Systematic Mapping Study of Model Transformations for Concrete Problems

Edouard Batot, Houari Sahraoui, Eugene Syriani, Paul Molins and Wael Sboui
DIRO, Université de Montréal, Canada
{batotedo,sahraouh,syriani,molinspa,sbouiwae}@iro.umontreal.ca

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 [1]. 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 [2]. 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 [3]. 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 [4]. As an example of transformation, in [5], 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: (1) transformations that improve the MDE infrastructure, such as code generators for programming languages [6, 7], and (2) transformations for domain-specific problems [8, 9].

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 [10].The second phenomenon is the advent of new initiatives for the creation of publicly available repositories of metamodels, models, and transformations (e.g., ReMoDD [11] 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 to derive MDE artifacts such as metamodel well-formedness rules [12] or model transformations [13]. 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/technical need.

To help answering such a question, we conducted a systematic mapping study [14], 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. **EB** ►*modif*◄ 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. In Section 3, we present specifically the results of the paper selection phase. We report the results of the systematic mapping using a predefined classification scheme in Section 4. The validity of the study is discussed in Section 5. We briefly outline related work in

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

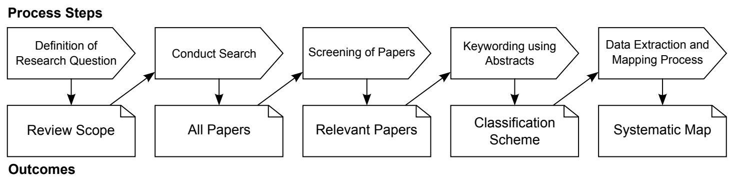


Figure 1: The systematic mapping process.

Section 6 and finally conclude in Section 7.

2. PROCEDURE

In order to discover the trends in concrete model transformations, we conducted a systematic mapping study [14]. The process defined by Petersen et al. [14] is outlined in Fig. 1. The following describes the main activities we performed.

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?

Selection of Source

With these two objectives in mind, we determine the scope of the search to be contributions in the literature that present a model transformation for a concrete problem. For example, the contribution of [5] is to present the transformation from UML activity diagrams to Petri nets. Formulating a query that retrieves precisely research contributions for such *concrete model transformations* presents many difficulties, given that this term will most likely not appear in the title, abstract, or content of indexed publications. Therefore, we opted for a less specific query even if it meant to review a larger corpus. We used the following search string to query databases for papers describing concrete model transformations:

```
(``Model driven engineering'' OR
``Model driven software'' OR
``Model based software'')
AND (``transform*'' OR ``mapping'')
```

Several online databases archive valuable MDE literature. For this study, we used Scopus[3] which has a better coverage than specific publishers (such as IEEE Xplore and Springer Link) and can export all results of a search automatically to a workable format (unlike e.g., Google Scholar). Moreover, we manually added papers to the corpus from the following specialized forums, which are likely to have papers of interest to this study: the International Conference on Model Transformation (ICMT) ; the European Conference on Modelling Foundations and Applications (ECMFA) ; the International Conference on Model Driven Engineering Languages and Systems (MODELS) ; and, the Journal on Software and Systems Modeling (SOSYM)

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

2.1 Screening Procedure

Screening is the most crucial phase in the systematic mapping process [14]. We followed a two-stage screening procedure: automatic filtering, then title and abstract screening. 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. When both reviewers of a paper disagree upon the inclusion or exclusion of the paper, a physical discussion is required. If the conflict is still unresolved, a third senior reviewer is involved in the discussion until a consensus is reached. To determine a fair exclusion process, a senior reviewer reviews a sample of no less than 20% of the excluded papers at the end of the screening phase, to make sure that no potential paper is missed.

Inclusion/Exclusion Criteria

A paper is included if its title or abstract explicitly mentions a model transformation in its peculiar context.

Results from the search were first filtered to automatically discard records that were outside the scope of this study: papers not in the software engineering domain (since the terms “model” and “transformation” appear in many other fields), with less than five pages of length (e.g., proceedings preface), not a scientific paper (e.g., white papers), not written in English, or not published between 2005 and 2014 (the covered period). Then, papers were excluded through manual inspection based on the following criterion: 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.

2.2 Classification scheme

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. The classification scheme will help analyze overall results and give an overview of the trends and characteristics of concrete model transformations.

We partitioned each category of the classification scheme into distinct classes. We assigned each concrete model transformation to the corresponding class by answering the questions in the second column of Table 1. Particular precisions are added to some categories below.

