2010 34th Annual IEEE Computer Software and Applications Conference Workshops

The Right Tool for the Right Job: Assessing Model Transformation Quality

M.F. van Amstel
Eindhoven University of Technology
P.O. Box 513, 5600 MB, Eindhoven
The Netherlands
M.F.v.Amstel@tue.nl

***Abstract*—Model-Driven Engineering (MDE) is a software engineering discipline in which models play a central role. One of the key concepts of MDE is model transformations. Because of the crucial role of model transformations in MDE, they have to be treated in a similar way as traditional software artifacts. They have to be used by multiple developers, they have to be maintained according to changing requirements and they should preferably be reused. It is therefore necessary to define and assess their quality. In this paper, we give two definitions for two different views on the quality of model transformations. We will also give some examples of quality assessment techniques for model transformations. The paper concludes with an argument about which type of quality assessment technique is most suitable for either of the views on model transformation quality.**

***Keywords*-Model-Driven Engineering, Model Transformation, Quality, Metrics**

I. Introduction

Model-Driven Engineering (MDE) is a software engineering discipline in which models play a central role throughout the entire development process [1]. MDE combines domain-specific modeling languages for modeling (software) systems, and model transformations for, among others, synthesizing them. A domain-specific language (DSL) is a language that offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain [2]. A DSL enables software engineers to model using domain concepts rather than concepts provided by existing formalisms, which typically do not provide the required or correct abstractions. Model transformations are used to automatically transform models expressed in these DSLs into different models. These models can either be expressed in the same formalism or in a different formalism, such as a formalism for model verification or code. Using model transformations, a model specified in a DSL can be used for various different purposes. These model transformations can be reused for every model specified using that DSL. This *as-is* type of reuse is mentioned in [3] for application generators such as model transformations in general, i.e., they are typically implemented once and reused often. Although as-is reuse is most prominent in the context of model transformations, *reuse-with-modify* also plays an important role. As the need for a new target platform arises, which can be the case if for example Java code is required instead of C code, new model transformations will have to be developed. In such cases it is desirable to reuse as much as possible from existing model transformations. Not only may the target platforms be subject to change. The source formalism, i.e., the DSL, may evolve for various reasons as well [4]. When a DSL evolves, the accompanying model transformations should evolve accordingly. To facilitate this, it is important that model transformations are reusable and maintainable. It is therefore necessary to define and assess their quality [5], [6].

In this paper, we will distinguish two views on model transformation quality, viz., internal and external quality. We will also define two different ways of assessing model transformation quality, viz., directly and indirectly. We will give examples of techniques for both direct and indirect quality assessment. Finally, we will explain advantages and disadvantages of these measurement techniques for both internal and external model transformation quality.

II. Quality of Model Transformations

A. Preliminaries

Figure 1 schematically depicts the context of a model transformation. Formal model M is the source model of a model

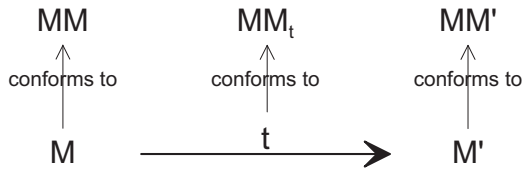

Figure 1. Model transformation

transformation $\mathtt{t}$ that generates formal model M' as target model. Model M conforms to metamodel MM and model M' conforms to metamodel MM', where possibly $\mathsf{MM} = \mathsf{MM}'$. Model transformation $\mathtt{t}$ itself can also be considered as a formal model that conforms to metamodel $\mathsf{MM}_\mathtt{t}$. A metamodel defines the concepts and relations between these concepts for creating a class of models. In fact, a metamodel is a model of a modeling language [7].

