Systematic Mapping Study of Model Transformations for Concrete Problems

Edouard Batot, Houari Sahraoui, Eugene Syriani, Paul Molins and Wael Sboui
DIRO, Université de Montréal, Montréal, Canada

Keywords: Model Transformation, Systematic Mapping, Software Engineering.

Abstract: As a contribution to the adoption of the Model-Driven Engineering (MDE) paradigm, the research community has proposed concrete model transformation solutions for the MDE infrastructure and for domain-specific problems. However, as the adoption increases and with the advent of the new initiatives for the creation of repositories, it is legitimate to question whether proposals for concrete transformation problems can be still considered as research contributions or if they respond to a practical/technical work. In this paper, we report on a systematic mapping study that aims at understanding the trends and characteristics of concrete model transformations published in the past decade. Our study shows that the number of papers with, as main contribution, a concrete transformation solution, is not as high as expected. This number increased to reach a peak in 2010 and is decreasing since then. Our results also include a characterization and an analysis of the published proposals following a rigorous classification scheme.

1 INTRODUCTION

Model-Driven Engineering (MDE) has been gaining much popularity in the last decade (Whittle et al., 2014). It is an area of software engineering where problems are expressed at levels of abstraction closer to the problem domain, rather than the domain of code, and software is realized through automated transformation of domain models.

MDE, like any new technology, follows different stages of adoption as described by (Moore, 2002). At each stage, new contributions in research solve part of the problems that represent major adoption obstacles. One of the early obstacles to MDE adoption is the difficulty to write and reuse model transformations for many concrete problems. Indeed, the availability of automated transformations is a prerequisite and a founding principle of MDE.

In MDE, a model transformation is the automatic manipulation of a model following a specification defined at the level of metamodels (Lúcio et al., 2014). It is an operation that accepts a source model as input and produces a target model as output, where each model conforms to its respective metamodel. Typically, a model transformation is defined by a set of declarative rules to be executed (Czarnecki and Helsen, 2006). As an example of transformation, in (Syriani and Ergin, 2012), UML activity diagrams are transformed into Petri nets for simulation and analysis purposes.

With the concern of the adoption of the MDE paradigm, the MDE research community has proposed solutions to concrete model transformation problems. These proposals can be classified into two categories: transformations that improve the MDE infrastructure, such as code generators for programming languages (Funk et al., 2008; Di Marco and Pace, 2013), and transformations for domain-specific problems (Winkler et al., 2012; Yue and Ali, 2012).

However, in recent years, two phenomena changed the landscape of MDE. First, MDE is more and more used in industry. Many studies showed that this new technology is used on strategic projects in many industrial organizations (Mohagheghi et al., 2013).The second phenomenon is the advent of new initiatives for the creation of publicly available repositories of metamodels, models, and transformations (e.g., ReMoDD (France et al., 2007) and ATL Transformations Zoo[1]). In particular, the Transformation Tool Contest (TTC)[2] proposes publicly available solutions to specific problems expressed as model transformations. Both phenomena change the needs in research towards, among others, automated approaches

[1]www.eclipse.org/atl/atlTransformations/
[2]www.transformation-tool-contest.eu/

Batot, E., Sahraoui, H., Syriani, E., Molins, P. and Sboui, W.
Systematic Mapping Study of Model Transformations for Concrete Problems.
DOI: 10.5220/0005657301760183
In *Proceedings of the 4th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2016)*, pages 176-183
ISBN: 978-989-758-168-7
Copyright © 2016 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

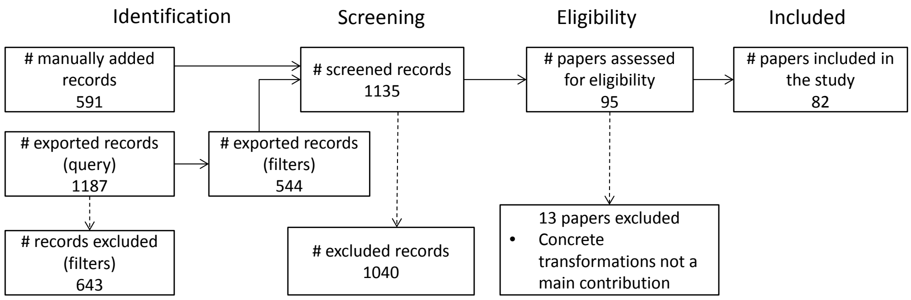

Figure 1: Flow of information during the selection process.