Intents As recognized by Lucio et al. [3], determining the appropriate intent of a transformation is difficult and subjective. Therefore, we chose to classify concrete transformations according to the nine intent categories they propose.

Orientation We consider that an industrial actor is involved actively if at least one author has an industrial affiliation.

Transformation language A dedicated language is one whose sole purpose is to perform model transformations, such as ATL [16]

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 [3]?
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 [15]?
Orientation	Does the transformation involve an author from *industry* or only *academic* authors are concerned?

or QVT [17]. Otherwise, the transformation is either implemented in a general-purpose programming language e.g., Java, or in an adhoc prototype designed for a specific transformation, as in [18]. A transformation may also be only described formally or not implemented at all. The latter three are grouped into the *other* class.

3. SELECTION

Fig. 2 summarizes the flow of information through the selection process of this study. As mentioned in Section 2, we relied on two sources to search for the papers to include in our study: Scopus and specialized forums. For both sources, we considered the period from 2005 to 2014. In what follows, we present the different steps of the selection process we followed.

3.1 Paper Identification

The identification step with Scopus was done in two phases: querying and filtering. We obtained 1 187 candidate papers that satisfy the search query. We then applied the set of filters described in Section 2.1 to reduce this number. The filters eliminated 643 papers, which left 544 papers obtained from Scopus.

For the manual addition, we considered all long papers published in the four MDE forums. We chose 2005 to be our starting year because that is when the first edition of MODELS started, which is considered the top venue for MDE research. Prior to that, the UML CONFERENCE SERIES did not focus on MDE, but on UML and there was still no common consensus on terminology until then. Excluding the UML CONFERENCE SERIES, SOSYM is the only forum that has publications prior to 2005. Thus, to be consistent with the database identification, we excluded those issues to end with a total of 591 papers. After combining the two sources, a corpus of 1 135 papers was considered for the screening phase.

3.2 Screening

As mentioned in Section 2.1, each paper was screened by two reviewers to decide for its inclusion depending on the four following exclusion criteria:

1. The paper is not in software engineering field.
2. It is not a full conference or journal paper.
3. The paper is not about model transformation.
4. The main contribution of the paper is not transformation for a concrete problem.

To support the reviewers in their screening task and automate the process of the screening results, we used a home-made tool [19]. This interface allows reviewers to iterate over papers, while presenting the authors, title and abstract for each. After reading this information, the reviewer can either accept the paper by pressing the OK button, or reject it by selecting an exclusion reason. As the same reviewer has to go through hundreds of papers, it is possible to save a session. When all reviewers complete their screening, the tool compares their ratings and classifies every paper as “included” (both reviewers accepted), “excluded” (both reviewers rejected), or “conflict” (one accepted and the other rejected).

In our study, among the 1 135 screened papers, 1 040 were excluded, 83 were included, and 105 received conflicting ratings (about 9% only). In almost all cases of conflict, one of the reviewers excluded the paper because the proposed transformation is not the main contribution of the paper (criterion 4), whereas, the other asserted the opposite. We were expecting such conflicts since, in many cases, it is difficult to decide, only from the title and abstract, whether a transformation for a concrete problem is proposed and whether this is the main contribution of the paper. These conflicts were resolved in physical meetings and only 12 out of the 105 papers were finally included for a total of 95 papers. For the excluded paper, the most frequent exclusion criterion was 4, which accounted for half of the exclusions. The second most important criterion was 3 (40%). That is because many papers cover MDE aspects other than transformations. Finally, around 10% of the papers were excluded because of criteria 1 and 2. In fact, these latter criteria were in place to catch papers that escaped the database filtering.

During the screening, we had to deal with a special group of papers that propose a generic transformation framework for a category of problems. Usually, these contributions include a domain-specific language (DSL) for expressing the transformations, while encoding generic transformation modules to support the expressed transformations. We decided to include the papers when the emphasis is put on these generic modules rather than the DSL itself. Thus, an example of papers that we included is [20] because most of the proposal is dedicated to the generic transformation of web 2.0 databases into data models. Another example of retained papers is [8] as it focuses on a domain-specific model to Petri nets transformation templates. In contrast, some papers of the same group were not included because either the category of target problems is too large to have significant encoded transformation semantics (e.g., [21] for tree based data-structures transformations) or the emphasis is put on the DSL as in [22] for modeling model slicers.

