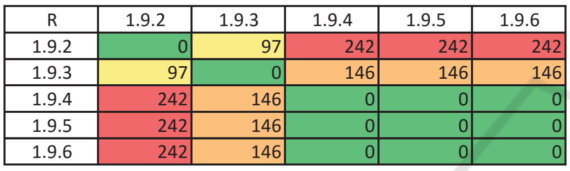


Figure 5: *NoE* in different Modelica meta-model releases.

In order to dive deeper into the analysis of Modelica meta-model changes, we calculated the results of the *NoC* measure. Figure 6 shows the *NoC* between each two meta-model releases.



R	1.9.2	1.9.3	1.9.4	1.9.5	1.9.6
1.9.2	0	97	242	242	242
1.9.3	97	0	146	146	146
1.9.4	242	146	0	0	0
1.9.5	242	146	0	0	0
1.9.6	242	146	0	0	0

Figure 6: *NoC* between Modelica meta-model releases.

As expected, the wider the gap between releases is, the bigger number of changes we have. Moreover, we can see that the *NoC* between consecutive releases is not very big (0 to 146 changes). We believe that this is due to the fact that we analyzed minor releases of the Modelica meta-model only.

Finally in order to identify the types of changes that are driving the evolution of the Modelica meta-model, we calculated the Number of Added Elements (*NoAE*), Number of Removed Elements (*NoRE*) and Number of Modified Elements (*NoME*) between the consecutive meta-model releases. Figure 7 shows the *NoAE* and *NoME* while the *NoRE* does not appear because no elements were removed between the analyzed versions (probably due to strong backwards compatibility requirements).

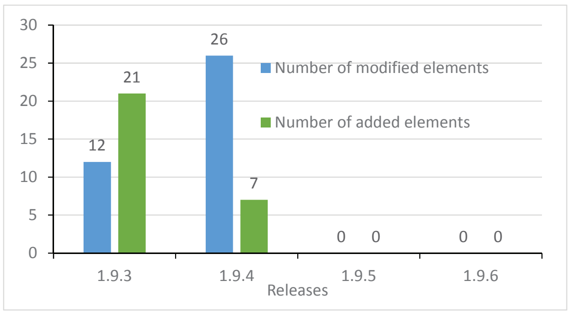


Figure 7: Modelica meta-model *NoAE* vs. *NoME*.

The figure shows that, apart from releases *1.9.2* and *1.9.3* where the *NoME* is equal to *NoAE*, in other releases changes are mostly driven by modifications of the existing elements.

4.4 UML Measurements

The size of the UML meta-model and its evolution, as a result of calculating the *NoE* of the analyzed releases of UML, is presented in Figure 8.

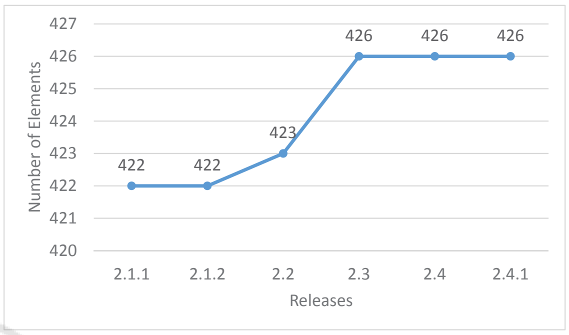
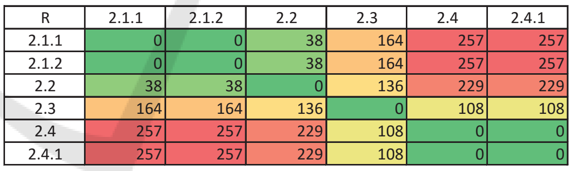


Figure 8: *NoE* in different UML meta-model releases.

We can see that the *NoE* slightly increases between releases *2.1.2* and *2.3* (4 new *Elements* are introduced, 1%), while it remains stable between releases *2.3* and *2.4.1*.

The results of the *NoC* measure between each two analyzed releases of the UML meta-model are presented in Figure 9.



R	2.1.1	2.1.2	2.2	2.3	2.4	2.4.1
2.1.1	0	0	38	164	257	257
2.1.2	0	0	38	164	257	257
2.2	38	38	0	136	229	229
2.3	164	164	136	0	108	108
2.4	257	257	229	108	0	0
2.4.1	257	257	229	108	0	0

Figure 9: *NoC* between UML meta-model releases.

As expected, the *NoC* increases gradually between releases that are further apart from each other (e.g., releases 2.1.1 and 2.4.1). Moreover, we can see that there are no changes between the minor releases of UML (change in the third digit), e.g., between releases 2.1.1 and 2.1.2.

Finally, Figure 10 shows the results of *NoAE*, *NoRE* and *NoME* measures between the consecutive releases of the UML meta-model.