to derive MDE artifacts such as metamodel well-formedness rules (Faunes et al., 2013) or model transformations (Baki et al., 2014). In particular, it is legitimate to question whether proposals for specific artifacts, such as metamodels and transformations of concrete problems can still be considered as research contributions, or if they respond to a practical need.

To help answering such a question, we conducted a systematic mapping study (Petersen et al., 2008), covering the last decade (2005-2014), that aims at understanding the trends and characteristics of model transformations proposed for concrete problems and published in research forums. We opted for a systematic empirical process to minimize bias and maximize reproducibility of the study. In addition to study the evolution of the amount of published material during the considered period, we are also interested in various characteristics of these transformations such as their nature, the used languages, the involved metamodels, and the relation to industry.

Our results show that the number of papers with as main contribution a concrete transformation increased since 2005 to reach a peak in 2010, but has been decreasing since then. Among other results, the majority of the proposals deals with general modeling languages as compared to proposals for domain-specific problems. An interesting number of the proposals are industry-oriented.

The remainder of the paper is organized as follows. In Section 2, we describe the paper selection and analysis procedure. We report the results of the systematic mapping using a predefined classification scheme in Section 3. We briefly outline related work in Section 4 and finally conclude in Section 5.

2 PROCEDURE AND SELECTION

The following subsections describe the main activities we performed (from (Petersen et al., 2008)) in order to discover the trends in model transformation.

2.1 Research Objectives

As motivated in Section 1, the modeling community values to know whether producing model transformations that solve concrete problems is useful and still considered a research contribution. We, therefore, formulate our research objectives with the following two research questions:

- **RQ1:** What are the trends in concrete model transformations?
- **RQ2:** What are the characteristics of these transformations?

2.2 Paper Selection

With these two objectives in mind, we determine the scope of the search to be contributions in the literature, published between 2005 and 2014, that present a model transformation for a concrete problem, e.g., transformation from UML activity diagrams to Petri nets (Syriani and Ergin, 2012).

To retrieve the paper of interest, we queried the Scopus database[3]. Moreover, we manually added papers to the corpus from the following specialized forums, which are likely to have papers of interest to this study: ICMT, ECMFA, MODELS, and SOSYM.

As shown in Fig. 1, after the querying and filtering, we obtained 544 papers from Scopus. For the manual addition, we considered all the 591 long papers published in the four MDE forums between 2005 and 2014. After combining the two sources, a corpus of 1 135 papers was considered for the screening phase.

Screening is the most crucial phase in the systematic mapping process (Petersen et al., 2008). In order to avoid the exclusion of papers that should be part of the final corpus, we followed a strict screening procedure. With four reviewers at our disposal (co-authors of this paper), each article is screened by two reviewers independently.

[3]Scopus archives over 55 million records from over 5,000 publishers www.elsevier.com/online-tools/scopus.

Table 1: Classification scheme.

Category	Description
Transformation kind	Does the transformation operate on the *structure*, mainly the syntax (e.g., migration), or on the *behavior*, mainly the semantics (e.g., simulation), of the models involved?
Metamodel kind	Do the input and output metamodels describe a *general-purpose language* (e.g., UML, source code) or *domain-specific language*?
Model kind	Do the transformed models come from *industrial* data (e.g., private), from *publicly* available data (e.g., open-source), or from made-up *toy examples*?
Intent	Under which *intent category* does the transformation fall, as defined in (Lúcio et al., 2014)?
Transformation language	Is the model transformation expressed using a *dedicated language* for transformations (e.g., ATL), a *programming language* (e.g., Java), or in *another* way (e.g., formally, without implementation)?
Validation	Is the transformation verified and validated *formally* (e.g., proving properties), *empirically* (e.g., case study, validated on multiple models), or is there *no validation* (e.g., informal argumentation)?
Scope	Is the transformation *exogenous* (defined on different metamodels), *outplace* (operates on different models, but defined on the same metamodel), or *inplace* (operates on the same models) as defined in (Mens and Van Gorp, 2006)?
Orientation	Does the transformation involve an author from *industry* or only *academic* authors are concerned?