There are many different definitions around for what a model is. The one used in this paper is the following. *A model is an abstraction of an entity with a specific purpose.* This is a broad definition that allows us to consider artifacts such as source code as models as well. Using this

978-0-7695-4105-1/10 $26.00 © 2010 IEEE
DOI 10.1109/COMPSACW.2010.22

definition, we also cover a broad range of different types of model transformations, viz., model-to-model transformations, text-to-model transformations, model-to-text transformations, and text-to-text transformations. Note that we also do not distinguish between modelware and grammarware [8].

B. Quality of Model Transformations

Before explaining how the quality of model transformations can be assessed, we first need to define what quality is in the context of model transformations. The quality of model transformations can be viewed in two ways. Therefore, we provide two definitions.

The *internal quality* of a model transformation is the quality of the transformation artifact itself. The internal quality of a model transformation is of importance when a model transformation needs to be reused or maintained. The quality attributes that describe the internal quality of a model transformation we consider are: *understandability*, *modifiability*, *reusability*, *modularity*, *completeness*, *consistency*, and *correctness*. Most of these quality attributes have already been defined earlier for software artifacts in general [9], [10]. In [11], we explain why they are relevant for model transformations in particular. Note that in this paper we will focus on quality attributes related to maintainability and reusability only. Other quality attributes, such as for example performance, are not considered.

The *external quality* of a model transformation is the quality change induced on a model by the model transformation. The external quality of a model transformation has a determining influence on the quality of the output model of a model transformation. The quality attributes that are of importance depend on the way the output model needs to be used in the remainder of the development process. If the model is needed for manual processing, e.g., as documentation, quality attributes such as understandability and modifiability are relevant. These quality attributes may, however, be completely irrelevant when the model needs to be used as input for another model transformation only.

The research presented in this paper focusses on the internal quality of model transformations. However, we will also address issues regarding the assessment of their external quality.

III. Quality Assessment of Model Transformations

Various techniques exist for assessing the quality of various different kinds of models. Here we list just a few of them. In [12], a framework is described for evaluating the quality of models in general. Techniques tailored towards assessing the quality of UML models using, among others, metrics can be found in [13], [14]. In [15], techniques are described for analyzing different characteristics of software using metrics. Such techniques can be used to assess the quality of the input model and the output model of a model transformation.

One way of measuring the (internal or external) quality of model transformations is by comparing the quality of an input model with the quality of the output model generated from it. More formal, model transformation quality $Q = \bigoplus(f(\mathsf{M}), g(\mathsf{M}'))$, where $\bigoplus$ is a comparison operator and functions f and g are used to assess the quality of models M and M' respectively. The quality of the model transformation is not assessed directly, but by comparing its input and corresponding output. Therefore, we refer to this type of quality assessment as *indirect quality assessment*. In the previous paragraph, it was noted that metrics are often used for assessing the quality of models. Therefore, indirect quality assessment of model transformation will come down to comparing (aggregations of) metric values. In [16], indirect quality assessment is advocated for assessing the external quality of model transformations. The authors propose to extend the metamodels of the source and target models of a model transformation with metrics and their calculation methods. These metrics can then be used to assess the quality change of a model induced by a model transformation. In this way, the external quality of the model transformation is assessed.

Another way of measuring the (internal or external) quality of model transformations is by measuring the model transformation itself. More formal, model transformation quality $Q = f(\mathsf{t})$ for some function f. Since the quality of the model transformation is assessed directly, we refer to this type of quality assessment as *direct quality assessment*. In [5], we advocate this approach for assessing the internal quality of model transformations. We have defined a set of metrics for text-to-text model transformations defined using ASF+SDF. In Section III-B0a, we will elaborate on this.

A. Indirect Quality Assessment of Model Transformations

A taxonomy of model transformations is presented in [17]. The authors distinguish, among others, between endogenous and exogenous model transformations. In an endogenous model transformation, the source and target model are expressed in the same metamodel (or grammar), whereas the source and target model of an exogenous model transformation are expressed in a different metamodel (or grammar). Typical examples of endogenous model transformations are transformations for model refactoring and refinement. Typical examples of exogenous model transformations are code generators and transformations for migration. In the remainder of this section, we will use this distinction to elaborate on indirect quality assessment of model transformation using metrics. We will also elaborate on a form of quality assessment that is aimed at validating the correctness of a model transformation that does not involve metrics.