The figure shows that the evolution of UML is mostly driven by the modification of existing elements. We can also verify that the change in the *NoAE* and *NoRE* matches the evolution of the *NoE* presented in Figure 8 (e.g., 6 new elements and 3 removed elements between releases 2.2 and 2.3 yields 3 more elements in release 2.3).

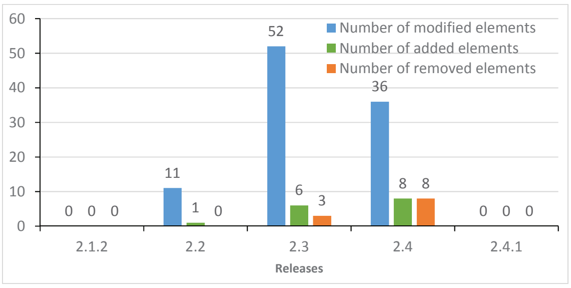


Figure 10: UML meta-model *NoAE* vs. *NoRE* vs. *NoME*.

5 VALIDATION AND DISCUSSION

There are five important points from this study that require further discussion. First, quantitative analysis is usually an efficient approach for making preliminary assessments of certain phenomenon with low cost. However, deeper qualitative analysis is usually expected to follow for more precise assessment. In our case, *NoC* can be used for initial understanding of the amount of changes between different meta-model versions, but in order to understand their consequences, e.g., impact on the compliant models and modeling tools, analysis of the actual changes is needed. Therefore, tools that we developed are also able to present the list of Modelica/UML meta-model changes with their description. Additionally, role-based analysis of meta-model changes can be helpful in evaluating which roles/teams will be mostly affected by the adoption of a new meta-model version.

Second, as the data-model we used in this study has been designed for measuring the evolution of the AUTOSAR meta-model (Durisic et al., 2014), we could have designed a data-model that is more inline with the structure of the UML and Modelica meta-models. Nevertheless, we show that it is possible to use a unified data-model for measuring the evolution of multiple domain-specific and general-propose meta-models. This may, however, require certain transformations of the analyzed meta-models.

Third, we can observe that there are no changes in the Modelica meta-model between releases 1.9.4 and 1.9.6 and UML meta-model releases 2.1.1. and 2.1.2, and releases 2.4 and 2.4.1. In case of Modelica, this is because we analyzed only the ”OMCompiler” that contains the definition of the meta-model. Therefore there may still be changes affecting other parts of Modelica. In case of UML, this may be a sign that there may be additional properties contained in the meta-model that are not captured by our data-model.

Fourth, we made a clear separation between attributes and connectors in a way that attribute is of primitive type while connector is of element (meta-class) type. Therefore, it would then make sense to include a list of primitive types (e.g. *String*) in a meta-model instead of just identifying them while parsing the meta-model into an instance of our data-model.

Finally, referring to UML as a general-purpose meta-model is usually correct. However when it comes to Modelica, it depends on the interpretation. A mechatronics engineer may refer to it as a general-purpose while a software engineer may refer to it as a domain-specific meta-model used for modeling systems with electrical and mechanical components.

6 RELATED WORK

A number of studies focus on automated model transformations, e.g., Becker et al. (Becker et al., 2007) present a process model for updating model instances according to the classification of meta-model changes. Sprinkle et al. (Sprinkle and Karsai, 2004) developed a visual language for describing the evolution of domain-specific meta-models that can also perform model transformations for meta-model updates. Vara et al. present a process for automated generation of model transformations using a set of modeling tools that can be used for mitigating existing models from one meta-model release to another (Vara et al., 2008). Kuzniarz et al. (Kuzniarz and Staron, 2003) describe UML transformations and define a systematic way of transforming UML meta-model, similar to our work presented in Figure 4.

Related to our data-model, Wachsmuth presents transformation library for stepwise adaptation of MOF compliant meta-models, that involves refactoring, construction and destruction of the meta-model elements and their properties and relations, and co-adaptation of the complaint models (Wachsmuth, 2007). Cicchetti et al. (Cicchetti et al., 2007) describe a language for comparing different model and meta-model versions based on the ”difference meta-model” that is derived from the analyzed model. This data-model is also able to describe additions, deletions and changes to the model elements, but it is too general for defining changes relevant for architectural domain-specific meta-models.

A number of studies also analyze the applicability of different metrics for measuring certain properties of meta-models such as the studies of Ma et al. (Ma et al., 2013) and Di Rocco et al. (Rocco et al., 2014) that focus on the model evolution. Combining these metrics with the measures of meta-model change, e.g., *NoC*, could be used to better assess the evolution of domain-specific (meta-)models.