A paper is included if its title or abstract explicitly mentions a model transformation in its peculiar context. It is excluded if either the transformation is not the main topic of the paper or the abstract indicates the paper proposes a generalization of a transformation technique rather than an actual concrete transformation. Among the 1 135 screened papers, 1 040 were excluded, 83 were included directly (both reviewers agreed), and 12 were included after conflict resolution, for a total of 95 papers. After screening, the full text of the 95 papers were read in order to detect situations where the abstract suggested to be in favor of including the paper, whereas the content of the paper suggested otherwise. This step excluded 13 additional papers from the study. Indeed, although the reviewers concluded from the abstract that the papers satisfy the four criteria, the in-depth examination showed that the proposed transformations did not represent the main contribution of the papers. The final number of papers considered for the study is then 82 [4].

[4] The complete list is available online geodes.iro.umontreal.ca/modeltransformSMS/

When we determined the objectives of this study, we already had in mind some of the criteria with which we were going to evaluate the papers. While reading all the abstracts during screening, we refined these criteria until we obtained the classification scheme in Table 1. It will be used to classify all retained papers along different categories that are of interest in order to answer our research questions.

3 ANALYSIS

In this section, we analyze the retained papers according to the classification scheme in order to answer the research questions stated in Section 2.

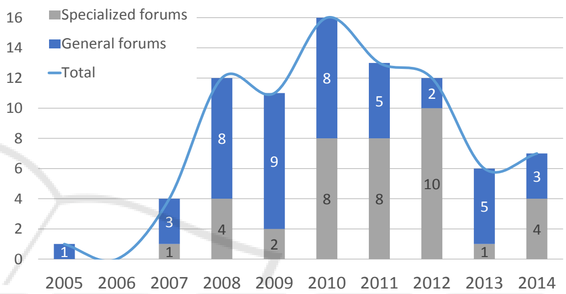

Figure 2: Evolution of concrete model transformations.

3.1 Evolution of Concrete Model Transformations

The number of papers with a concrete model transformation as a main concern increased since 2005 to reach a peak in 2010 and has been decreasing since then, as depicted in Fig. 2. We distinguish between papers published in general software engineering forums and forums specialized in MDE. We note that both general and specialized forums follow a similar trend. However, we notice a shift towards specialized forums around 2010. This is a recurring pattern that often occurs when a new research community gains popularity. The drop in 2013 may indicate that producing concrete model transformations is becoming a development task rather than a research endeavor.

3.2 Characteristics of Concrete Model Transformations

Now that we shed light on a general trend in the model transformation community, we investigate the characteristics of concrete model transformations in order to answer RQ2, using the classification scheme presented in Table 1.

SCIENCE AND TECHNOLOGY PUBLICATIONS

3.2.1 Intent

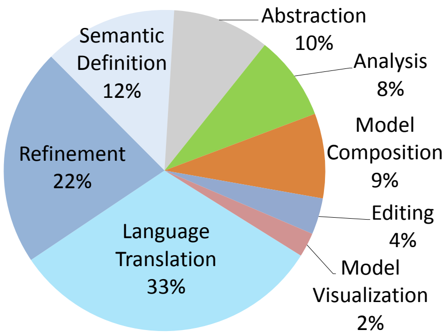

Figure 3: Distribution of intent category.