3.3 Eligibility

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

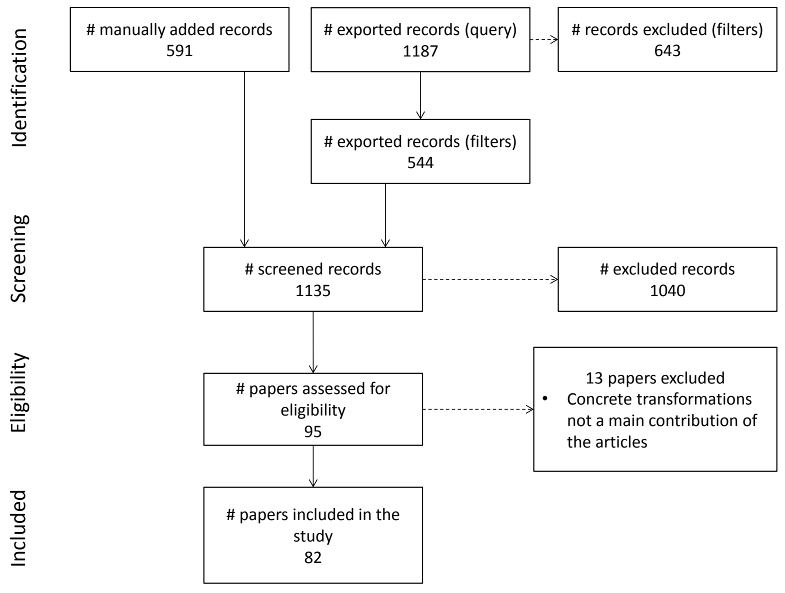

Identification
Screening
Eligibility
Included
# manually added records 591
# exported records (query) 1187
# records excluded (filters) 643
# exported records (filters) 544
# screened records 1135
# excluded records 1040
# papers assessed for eligibility 95
13 papers excluded
• Concrete transformations not a main contribution of the articles
# papers included in the study 82

Figure 2: Flow of information during the selection process

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 82footnote The complete list is available online `geodes.iro.umontreal.ca/modeltransformSMS/`.

4. 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.

4.1 Evolution of concrete model transformations

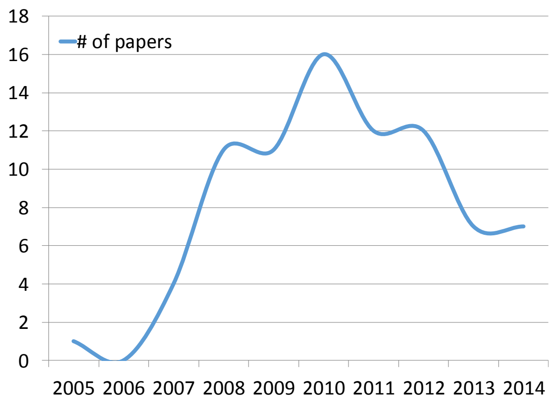
The number of papers with a concrete model transformation as a main concern increased since 2005 to reach a peak in 2010 and has been decreasing since then, as depicted in Fig. 3a. In Fig. 3b, we distinguish between papers published in general software engineering forums and forums specialized in MDE. 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.

4.2 Characteristics of concrete model transformations

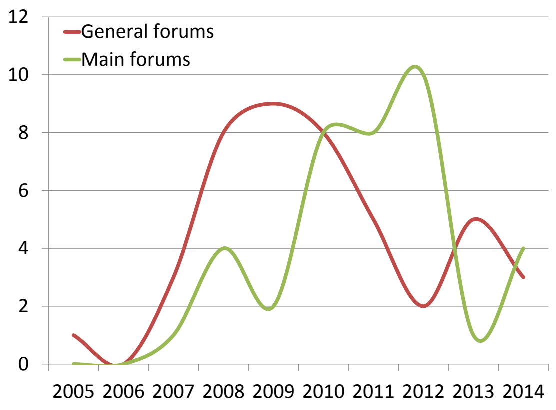
Now that we shed light on a general trend in the model transformation community, we investigate the characteristics of concrete model transformations in order to answer RQ2, using the classification scheme presented in Table 1.