1) Exogenous Model Transformations: Indirect quality assessment of exogenous model transformations using metrics is hard, if not impossible. Suppose metrics can be extracted from both the source and the target model. This results in two sets of metrics that are possibly overlapping. If the sets of metrics do not overlap, it is hard, maybe even impossible, to select subsets of metrics that can be used for comparison. In this case, care has to be taken in selecting these subsets to ensure that the comparison is appropriate. If the sets of metrics are overlapping, comparison is still not straightfor-

ward. Similar metrics for different formalisms may require different interpretations. For example the scale, or the unit of a metric may be different. Again, care has to be taken that the metrics selected for comparison are comparable at all. In some cases, the source or target formalism of a model transformation inherently imply extreme values for certain metrics, e.g., when certain libraries are required. To ensure a fair comparison, these outlying values have to be corrected for. The risk in indirect quality assessment of exogenous model transformations is that it results in comparing apples and oranges. This holds for both internal and external quality.

2) Endogenous Model Transformations: Indirect quality assessment of endogenous model transformations using metrics is also not trivial. In case of endogenous model transformations, the formalism in which the source and target model are expressed is the same. Therefore the same measurement techniques can be used to assess both the quality of the source model and the quality of the target model. This means that the results of the quality assessments are comparable. However, this does not necessarily mean that they can be used to assess the internal quality of a model transformation.

Consider a model transformation that extends a model. The level of abstraction and the platform do not change in this case. This type of model transformation is called a horizontal model transformation [17]. Usually, model size has a negative correlation with understandability and modifiability. Note that this is not always the case since a large, but properly modularized model may in fact be more understandable than a smaller, but improperly modularized one. Typically, this means that extension of a model has a negative effect on the understandability and the modifiability of the model. However, this does not necessarily imply that the model transformation is hard to understand or to modify as well. In fact, the model transformation may be really simple. The opposite also holds. Consider a model transformation that removes elements from a model. Since model size has a negative correlation with understandability and modifiability, such a change implies that the understandability and the modifiability of the model increase. Again, it is not necessarily the case that the model transformation is easy to understand or to modify as well. It may very well be that a complex algorithm is required for determining which elements need to be removed. A similar story holds for vertical endogenous model transformations, i.e., model transformations in which the formalism does not change, but the level of abstraction does, such as refining or abstracting model transformations. Another, similar, example is the quality attribute completeness. The application of a model transformation to a model may result in an increase or decrease of the completeness of the model, but this does not necessarily imply that the model transformation itself is (in)complete. These examples illustrate that metrics used for evaluating certain quality attributes for a model, may be completely unrelated to similar quality attributes for a model transformation. Therefore, care has to be taken that appropriate metrics are selected for the comparison when assessing the internal quality of endogenous model transformations indirectly.

The examples in the previous paragraph do address the external quality of model transformation. Metric values acquired by indirect measurement can be compared to assess the quality change induced by a model transformation. The extending model transformation returns a target model that is less understandable and modifiable. This means that the quality change induced by the model transformation is negative. It therefore has low external quality. Something similar holds for the other examples as well. Therefore, for assessing the external quality of a model transformation, indirect measurement seems to be appropriate. Note, however, that indirect measurement is case specific, i.e., the outcome will be different for every pair of source model and resulting target model. This means that it may be hard to draw general conclusions about the external quality of a model transformation.