Three classes of intents account, each, for more than 10% of the classified papers, as depicted in Fig. 3. The most popular transformation intent class is language translation, which encompasses a third of the analyzed transformations. Such transformations establish a bridge between a source and a target modeling language to achieve tasks that are difficult (or impossible) to perform on the source language (Yang et al., 2014) or to make use of a specific tool (Winkler et al., 2012). The second most frequent class of intent is refinement with 22%. Code synthesis is the dominant transformation as in (Di Marco and Pace, 2013), where wireless sensor network code is generated from a UML profile. Nevertheless, other kinds of refinements are also present, such as refining a requirement model into a platform-independent model of a multi-agent system (Harbouche et al., 2013). Semantic definition is another predominant intent class with 12%. On the one hand, there are simulation transformations that encode the operational semantics of a language. On the other hand, there are translational semantics transformations, whose purpose is to translate one metamodel into another in order to define its semantics, as in (Wagelaar et al., 2012). The lesser popularity of the remaining intents (10% or less) is due to the fact that approaches to perform, for example, analysis and visualization, already exist in non-MDE technologies. Therefore, the community has not been focusing on explicitly modeling these tasks by means of model transformation.

3.2.2 Transformation Kind

As depicted in Fig. 4 (right), a striking majority of the papers deal with structural transformations. Behavioral transformations are scarce, mostly designed to analyze software evolution and perform simulations. This accords with the distribution of intents. (Gîrba et al., 2005) were precursors and showed how "one can manipulate time information just like structural information". In fact, analysis or simulation of the behavior of systems through transformations are often preceded by a structural translation of models, in order to reuse existing tools: e.g., transforming a domain-specific language or UML (Anastasakis et al., 2007) into Alloy for analysis purposes.

3.2.3 Scope

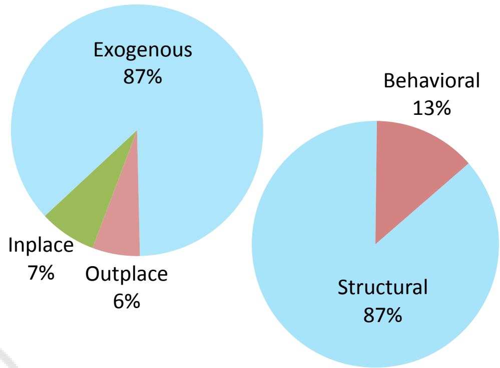

Figure 4: Distribution of scope and transformation kind category.

The lack of experience of the community in terms of endogenous transformation is put to light in Fig. 4 (left). Only 13% of the transformations were endogenous, inplace (e.g., simulating the token behavior of Petri nets (Syriani and Ergin, 2012)) or outplace (e.g., performing row and column manipulations on spreadsheets (Cunha et al., 2012)). In fact, this corroborates with the intent distribution, since the three most popular intents are typically implemented by means of exogenous transformations. Many of the proposed endogenous transformations are implemented by mean of graph transformations (see, for example, (Amelunxen and Schürr, 2008)). In addition to refactoring, simulation and analysis, metamodel-model co-evolution and metamodel-transformation co-evolution (e.g., (Garcés et al., 2014)) are naturally favored application domains, as they require modifying existing models. Other papers deal with model synchronization (under the model composition intent class), which is a special case as one can see it as an exogenous transformation. Indeed, the synchronization is performed between two models, generally belonging to two different metamodels. However, synchronization can also be seen as the evolution of a given model to handle new constraints resulting from the modification of another model. This is the case, for example, in (Xiong et al., 2013) where the authors propose an algorithm for synchronizing concurrent models.

3.2.4 Metamodel Kind

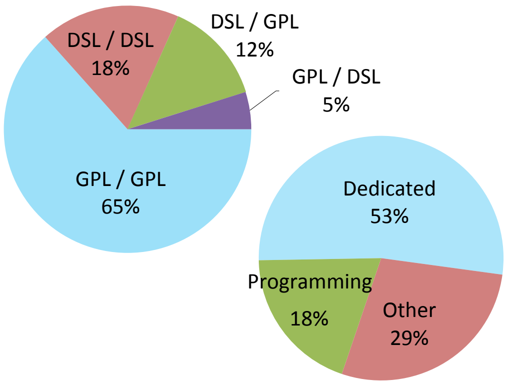

Figure 5: Distribution of metamodel-kind and tansformation-language category.