4.2.1 Intent

Three classes of intents account, each, for more than 10% of the classified papers, as depicted in Fig. 4. 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



(a) Number of concrete model transformations published between 2005 and 2014



(b) Number of concrete model transformations per source

Figure 3: Evolution of concrete model transformations

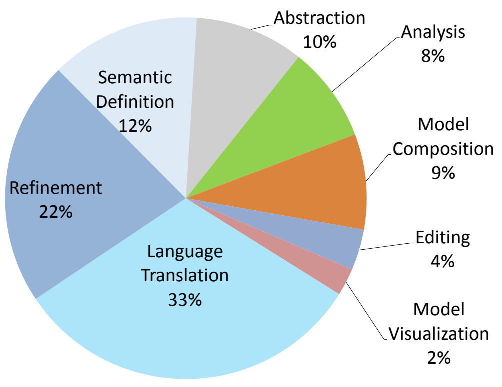


Figure 4: Distribution of intent category

to achieve tasks that are difficult (or impossible) to perform on the source language [23] or to make use of a specific tool [8]. The second most frequent class of intent is refinement with 22%. Code synthesis is the dominant transformation as in [7], 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 [24]. 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, such as the one in [25]. 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 [26]. 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.

4.2.2 *Transformation kind*

As depicted in Fig. 5, 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. Girba et al. [27] 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 [28] or UML [29] into Alloy for analysis purposes.

4.2.3 *Scope*

The lack of experience of the community in terms of endogenous transformation is put to light in Fig. 5. Only 13% of the transformations were endogenous, inplace (e.g., simulating the token behavior of Petri nets [5]) or outplace (e.g., performing row and column manipulations on spreadsheets [30]). 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, [31] and [32]). In addition to refactoring, simulation and analysis, metamodel-model co-evolution (e.g., [33]) and metamodel-transformation co-evolution (e.g., [34]) are naturally favored application domains, as they require modifying existing models. Other papers deal with model synchronization (under the model composition intent class), which

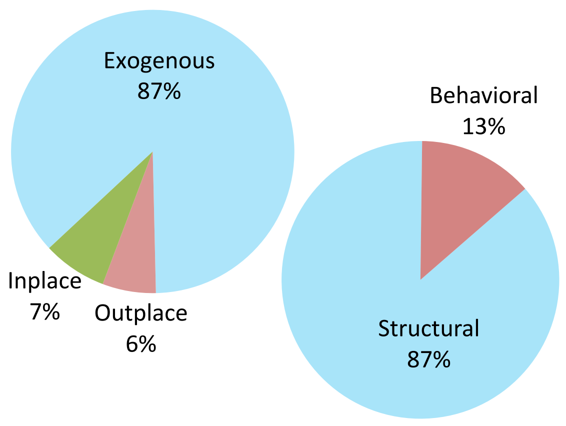


Figure 5: Distribution of scope and transformation kind category

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 [35] where the authors propose an algorithm for synchronizing concurrent models.

4.2.4 *Metamodel kind*

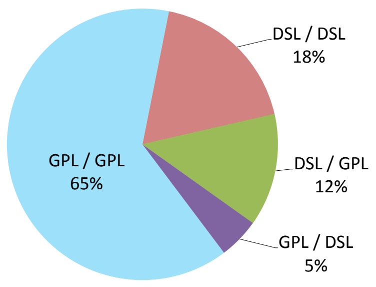


Figure 6: Distribution of metamodel-kind category

Fig. 6 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., [36]). In particular, (a subset of) UML is the most frequently used metamodel. General-purpose languages, such as Ecore (UML), XML [37], and Petri nets [5], 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 [38]. Most of the papers involving source and/or target domain-specific metamodels deal with business concerns, like QoS in [31] and business models in [39]. As another example, Zha et al. [40] describe an approach to verify workflow processes by translating them into Petri nets to check for some properties. Other domain-specific to general-purpose language transformations range from particular aspects of application development such as user-interface generation [41] to very specific domains like configuration of video surveillance systems [38]. Finally, a representative case of DSL-to-DSL transformations is the one in [42]. In this work, the transformation aims at migrating models expressed using a General Motors domain-specific language to AUTOSAR.

4.2.5 *Transformation language*

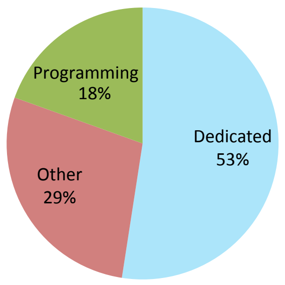


Figure 7: Distribution of tansformation-language category

As Fig. 7 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 [26] and QVT [39]), and the other half with less popular languages [25, 41].Only 18% of the papers still used programming languages for their transformations. These are mainly written in Java for the Eclipse platform [43], but also Prolog [37] and XSLT [44]. The remaining 29% of the papers only described the transformation, either without implementation or by implementing it in a one-time use tool [18]. These observations reveal that model transformation is being recognized as a paradigm in itself.