3) Validation of Model Transformations: There are also different approaches, i.e., not using metrics, in which the quality of a model transformation is assessed indirectly. In [18] and in [19], two similar approaches are discussed that use model checking to validate whether model transformations on behavioral models preserve certain behavioral properties. This type of validation can be used to get insight in the correctness of model transformations. A schematic overview of the approaches described in [18] and [19] is depicted in Figure 2. In both approaches, a source model M and the resulting

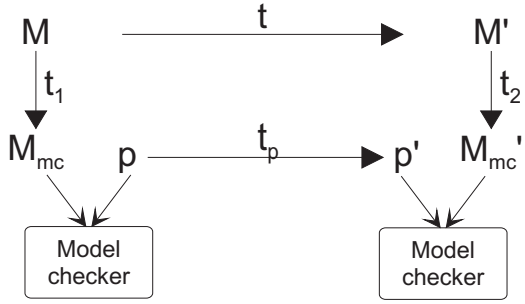

Figure 2. Model checking model transformations

target model M' of a model transformation t are transformed by model transformations t_1 and t_2 into models that can be used for model checking, M_{mc} and $M_{mc}{}'$ respectively. For the transformed source model M_{mc}, a property p is defined that it should satisfy. This can be verified by a model checker. This property is then transformed using another transformation t_p into an equivalent property p' that fits the transformed target model $M_{mc}{}'$. Again, a model checker can be used to verify whether model $M_{mc}{}'$ satisfies property p'. If this is the case, it can be concluded that model transformation t maintains the validity of property p.

The approach discussed in [18] works exactly as discussed. There are, however, a few threats to the validity of this approach. The reliability of the approach is affected by three model transformations, viz., model transformations t_1, t_2, and t_p. For the correctness of model transformations t_1 and t_2, the authors have a solution. For the correctness of

transformation t_p, a domain expert is required. The approach discussed in [19] assumes that model transformation t is an endogenous model transformation. This implies that the model transformations used for generating models M_{mc} and $M_{mc}{}'$ can be the same, i.e., $t_1 = t_2$. This also implies that the property that needs to be verified can remain the same in most cases, i.e., $p = p'$. Therefore the reliability of this method is affected by one model transformation only, i.e., the one that transforms the source and target model of model transformation t into the models suitable for model checking. However, it is applicable to endogenous model transformations only.

Note that verifying whether properties are maintained by a model transformation does not guarantee the correctness of that model transformation. It expresses that the model transformation maintains the properties that have been verified and for the models against which they have been verified only. Therefore, the strength of this method depends on the amount and quality of realistic models and properties that are verified. To guarantee full correctness of model transformations, other techniques should be used. In [20], an approach is described that uses a theorem prover to ensure semantic equivalence between a model and the code generated from it by a model transformation.

B. Direct Quality Assessment of Model Transformations

When using direct quality assessment of model transformations, the model transformation itself is measured. The metamodels (or grammars) are not taken into account. It is therefore irrelevant whether the model transformation is endogenous or exogenous. Hence, we will not make this distinction in this section. Metrics used for direct quality assessment of model transformations are not specific for a pair of metamodels, but specific for a model transformation formalism. Therefore, this methodology is more generally applicable. This also means that different model transformation formalisms require different metrics. In Section III-B0a we will see, however, that similar metrics have been defined for different model transformation formalisms. The metrics are specific for a model transformation formalism, but the model transformation itself is specific for its source and target metamodel. It is therefore to be expected that metric values depend on the metamodels. Researching the nature of this dependency is a point for future work.

Since the metamodels of the source models and the target models are not taken into account, it is hard to use direct quality assessment for assessing the external quality of a model transformation. Only very general observations can be made. Consider an endogenous model transformation. Suppose that a transformation rule replaces one model element in the source model by a number of model elements in the target model. Since this increases the size of the model, it is likely that understandability and modifiability of the model decrease. However, this does not necessarily have to be the case. It could be that only part of the model is transformed by the model transformation and that in fact the size of the model has decreased. Without knowledge of the involved metamodels it is impossible to detect this, since it cannot be determined from the model transformation alone what parts of the metamodels are covered by it.