Fig. 5 (left) shows the distribution of combinations between general-purpose and domain-specific source and target metamodels of a transformation. Overall, two thirds of the approaches favor reusing existing tooling to simulate their systems, thus transforming between general-purpose languages (e.g., (Denil et al., 2014)). In particular, (a subset of) UML is the most frequently used metamodel. General-purpose languages, such as Ecore (UML), XML (Eichberg et al., 2010), and Petri nets (Syriani and Ergin, 2012), are mostly used as target, again to favor reuse of existing technologies. Domain-specific metamodels are typically used in transformations to translate from one language to another (Acher et al., 2009). Most of the papers involving source and/or target domain-specific metamodels deal with business concerns, like business models in (Siqueira and Silva, 2014). Other domain-specific to general-purpose language transformations range from particular aspects of application development such as user-interface generation (Pastor, 2007) to very specific domains like configuration of video surveillance systems (Acher et al., 2009). Finally, a representative case of DSL-to-DSL transformations is the one in (Selim et al., 2012). In this work, the transformation aims at migrating models expressed using a General Motors domain-specific language to AUTOSAR.

3.2.5 Transformation Language

As Fig. 5 (right) illustrates, more than half of the transformations are implemented in a language dedicated to model transformations. Half of those are implemented in languages considered de facto standards (ATL (Wagelaar et al., 2012) and QVT (Siqueira and Silva, 2014)), and the other half with less popular languages (Pastor, 2007).Only 18% of the papers still used programming languages for their transformations. These are mainly written in Java for the Eclipse platform (Buchmann et al., 2011), but also Prolog (Eichberg et al., 2010) and XSLT. The remaining 29% of the papers only described the transformation, either without implementation or by implementing it in a one-time use tool (Yue and Ali, 2012). These observations reveal that model transformation is being recognized as a paradigm in itself.

3.2.6 Model Kind and Orientation

On the one hand, in the vast majority of cases, we found that scalable examples are not a priority: over two-thirds of the papers illustrate their results with toy examples, often made up for the particular work. Although these toy examples cannot provide compelling evidence about the quality of the proposed transformation, they have the advantage of clearly illustrate the transformation and then favor its adoption by the potential users (Whittle et al., 2014). On the other hand, industrial data shows how strong-built and scalable the technology is. Among the few cases where large sets of industrial data are involved, it is worth mentioning the work by (Hermann et al., 2014), in which data provided by satellite Astra is used to validate the proposed transformation. Sometimes, although industrial data is used, its size is too small to produce compelling evidence of the quality/usefulness of the proposed transformation (Selim et al., 2012). Larger data models, from industry or that are publicly available, have been slowly gaining momentum in the past few years (Yue and Ali, 2012), as indicated in Fig. 6. This is certainly influenced by the fact that an industrial stakeholder was involved for 20% of the papers, which has happened predominantly in the 2010-2012 period. It is interesting to note that papers with industrial authors follow the same trend as those with academic authors only: most publications in 2010, mainly exogenous transformations, but with more industrial models.

3.2.7 Validation

Concrete model transformations are being validated empirically more often with time (see Fig. 6). This observation corroborates with a previous study in 2011 (Carver et al., 2011). Indeed, since 2011, MODELS has increased the number of pages for submissions in order to give more space for a discussion about validation. Half of the papers have validated their work empirically as in (Yue and Ali, 2012), with a peak in 2012. Nevertheless, most validations were performed on small examples (Siqueira and Silva, 2014) which reduces the scalability of the validation

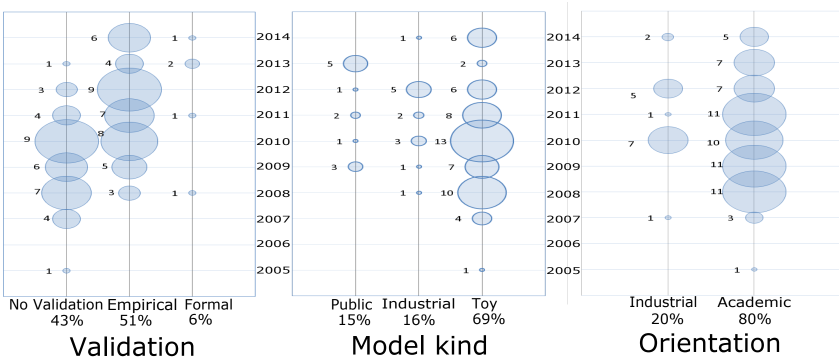

Figure 6: Evolution of validation, model kind and orientation, and their influences.