4.2.6 *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 [1]. 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. [45], 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 (e.g., see the work reported in [42]). Larger data models, from industry or that are publicly available, have been slowly gaining momentum in the past few years [9], as indicated in Fig. 8. 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.

4.2.7 *Validation*

Concrete model transformations are being validated empirically more often with time, as illustrated in the bubble graph of Fig. 8. This observation corroborates with a previous study in 2011 [46]. 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 [9], with a peak in 2012. Nevertheless, most validations were performed on small examples [39] which reduces the scalability of the validation to its peculiar context. Furthermore, Fig. 8 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 [45] and [9], propose strongly built validations, addressing both performances and accuracy aspects. Validation is also made by simulation as in [36] 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, e.g., [47].

4.3 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. 3a: 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 [1], 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.

EB ▶*modif*◀ 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, stated for RQ1, that model transformation is reaching.

5. THREATS TO VALIDITY

The goal of this study is to draw a global portray of the transformations that are proposed as research contributions. Although we followed a systematic process, some potential threats could limit the validity of our results. We next discuss the most important ones.

The first threat is related to the coverage of the existing literature. It is difficult to be exhaustive about all the published work on transformations for concrete problems. To maximize the coverage, we selected Scopus, one of the most exhaustive publication databases. However, as Scopus does not index Springer publications, we added manually all the papers from the four major forums in model transformation. **EB** ▶*TTC -> comment*◀ **EB** ▶*modif*◀ Even if we cannot guarantee a full coverage of the published material, our sample is large and representative enough to produce trustworthy results in terms of evolution trends and transformation characteristics.

The second potential limitation is the difficulty to encode the

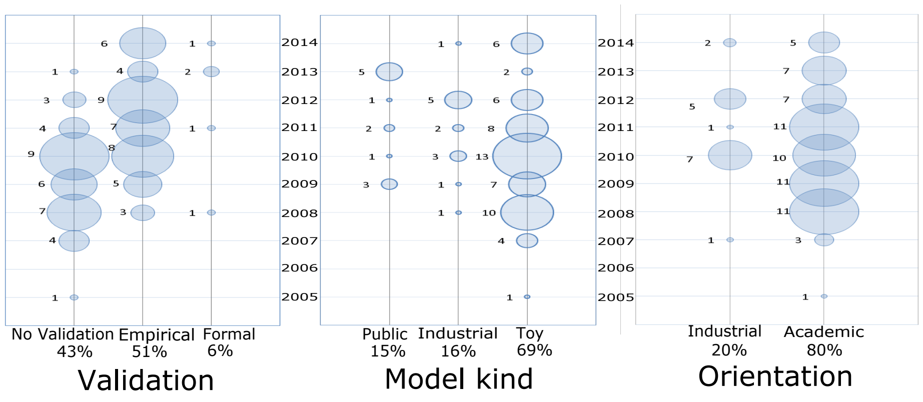


Figure 8: Evolution of validation, model kind and orientation, and their influences

scope of the study in a query. Indeed, there are no consensual terms to refer to concrete model transformation proposals. To avoid missing relevant paper, we opted for a generic query, which includes the context of MDE and only the term "transformation" and its variants. This, however, increased the number of paper to screen. **EB** ▶*modif*◀

The third potential threat to the validity of our study is related to the impact of the exclusion criteria. First, we excluded short papers, workshop contributions and gray literature. Because we are studying transformations as main research contributions, we only focused on long papers of the main research forums and neglected other contributions, such as technical reports and position papers. We are aware, however, that for some workshops, fully-fledged transformations can be published. The second impact of the exclusion criteria is the potential subjectivity of the decision whether a transformation is the main contribution of a paper. At least, one of the reviewers worked with a more relaxed meaning of this criterion, which produced the majority of the conflicting results. All these conflicts were resolved during the physical meeting and did not require the intervention of the third reviewer.

The fourth identified threat concerns the classification scheme. In our scheme, all the possible classes of a category are mutually exclusive, i.e., a paper cannot have more than one class for each category. Still, the classification was difficult for very few papers as more than one class was possible. This was particularly the case for the categories *Transformation kind* when the transformation involves structural and behavioral aspects, *Transformation language* when more than one language is involved, and *Validation* when both formal and empirical validations are performed. In such situations, we weighed the importance of each class and assigned the most important class. For example, in [6], C code is generated from structural and behavioral UML diagrams. We assigned the behavioral class to *Transformation kind*, since this transformation targets executable code.