a) Direct Quality Assessment of ASF+SDF Model Transformations: In [5], we presented an approach for direct assessment of the internal quality of model transformations using metrics. The study is focussed on model transformations implemented in ASF+SDF [21]. ASF+SDF is a term rewriting system that can be used to define modular text-to-text model transformations based on grammars using conditional rewrite rules. Three categories of metrics were defined, viz., function metrics, module metrics, and consistency metrics. The function metrics provide information about, among others, the size and interdependencies of the transformation functions that make up the model transformation. Examples of function metrics are: number of transformation functions, number of conditions per function, and function fan-in. Most of the function metrics can be aggregated to module level. This leads to module metrics as: number of transformation functions per module, and module fan-in. Other module metrics, such as number of import declarations, measure module interdependencies. The last category of metrics measure (in)consistencies (between modules) in a model transformation. An example of such a metric is: number of different types per variable name. Although most of the metrics that were defined for ASF+SDF may appear to be specific for that formalism, this is not the case. In two studies [22], [23] of the ATL model transformation formalism [24], similar metrics were defined.

The goal of the research is to determine which (sets of) metrics can be used for assessing the quality attributes that are relevant for model transformations. To obtain such a relation between metrics and quality attributes, we performed an empirical study. A tool was created that can automatically extract metrics from an ASF+SDF model transformation. This tool was used to extract metrics from a heterogeneous collection of model transformations. The same collection of model transformations was quantitatively evaluated by experts. We analyzed the correlations between the metrics data and the expert feedback. Significant correlations with metrics were found for almost all quality attributes.

From this study we can conclude what aspects of model transformations contribute to a higher or lower value for certain quality attributes. These metrics can be used to compare model transformations on their internal quality. For more information on this empirical study, the reader is referred to [5].

IV. Conclusions and Future Work

A. Conclusions

In this paper, we have addressed the need for techniques to analyze the quality of model transformations. We gave two definitions for model transformation quality, viz., internal quality and external quality. We also defined and gave examples of two different types of quality assessment techniques for model transformations, viz., direct assessment and indirect assessment. In theory, both assessment techniques can be used

to assess both types of quality. However, both techniques have their advantages and disadvantages.

Indirect quality assessment has as disadvantage that it cannot always be used. It may be the case that metrics for the source model and for the target model of a model transformation are incomparable. Also, indirect quality assessment provides a partial conclusion only, i.e., it is as good as the amount of cases that have been assessed. Direct quality assessment is specific for a model transformation formalism and not for a pair of metamodels. It is therefore more generally applicable.

Direct quality assessment should be used for assessing the internal quality of model transformations only. It does not make sense to use direct quality assessment for assessing the external quality of a model transformation, since the metamodels of the source and target models are not taken into account. Indirect quality assessment may be used for assessing the external quality of a model transformation, but only if comparison of the source and the target models is possible. Indirect quality assessment should in general not be used for assessing the internal quality of a model transformation. At least not using metrics, since metrics that are used for evaluating quality attributes for a model may be unrelated to the same quality attributes for model transformations. However, in Section 2 we showed an example of how the internal quality of model transformations can be assessed indirectly. In this case, no metrics were involved. Table I summarizes the conclusions.

		Assessment technique	
		Direct	Indirect
Quality Type	Internal	✓	Not using metrics
	External	×	if $f(\mathsf{M})$ and $g(\mathsf{M}')$ are comparable

Table I
TYPES OF QUALITY AND QUALITY ASSESSMENT TECHNIQUES

B. Future Work

Model transformations are in many ways similar to traditional software artifacts, i.e., they have to be used by multiple developers, they have to be maintained according to changing requirements and they should preferably be reused. Therefore, they need to adhere to similar quality standards. To achieve these standards, there should be a methodology for developing model transformations with high internal quality. This could be in the form of a set of guidelines which, if adhered to, lead to such high-quality model transformations. Before such guidelines can be defined, it is necessary to investigate what the key factors are that influence the internal quality of model transformations. Therefore it is desirable that the quality of model transformations created using a variety of formalisms is assessed. In [5] such techniques have been described for ASF+SDF, a grammarware formalism and in [22], [23] for ATL, a modelware formalism. We would like to investigate similar direct measurement techniques for different model transformation formalisms such as GrGen [25], a graph transformation formalism and Xtend [26], another modelware formalism. The lessons learned from developing and applying these measurement techniques, can be used for defining a methodology for developing model transformations with high internal quality.