to its peculiar context. Furthermore, Fig. 6 shows that a drop in studies using toy models correlates with the increase of empirical validation. With respect to the forms of validation, case studies are the most used with, as mentioned earlier, small illustrating examples. Still, some contributions, such as (Hermann et al., 2014) and (Yue and Ali, 2012), propose strongly built validations, addressing both performances and accuracy aspects. Validation is also made by simulation as in (Denil et al., 2014) but they remain scarce. Finally, in the very few papers that use theoretical validation, this is implicit as the proposed transformation is itself described formally.

3.3 Discussion

With respect to RQ1, after tracing correspondences between the statistical results from this classification, there are two possible explanations for the trend observed in Fig. 2: either the interest in model transformation is decreasing or the model transformation community has reached an appropriate level of maturity and adoption since 2013. However, the former should be discarded because of the continued popularity of transformation-exclusive venues: ICMT and TTC. Indeed, its main artifacts—concrete model transformations—have been pulling out from the research literature since 2009 and are becoming considered as development tasks. This observation is corroborated by the recent survey in (Whittle et al., 2014), in which, a large number of surveyed actors use MDE with concrete domain-specific artifacts. By no means should one conclude that all transformation problems have been solved. Instead, this indicates that significant scientific contributions are now being exploited to solve practical problems.

Answering RQ2 suggests that the model transformation community has favored exploring exogenous transformations, that are structural in order to translate, refine/synthesize code, or to give precise meaning to models. This is mainly a consequence of wanting to reuse existing non-modeled software, as opposed to modeling the solution in a model transformation for behavioral transformations, such as analysis and simulation. As a result, a plethora of tools have emerged for implementing model transformation, without one clear outlier. Although languages dedicated to model transformations are preponderant, programming languages are still used. From another perspective, transformations are seizing to be applied on toy models and, since 2010, are becoming more applied in larger case studies. Finally, research in model transformations is heading for a more stable and grounded validation. The number of studies validated empirically has been gradually increasing in the past decade. This confirms the level of maturity reached by model transformation, as stated for RQ1.

4 RELATED WORK

Several works attempted to classify model transformations. However, to the best of our knowledge, this paper is the first to propose a systematic mapping study for model transformations.

(Mens and Van Gorp, 2006) proposed a taxonomy of model transformation based on how models are manipulated and their execution strategies. Related to this contribution, (Czarnecki and Helsen, 2006) classified the features offered by languages to express model transformations. For a separate but related matter, (Lúcio et al., 2014) have cataloged the different use cases and intents where a model transformation can be used.

Nevertheless, there have been several empirical studies about various aspects of MDE. Concerning the MDE adoption, (Hutchinson et al., 2011; Mohagheghi et al., 2013) performed user studies in the form of interviews and surveys among developers to investigate how MDE technologies are applied in industry. Similarly and with respect to the development process, (Martínez et al., 2014) compared the performance of maintenance tasks when using an MDE approach against a code-centric approach.

From a product perspective, (Vanderose and Habra, 2008) propose an approach to empirically evaluate quality factors of developed artifacts in MDE. In the same vein, (Fernandez et al., 2013) performed a series of user studies to assess the usability of web applications developed in an MDE process.

The closest study to the one in this paper is (Carver et al., 2011): a systematic survey focused on MODELS in order to understand the frequency with which empirical evaluations have been reported.

5 CONCLUSION

In this paper, we report on a systematic mapping study to understand the trends and characteristics of model transformations for concrete problems. Our study uses a major online database, Scopus, along with the published articles in the four major MDE forums. This study, which covers the period 2005-2014, was conducted following the systematic mapping process. First, we collected all publications found by querying the database and by gathering the papers of the specialized forums. Then, we screened them using their abstract to ensure that they were eligible for our analysis. Finally, we analyzed every included article to classify the proposed transformation according to a predefined scheme.