6. RELATED WORK

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

In [15], the authors proposed a taxonomy of model transformation based on how models are manipulated and their execution strategies. Related to this contribution, in [4], the authors classified the features offered by languages to express model transformations. For a separate but related matter, the authors of [3] have cataloged the different use cases and intents where a model transformation can be used. We have taken into consideration all these aspects in the classification scheme we propose for concrete model transformations. However, unlike this paper, these studies were not performed following a systematic process.

Nevertheless, there have been several empirical studies about various aspects of MDE. Concerning the MDE adoption, the authors of [48, 10, 49] 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, Martinez et al. performed a user study to compare the performance of maintenance tasks when using an MDE approach against a code-centric approach [50]. Additionally, in [51], the authors propose a framework to empirically evaluate model-driven development of software product lines.

From a product perspective, Vanderose et al. propose an approach to empirically evaluate quality factors of developed artifacts in MDE [52]. In the same vein, in [53], the authors performed a series of user studies to assess the usability of web applications that have been developed in an MDE process.

The closest study to the one in this paper is [46], where the authors surveyed all publications from MODELS in order to understand the frequency with which empirical evaluations have been reported. The authors followed a systematic process, but with a narrower scope: MODELS papers only.

7. 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.

8. REFERENCES

[1] J. Whittle, J. Hutchinson, and M. Rouncefield, "The state of practice in model-driven engineering," *Software, IEEE*, vol. 31, pp. 79–85, 2014.

[2] G. Moore, *Crossing the Chasm: Marketing and Selling Disruptive Products to Mainstream Customers*. HarperBusiness Essentials, 2002.

[3] L. Lúcio, M. Amrani, J. Dingel, L. Lambers, R. Salay, G. Selim, E. Syriani, and M. Wimmer, "Model transformation intents and their properties," *Software & Systems Modeling*, pp. 1–38, 2014.

[4] K. Czarnecki and S. Helsen, "Feature-Based Survey of Model Transformation Approaches," *IBM Systems Journal*, vol. 45, no. 3, pp. 621–645, 2006.

[5] E. Syriani and H. Ergin, "Operational Semantics of UML Activity Diagram: An Application in Project Management," in *RE 2012 Workshops*. IEEE, 2012, pp. 1–8.

[6] M. Funk, A. Nyßen, and H. Lichter, "From UML to ANSI-C - An Eclipse-Based Code Generation Framework," in *International Conference on Software and Data Technologies*. INSTICC Press, 2008, pp. 12–19.

[7] A. Di Marco and S. Pace, "Model-driven approach to Agilla Agent generation," in *Wireless Communications and Mobile Computing Conference*, 2013, pp. 1482–1487.

[8] U. Winkler, M. Fritzsche, W. Gilani, and A. Marshall, "BOB the Builder: A Fast and Friendly Model-to-PetriNet Transformer," in *Modelling Foundations and Applications*, ser. LNCS, vol. 7349. Springer, 2012, pp. 416–427.

[9] T. Yue and S. Ali, "Bridging the Gap between Requirements and Aspect State Machines to Support Non-functional Testing: Industrial Case Studies," in *Modelling Foundations and Applications*, vol. 7349. Springer, 2012, pp. 133–145.

[10] P. Mohagheghi, W. Gilani, A. Stefanescu, and M. Fernandez, "An empirical study of the state of the practice and acceptance of model-driven engineering in four industrial cases," *Empirical Software Engineering*, pp. 89–116, 2013.

[11] R. France, J. Bieman, and B. Cheng, "Repository for Model Driven Development (ReMoDD)," in *Models in Software Engineering*, vol. 4364. Springer, 2007, pp. 311–317.

[12] M. Faunes, J. Cadavid, B. Baudry, H. Sahraoui, and B. Combemale, "Automatically Searching for Metamodel Well-Formedness Rules in Examples and Counter-Examples," in *Model-Driven Engineering Languages and Systems*. Springer, 2013, pp. 187–202.

[13] I. Baki, H. Sahraoui, Q. Cobbaert, P. Masson, and M. Faunes, "Learning Implicit and Explicit Control in Model Transformations by Example," in *Model-Driven Engineering Languages and Systems*. Springer, 2014, pp. 636–652.

[14] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic Mapping Studies in Software Engineering," in *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE'08. British Computer Society, 2008, pp. 68–77.

[15] T. Mens and P. Van Gorp, "A Taxonomy of Model Transformation," in *International Workshop on Graph and Model Transformation*, vol. 152, 2006, pp. 125–142.