ACKNOWLEDGMENT

This work has been carried out as part of the FALCON project under the responsibility of the Embedded Systems Institute with Vanderlande Industries as the industrial partner. This project is partially supported by the Netherlands Ministry of Economic Affairs under the Embedded Systems Institute (BSIK03021) program.

REFERENCES

[1] D. C. Schmidt, "Model-driven engineering," *Computer*, vol. 39, no. 2, pp. 25 – 31, Feb. 2006.

[2] A. van Deursen, P. Klint, and J. Visser, "Domain-specific languages: An annotated bibliography," *SIGPLAN Notices*, vol. 35, no. 6, pp. 26 – 36, Jun. 2000.

[3] C. W. Krueger, "Software reuse," *ACM Computing Surveys*, vol. 24, no. 2, pp. 131 – 183, Jun. 1992.

[4] M. F. van Amstel, M. G. J. van den Brand, and L. J. P. Engelen, "An exercise in iterative domain-specific language design," 2010, submitted.

[5] M. F. van Amstel, C. F. J. Lange, and M. G. J. van den Brand, "Using metrics for assessing the quality of ASF+SDF model transformations," in *Proceedings of the Second International Conference on Model Transformation*, ser. Lecture Notes in Computer Science, R. F. Paige, Ed., vol. 5563. Zürich, Switzerland: Springer, Jun. 2009, pp. 239 – 248.

[6] P. Mohagheghi and V. Dehlen, "Developing a quality framework for model-driven engineering," in *Models in Software Engineering, Workshops and Symposia at MoDELS 2007, Reports and Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 5002. Nashville, USA: Springer, Oct. 2007, pp. 275 – 286.

[7] E. Seidewitz, "What models mean," *IEEE Software*, vol. 20, no. 5, pp. 26 – 32, Sep./Oct. 2003.

[8] M. Wimmer and G. Kramler, "Bridging grammarware and modelware," in *MoDELS 2005 Satellite Events*, ser. Lecture Notes in Computer Science, J.-M. Bruel, Ed., vol. 3844. Montego Bay, Jamaica: Springer, Oct. 2005, pp. 159 – 168.

[9] B. W. Boehm, J. R. Brown, H. Kaspar, M. Lipow, G. J. Macleod, and M. J. Merrit, *Characteristics of Software Quality*. North-Holland, 1978.

[10] ISO - International Organization for Standardization, *ISO/IEC 9126-1:2001 - Software engineering – Product quality – Part 1: Quality model*, Geneva, Switzerland, 2001.

[11] M. F. van Amstel, C. F. J. Lange, and M. G. J. van den Brand, "Metrics for analyzing the quality of model transformations," in *Proceedings of the Twelfth ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE'08)*, G. Falcone, Y.-G. Guéhéneuc, C. Lange, Z. Porkoláb, and H. A. Sahraoui, Eds., Paphos, Cyprus, Jul. 2008, pp. 41 – 51.

[12] J. Krogstie, O. I. Lindland, and G. Sindre, "Defining quality aspects for conceptual models," in *Proceedings of the IFIP international working conference on Information system concepts*, ser. IFIP Conference Proceedings, E. D. Falkenberg and W. Hesse, Eds., vol. 26. Marburg, Germany: Chapman & Hall, Mar. 1995, pp. 216 – 231.

[13] C. F. J. Lange, "Assessing and improving the quality of modeling: A series of empirical studies about the UML," Ph.D. dissertation, Eindhoven University of Technology, Eindhoven, The Netherlands, 2007.