In addition to the findings discussed throughout this paper, our study is a contribution to a global assessment of the state of research and adoption of MDE. Indeed, as for any new technology, it is our duty as a research community to reflect globally on what is relevant for research and what should be treated as technical problems. In this context, we plan to periodically repeat this study to have an up-to-date portray of the situation. We also plan to perform a similar study on additional MDE artifacts, in particular, metamodels. Finally, the classification scheme will be evolved to take into consideration new research results.

REFERENCES

Acher, M., Lahire, P., Moisan, S., and Rigault, J.-P. (2009). Tackling high variability in video surveillance systems through a model transformation approach. In *Models in Soft. Eng.*

Amelunxen, C. and Schürr, A. (2008). Formalising model transformation rules for UML/MOF 2. *IET Software*, 2(3).

Anastasakis, K., Bordbar, B., Georg, G., and Ray, I. (2007). UML2Alloy: A Challenging Model Transformation. In *Proc. of the Int. Conf. on Model-Driven Engineering Languages and Systems*.

Baki, I., Sahraoui, H., Cobbaert, Q., Masson, P., and Faunes, M. (2014). Learning Implicit and Explicit Control in Model Transformations by Example. In *Proc. of the Int. Conf. on Model-Driven Engineering Languages and Systems*.

Buchmann, T., Westfechtel, B., and Winetzhammer, S. (2011). ModGraph-A Transformation Engine for EMF Model Transformations. In *Int. Conf. on Software and Data Technologies*.

Carver, J. C., Syriani, E., and Gray, J. (2011). Assessing the Frequency of Empirical Evaluation in Software Modeling Research. In *EESSMod*, volume 785.

Cunha, J., Fernandes, J. P., Mendes, J., Pacheco, H., and Saraiva, J. (2012). Bidirectional Transformation of Model-Driven Spreadsheets. In *Theory and Practice of Model Transformations*.

Czarnecki, K. and Helsen, S. (2006). Feature-Based Survey of Model Transformation Approaches. *IBM Systems J.*, 45(3).

Denil, J., Mosterman, P. J., and Vangheluwe, H. (2014). Rule-based model transformation for, and in simulink. In *Symposium on Theory of Modeling and Simulation*.

Di Marco, A. and Pace, S. (2013). Model-driven approach to Agilla Agent generation. In *Wireless Communications and Mobile Computing Conf.*

Eichberg, M., Monperrus, M., Kloppenburg, S., and Mezini, M. (2010). Model-Driven Engineering of Machine Executable Code. In *Proc. of the Eur. Conf. on Modelling Foundations and Applications*.

Faunes, M., Cadavid, J., Baudry, B., Sahraoui, H., and Combemale, B. (2013). Automatically Searching for Metamodel Well-Formedness Rules in Examples and Counter-Examples. In *Proc. of the Int. Conf. on Model-Driven Engineering Languages and Systems*.

Fernandez, A., Abrahão, S., and Insfran, E. (2013). Empirical validation of a usability inspection method for model-driven Web devt. *Int. J. on Soft. and Systems Modeling*, 86.

France, R., Bieman, J., and Cheng, B. (2007). Repository for Model Driven Devt (ReMoDD). In *Models in Software Engineering*, volume 4364.

Funk, M., Nyßen, A., and Lichter, H. (2008). From UML to ANSI-C - An Eclipse-Based Code Generation Framework. In *Int. Conf. on Software and Data Technologies*.

Garcés, K., Vara, J. M., Jouault, F., and Marcos, E. (2014). Adapting transformations to metamodel changes via external transformation composition. *Int. J. on Soft. and Systems Modeling*.

Gîrba, T., Favre, J.-M., and Ducasse, S. (2005). Using Meta-Model Transformation to Model Software Evolution. *Electronic Notes in Theoretical Computer Science*, 137(3).