7 CONCLUSION

The main goal of our study was to assess the applicability of the measures of domain-specific meta-model evolution, such as *NoC*, on other domain-specific and general-purpose meta-models. In order to achieve our goal, we analyzed the applicability of the data-model behind these measures for measuring the evolution of Modelica and UML meta-models and calculated the measures on a set of chosen UML/Modelica releases.

As the answer to our research question, the results of this paper show high applicability of the *NoC* and other measures for measuring the evolution of both Modelica and UML meta-models. However, certain modifications to the data-model used for calculating the measures needed to be made. In particular, a subset of the data-model elements was taken in both cases of Modelica and UML, e.g., *TaggedValue*s and *Stereotype*s were excluded. Additionally, UML meta-model required a transformation related to the definition of UML *Association*s (*Connector*s) that need to be aggregated by the connected *Element*s instead of *Package*s. Nevertheless, the semantics of the meta-models was not changed and the data-model and measures could have been applied without modifications.

The thing this paper did not investigate is the actual benefit of the measurement results, i.e., using *NoC* for early estimation of impact of adopting new releases of Modelica and UML meta-model on the modeling tools and existing models in the development projects. This study represents a natural continuation of the presented work and could be extended in future by measuring the evolution of specific packages of Modelica, UML, or other meta-models related to different design roles in the development process.

ACKNOWLEDGEMENTS

The authors would like to thank Swedish Governmental Agency for Innovation Systems (VINNOVA) for funding this research (grant no. 2013-02630), and Adrian Pop from Linköping University who helped us with the access to the Modelica meta-model.

REFERENCES

Becker, S., Gruschko, B., Goldschmidt, T., and Koziolek, H. (2007). A Process Model and Classification Scheme for Semi-Automatic Meta-Model Evolution. In *Workshop on MDD, SOA und IT-Management*, pages 35–46.

Cicchetti, A., Ruscio, D. D., and Pierantonio, A. (2007). A Metamodel Independent Approach to Difference Representation. *Journal of Object Technology*, 6(9):165–185.

Cook, T. and Campbell, D. (1979). *Quasi-Experimentation: Design & Analysis Issues for Field Settings*. Houghton Mifflin.

Durisic, D., Staron, M., and Tichy, M. (2015). ARCA - Automated Analysis of AUTOSAR Meta-Model Changes. In *Workshop on Modelling in Software Engineering*, pages 30–35.

Durisic, D., Staron, M., Tichy, M., and Hansson, J. (2014). Evolution of Long-Term Industrial Meta-Models - A Case Study of AUTOSAR. In *Conference on Software Engineering and Advanced Applications*, pages 141–148.

Durisic, D., Staron, M., Tichy, M., and Hansson, J. (2016). Addressing the Need for Strict Meta-Modeling in Practice - A Case Study of AUTOSAR. In *Conference on Model-Driven Engineering and Software Development*, pages 317–322.

Fritzson, P. and Pop, A. (2011). Meta-Programming and Language Modeling with MetaModelica 1.0. Technical report, Dept. of Computer and Information Science, Linkping University.

Kuzniarz, L. and Staron, M. (2003). On Model Transformations in UML-Based Software Development Process. In *Conference on Software Enginnering and Applications*, pages 391–395.

Ma, Z., He, X., and Liu, C. (2013). Assessing the Quality of Metamodels. *Journal of Frontiers of Computer Science*, 7(4):558–570.

Modelica (2014). *Modelica Language Specification v3.3.1*. Modelica Association, www.modelica.org.

MOF (2004). *MOF 2.0 Core Specification*. Object Management Group, www.omg.org.

Rocco, J. D., Ruscio, D. D., Iovino, L., and Pierantonio, A. (2014). Mining Metrics for Understanding Metamodel Characteristics. In *Workshop on Modeling in Software Engineering*, pages 55–60.

Runeson, P., Höst, M., Rainer, A., and Regnell, B. (2012). *Case Study Research in Software Engineering: Guidelines and Examples*. John Wiley & Sons.

Sprinkle, J. and Karsai, G. (2004). A Domain-Specific Visual Language for Domain Model Evolution. *Journal of Visual Languages & Computing*, 15(3):291–307.

Staron, M. and Wohlin, C. (2006). An Industrial Case Study on the Choice Between Language Customization Mechanisms. In *Conference on Product-Focused Software Process Improvement*, pages 177–191.

Vara, J., Fabro, M. D., Jouault, F., and Bézivin, J. (2008). Model Weaving Support for Migrating Software Artifacts from AUTOSAR 2.0 to AUTOSAR 2.x. In *European Congress on Embedded Real Time Software*.

Wachsmuth, G. (2007). Metamodel Adaptation and Model Co-adaptation. In *European Conference on Object-Oriented Programming*, pages 600–624.