[14] M. F. van Amstel, C. F. J. Lange, and M. R. V. Chaudron, "Four automated approaches to analyze the quality of UML sequence diagrams," in *The First IEEE International Workshop on QUality Oriented Reuse of Software (QUORS'07) in proceedings of 31st Annual International Computer Software and Applications Conference*, vol. 2. Beijing, China: IEEE Computer Society, Jul. 2007, pp. 415 – 421.

[15] N. E. Fenton and S. L. Pfleeger, *Software Metrics: A Rigorous & Practical Approach*, 2nd ed. PWS Publishing Co. 1996.

[16] M. Saeki and H. Kaiya, "Measuring model transformation in model driven development," in *Proceedings of the CAiSE'07 Forum at the 19th International Conference on Advanced Information Systems Engineering*, ser. CEUR Workshop Proceedings, J. Eder, S. L. Tomassen, A. L. Opdahl, and G. Sindre, Eds., vol. 247. Trondheim, Norway: CEUR-WS.org, Jun. 2007.

[17] T. Mens and P. van Gorp, "A taxonomy of model transformation," in *Proceedings of the International Workshop on Graph and Model Transformation (GraMoT'05)*, ser. Electronic Notes in Theoretical Computer Science, G. Karsai and G. Taentzer, Eds., vol. 152. Tallinn, Estonia: Elsevier, Sep. 2006, pp. 125 – 142.

[18] D. Varró and A. Pataricza, "Automated formal verification of model tranformations," in *Proceedings of the 2003 Workshop on Critical Systems Development in UML (CSDUML'03)*, ser. Technical Report, J. Jürjens, B. Rumpe, R. France, and E. B. Fernandez, Eds., no. TUM-I0323. San Francisco, USA: Technische Universität München, Oct. 2003, pp. 63 – 78.

[19] M. F. van Amstel, M. G. J. van den Brand, and L. J. P. Engelen, "Using a domain-specific language and fine-grained model transformations to explore the boundaries of model verification," 2010, submitted.

[20] H. Giese, S. Glesner, J. Leitner, W. Schäfer, and R. Wagner, "Towards verified model transformations," in *Proceedings of the 3rd International Workshop on Model Development, Validation and Verification (MoDeV^{2}a'06)*, D. Hearnden, J. G. Süß, B. Baudry, and N. Rapin, Eds. Genova, Italy: Le Commissariat à l'Energie Atomique – CEA, Oct. 2006, pp. 78 – 93.

[21] A. van Deursen, "An overview of ASF+SDF," in *Language Prototyping: An Algebraic Specification Approach*, ser. AMAST Series in Computing, A. van Deursen, J. Heering, and P. Klint, Eds. World Scientific Publishing, 1996, vol. 5, ch. 1, pp. 1 – 29.

[22] M. F. van Amstel and M. G. J. van den Brand, "Quality assessment of ATL model transformations using metrics," 2010, submitted.

[23] A. Vignaga, "Metrics for measuring ATL model transformations," MaTE, Department of Computer Science, Universidad de Chile, Tech. Rep., 2009.

[24] F. Jouault and I. Kurtev, "Transforming models with ATL," in *MoDELS 2005 Satellite Events*, ser. Lecture Notes in Computer Science, J.-M. Bruel, Ed., no. 3844. Montego Bay, Jamaica: Springer, Oct. 2005, pp. 128 – 138.

[25] R. Geiß, G. V. Batz, D. Grund, S. Hack, and A. M. Szalkowski, "GrGen: A fast SPO-based graph rewriting tool," in *Proceedings of the Third International Conference on Graph Transformations (ICGT'06)*, ser. Lecture Notes in Computer Science, A. Corradini, H. Ehrig, U. Montanari, L. Ribeiro, and G. Rozenberg, Eds., vol. 4178. Natal, Brasil: Springer, Sep. 2006, pp. 383 – 397.

[26] M. Völter, "openArchitectureWare: a flexible open source platform for model-driven software development," in *Proceedings of the Eclipse Technology eXchange workshop (eTX) at the ECOOP 2006 Conference*, Nantes, France, Jul. 2006.