Harbouche, A., Erradi, M., and Mokhtari, A. (2013). Deriving Multi-Agent System Behavior. *Int. J. of Soft. Eng. and Its Applications*, 7(4).

Hermann, F., Gottmann, S., Nachtigall, N., Ehrig, H., Braatz, B., Morelli, G., Pierre, A., Engel, T., and Ermel, C. (2014). Triple Graph Grammars in the Large for Translating Satellite Procedures. In *Int. Conf. on Theory and Practice of Model Transformations*.

Hutchinson, J., Whittle, J., Rouncefield, M., and Kristoffersen, S. (2011). Empirical assessment of MDE in industry. In *Proc. of the Int. Conf. on Software Engineering*.

Lúcio, L., Amrani, M., Dingel, J., Lambers, L., Salay, R., Selim, G., Syriani, E., and Wimmer, M. (2014). Model transformation intents and their properties. *Int. J. on Soft. and Systems Modeling*.

Martínez, Y., Cachero, C., and Meliá, S. (2014). Empirical study on the maintainability of Web applications: Model-driven Engineering vs Code-centric. *Empirical Soft. Eng.*, 19(6).

Mens, T. and Van Gorp, P. (2006). A Taxonomy of Model Transformation. In *Int. Work. on Graph and Model Transformation*, volume 152.

Mohagheghi, P., Gilani, W., Stefanescu, A., and Fernandez, M. (2013). An empirical study of the state of the practice and acceptance of mde in four industrial cases. *Empirical Soft. Eng.*

Moore, G. (2002). *Crossing the Chasm: Marketing and Selling Disruptive Products to Mainstream Customers*.

Pastor, Ó. (2007). Generating User Interfaces from Conceptual Models: A Model-Transformation Based Approach. In *Computer-Aided Design of User Interfaces V*.

Petersen, K., Feldt, R., Mujtaba, S., and Mattsson, M. (2008). Systematic Mapping Studies in Software Engineering. In *Int. Conf. on Evaluation and Assessment in Soft. Eng.*, EASE'08.

Selim, G. M. K., Wang, S., C., J. R., and Dingel, J. (2012). Model Transformations for Migrating Legacy Models: An Industrial Case Study. In *Proc. of the Eur. Conf. on Modelling Foundations and Applications*.

Siqueira, F. L. and Silva, P. S. M. (2014). Transforming an enterprise model into a use case model in business process systems. *Int. J. on Soft. and Systems Modeling*, 96.

Syriani, E. and Ergin, H. (2012). Operational Semantics of UML Activity Diagram: An Application in Project Management. In *RE 2012 Workshops*.

Vanderose, B. and Habra, N. (2008). Towards a generic framework for empirical studies of Model-Driven Engineering. In *Work. on Empirical Studies of MDE*.

Wagelaar, D., Iovino, L., Di Ruscio, D., and Pierantonio, A. (2012). Translational Semantics of a Co-evolution Specific Language with the EMF Transformation Virtual Machine. In *Theory and Practice of Model Transformations*.

Whittle, J., Hutchinson, J., and Rouncefield, M. (2014). The state of practice in model-driven engineering. *Software, IEEE*, 31.

Winkler, U., Fritzsche, M., Gilani, W., and Marshall, A. (2012). BOB the Builder: A Fast and Friendly Model-to-PetriNet Transformer. In *Proc. of the Eur. Conf. on Modelling Foundations and Applications*.

Xiong, Y., Song, H., Hu, Z., and Takeichi, M. (2013). Synchronizing concurrent model updates based on bidirectional transformation. *Int. J. on Soft. and Systems Modeling*, 12(1).

Yang, Z., Hu, K., Ma, D., Bodeveix, J.-P., Pi, L., and Talpin, J.-P. (2014). From {AADL} to Timed Abstract State Machines: A verified model transformation. *Int. J. on Soft. and Systems Modeling*, 93.

Yue, T. and Ali, S. (2012). Bridging the Gap between Requirements and Aspect State Machines to Support Non-functional Testing: Industrial Case Studies. In *Proc. of the Eur. Conf. on Modelling Foundations and Applications*, volume 7349.