[16] F. Jouault, F. Allilaire, J. Bézivin, and I. Kurtev, "ATL: A model transformation tool," *Science of Computer Programming*, vol. 72, no. 1-2, pp. 31–39, 2008.

[17] *Meta Object Facility 2.0 Query/View/Transformation Specification*, Object Management Group Std. formal/2015-02-01, Rev. 1.2, feb 2015.

[18] T. Yue, S. Ali, and L. Briand, "Automated Transition from Use Cases to UML State Machines to Support State-Based Testing," in *Modelling Foundations and Applications*, vol. 6698. Springer, 2011, pp. 115–131.

[19] A. Seriai, O. Benomar, B. Cerat, and H. Sahraoui, "Validation of Software Visualization Tools: A Systematic Mapping Study," in *Working Conference on Software Visualization*, ser. VISSOFT. IEEE, 2014, pp. 60–69.

[20] O. Díaz, G. Puente, J. Cánovas Izquierdo, and J. G. Molina, "Harvesting models from web 2.0 databases," *Software and System Modeling*, vol. 12, no. 1, pp. 15–34, 2013.

[21] J. Bach, X. Crégut, P. Moreau, and M. Pantel, "Model transformations with Tom," in *International Workshop on Language Descriptions, Tools, and Applications*, 2012.

[22] A. Blouin, B. Combemale, B. Baudry, and O. Beaudoux, "Modeling Model Slicers," in *Model Driven Engineering Languages and Systems, 14th International Conference*, 2011, pp. 62–76.

[23] Zhibin Yang, Kai Hu, Dianfu Ma, Jean-Paul Bodeveix, Lei Pi, and Jean-Pierre Talpin, "From {AADL} to Timed Abstract State Machines: A verified model transformation," *Journal of Systems and Software*, vol. 93, pp. 42–68, 2014.

[24] A. Harbouche, M. Erradi, and A. Mokhtari, "Deriving Multi-Agent System Behavior," *International Journal of Software Engineering and Its Applications*, vol. 7, no. 4, pp. 137–156, 2013.

[25] E. Syriani and H. Vangheluwe, "Programmed Graph Rewriting with Time for Simulation-Based Design," in *Theory and Practice of Model Transformation*. Springer, 2008, pp. 91–106.

[26] D. Wagelaar, L. Iovino, D. Di Ruscio, and A. Pierantonio, "Translational Semantics of a Co-evolution Specific Language with the EMF Transformation Virtual Machine," in *Theory and Practice of Model Transformations*. Springer, 2012, pp. 192–207.

[27] T. Gîrba, J.-M. Favre, and S. Ducasse, "Using Meta-Model Transformation to Model Software Evolution," *Electronic Notes in Theoretical Computer Science*, vol. 137, no. 3, pp. 57–64, 2005.

[28] Z. Demirezen, M. Mernik, J. Gray, and B. Bryant, "Verification of DSMLs Using Graph Transformation: A Case Study with Alloy," in *Workshop on Model-Driven Engineering, Verification and Validation*, 2009, pp. 1–10.

[29] K. Anastasakis, B. Bordbar, G. Georg, and I. Ray, "UML2Alloy: A Challenging Model Transformation," in *Model Driven Engineering Languages and Systems*. Springer, 2007, pp. 436–450.

[30] J. Cunha, J. P. Fernandes, J. Mendes, H. Pacheco, and J. Saraiva, "Bidirectional Transformation of Model-Driven Spreadsheets," in *Theory and Practice of Model Transformations*. Springer, 2012, pp. 105–120.

[31] Amogh Kavimandan and Aniruddha S. Gokhale, "Automated Middleware QoS Configuration Techniques using Model Transformations," in *Workshops Proceedings of the 11th International IEEE Enterprise Distributed Object Computing Conference*, 2007, pp. 20–27.

[32] Carsten Amelunxen and Andy Schürr, "Formalising model transformation rules for UML/MOF 2," *IET Software*, vol. 2, no. 3, pp. 204–222, 2008.

[33] M. Wimmer, A. Kusel, J. Schönböck, W. Retschitzegger, W. Schwinger, and G. Kappel, "On using inplace transformations for model co-evolution," in *2nd International Workshop on Model Transformation with ATL*, 2010.

[34] K. Garcés, J. M. Vara, F. Jouault, and E. Marcos, "Adapting transformations to metamodel changes via external transformation composition," *Software and System Modeling*, no. 2, pp. 789–806, 2014.

[35] Y. Xiong, H. Song, Z. Hu, and M. Takeichi, "Synchronizing concurrent model updates based on bidirectional transformation," *Software and Systems Modeling*, vol. 12, no. 1, pp. 89–104, 2013.

[36] J. Denil, P. J. Mosterman, and H. Vangheluwe, "Rule-based model transformation for, and in simulink," in *Spring Simulation Multiconference, Proceedings of the Symposium on Theory of Modeling and Simulation*, 2014.

[37] M. Eichberg, M. Monperrus, S. Kloppenburg, and M. Mezini, "Model-Driven Engineering of Machine Executable Code," in *Modelling Foundations and Applications*. Springer, 2010, pp. 104–115.

[38] M. Acher, P. Lahire, S. Moisan, and J.-P. Rigault, "Tackling high variability in video surveillance systems through a model transformation approach," in *Models in Software Engineering*. IEEE Computer Society, 2009, pp. 44–49.

[39] F. L. Siqueira and P. S. M. Silva, "Transforming an enterprise model into a use case model in business process systems," *Journal of Systems and Software*, vol. 96, pp. 152–171, 2014.

[40] H. Zha, W. M. P. Aalst, J. Wang, L. Wen, and J. Sun, "Verifying workflow processes: a transformation-based approach," *Software and System Modeling*, vol. 10, no. 2, pp. 253–264, 2011.

[41] Ó. Pastor, "Generating User Interfaces from Conceptual Models: A Model-Transformation Based Approach," in *Computer-Aided Design of User Interfaces V*. Springer, 2007, pp. 1–14.

[42] G. M. K. Selim, S. Wang, J. R. C., and J. Dingel, "Model Transformations for Migrating Legacy Models: An Industrial Case Study," in *8th European Conference on Modelling Foundations and Applications*, 2012, pp. 90–101.

[43] T. Buchmann, B. Westfechtel, and S. Winetzhammer, "ModGraph-A Transformation Engine for EMF Model Transformations," in *International Conference on Software and Data Technologies*. SciTePress, 2011, pp. 212–219.

[44] E. Saleh, A. Kamel, and A. Fahmy, "A Model Driven Engineering Design Approach for Developing Multi-platform User Interfaces," *WSEAS Transactions on Computers*, vol. 9, no. 5, pp. 536–545, 2010.

[45] F. Hermann, S. Gottmann, N. Nachtigall, H. Ehrig, B. Braatz, G. Morelli, A. Pierre, T. Engel, and C. Ermel, "Triple Graph Grammars in the Large for Translating Satellite Procedures," in *7th International Conference on Theory and Practice of Model Transformations*, 2014, pp. 122–137.

[46] J. C. Carver, E. Syriani, and J. Gray, "Assessing the Frequency of Empirical Evaluation in Software Modeling Research," in *EESSMod*, vol. 785. CEUR-WS.org, 2011.

[47] F. Steimann, "Constraint-Based Model Refactoring," in *Model Driven Engineering Languages and Systems*. Springer Berlin Heidelberg, 2011, pp. 440–454.

[48] J. Hutchinson, J. Whittle, M. Rouncefield, and S. Kristoffersen, "Empirical assessment of MDE in industry," in *International Conference on Software engineering*. ACM, 2011, pp. 471–480.

[49] P. Baker, S. Loh, and F. Weil, "Model-Driven Engineering in a Large Industrial Context—Motorola Case Study," in *Model Driven Engineering Languages and Systems*. Springer, 2005, pp. 476–491.

[50] Y. Martínez, C. Cachero, and S. Meliá, "Empirical study on the maintainability of Web applications: Model-driven Engineering vs Code-centric," *Empirical Software Engineering*, vol. 19, no. 6, pp. 1887–1920, 2014.

[51] J. Gonzalez-Huerta, E. Insfrán, S. M. Abrahão, and G. Scanniello, "Validating a model-driven software architecture evaluation and improvement method: A family of experiments," *Information & Software Technology*, vol. 57, pp. 405–429, 2015.

[52] B. Vanderose and N. Habra, "Towards a generic framework for empirical studies of Model-Driven Engineering," in *Workshop on Empirical Studies of Model-Driven Engineering*. CEUR-WS.org, 2008.

[53] A. Fernandez, S. Abrahão, and E. Insfran, "Empirical validation of a usability inspection method for model-driven Web development," *Journal of Systems and Software*, vol. 86, no. 1, pp. 161–186, 2013.