Evaluating the Requirements Engineering Process in Model Transformation Development: A State of Practice Analysis

Sobhan Y. Tehrani[1*], Kevin Lano[2], Mohammadreza Sharbaf[3], Shekoufeh Rahimi[4], Steffen Zschaler[2], and Shirin Hussein[1]

[1*]Department of Computer Science, University College London, UK
[2]Department of Informatics, King's College London, UK
[3]Department of Engineering, University of Isfahan, Iran
[4]Department of Computing, School of Computing, Engineering and the Built Environment, University of Roehampton, London, UK

{sobhan.tehrani@ucl.ac.uk, kevin.lano@kcl.ac.uk, m.sharbaf@eng.ui.ac.ir, shekoufeh.rahimi@roehampton.ac.uk, steffen.zschaler@kcl.ac.uk, shirin.s.hussein@ucl.ac.uk}

Abstract

Model Transformations (MT) are a central element of Model-Driven Engineering (MDE) methods. As MT adoption increases in both industry and academia, there is a growing need for systematic software engineering practices, particularly in Requirements Engineering (RE) for MT development.

This paper investigates the state of RE in MT through two complementary empirical studies: semi-structured interviews with industry practitioners and a systematic literature review (SLR) analyzing published transformation cases. Both studies address the same research questions but differ in the populations they cover. The interviews focus on industrial settings, while the SLR reviews published work, the majority of which comes from academic sources. Our findings reveal that the RE processes used in MT development tend to be largely informal and lack structured methodologies. While some RE techniques such as prototyping and scenario-based generalization are used, they are typically applied in an ad-hoc manner based on personal experience rather than through a well-defined RE framework.

Our studies highlight challenges in stakeholder engagement in MT RE, particularly limited access to stakeholders, which restricts the effective application of RE techniques. Furthermore, our analysis identifies a predominant focus on MT implementation, with limited MT specification and systematic RE activities, which often leads to requirements being implicitly defined rather than explicitly documented. Despite these shared findings, the interview study and SLR differ in their perspectives: the interview study reflects real-world industrial constraints on requirements engineering, while the SLR reflects more research-driven RE practices.

These findings underscore the gap between research and practice in model transformations, and highlight the need for lightweight, structured RE frameworks tailored to MT development. Future work should focus on bridging this gap by integrating agile RE techniques with structured methodologies to support flexibility, traceability and stakeholder collaboration in MT projects.

1 Introduction

In this paper, we describe the results of empirical studies on requirements engineering (RE) in model transformation (MT) development. Interviews were conducted with industrial and

academic MT practitioners, and a systematic literature review was conducted. Our aim was to understand:

1. the current state of practice in requirements engineering for MT;
2. the issues involved in applying RE to MT development; and
3. the appropriate processes and techniques for requirements engineering in this context.

To this end, we need to understand the context in which model transformations are typically developed and what, if any, requirements engineering techniques are already applied. Gaining such understanding can help practitioners and researchers determine how existing RE techniques might be applied or adapted for the specific context of MT development.

1.1 Model Transformations

Transformations are used widely in model-driven engineering (MDE) and model-based development (MBD), their uses include *migration* of models from one language to another such as in [37], *refactoring* of models to improve quality such as [27], *refinement* of models from a specification to a design, or from design to implementation such as in [6], *code generation* to generate program code from models (model-to-text transformations) [10], *reverse-engineering* transformations mapping code or other text to models [23], and *bidirectional* transformations to synchronise two different models and to maintain their consistency [8]. Transformations can also be used for *data analysis* to extract and analyse information from models such as in [13]. *Semantic mapping* transformations map a model to a semantic domain to support precise analysis. An example is the transformation of Unified Modeling Language (UML) to Alloy [7]. Transformations are often categorised as model-to-model (M2M), model-to-text (M2T) or text-to-model (T2M) depending on whether their inputs/outputs are instance models of a particular metamodel, or are textual representations such as programs or reports.

It is important to distinguish between model transformation specification languages and transformation implementation languages, as they serve different roles in the development of model transformations. Model transformation specification languages define transformation logic at a high level, typically using a declarative approach that describes what is the intended relationship between the input and the transformed output without specifying how the transformation is executed. Examples of such languages include Query View Transformation (QVT) [34], Atlas Transformation Language (ATL) [15], ATOM [43] and $\mathcal{MT}$ [20], which allow transformations to be formally specified using structured rules.

Transformation implementation languages, in contrast, encompass both specification and implementation aspects, providing the means to execute transformations. Some, such as QVT Operational [34] and Epsilon Transformation Language (ETL) [18], follow a procedural paradigm, defining explicit transformation steps. Others, like ATL or UML-RSDS [25], support a hybrid definition approach, emphasizing rule-based mappings.

1.2 Requirements Engineering

RE is the process of identifying, analysing, documenting and validating the requirements of an application. This is often considered as only an initial stage of an overall development process, but it would be more accurate to regard it as an ongoing activity which should be engaged in during all development stages.

Kotonya *et al.* [40] have proposed a four-stage process model for the RE process. This model is widely accepted by researchers and professional experts. In this study, we have used this model as our template to investigate MT projects. According to this process model, the following are the most important phases of RE which have to be applied: (i) Domain analysis and requirements elicitation; (ii) Evaluation and negotiation; (iii) Specification and documentation; (iv) Validation and verification.

The initial stage (domain analysis and requirements elicitation) in RE is the process of obtaining a great deal of knowledge regarding the domain of the current problem, the organization/company confronting the problem and the existing system that is facing the problem. Once the required knowledge has been acquired, a draft document could be provided which would help the system developers to understand the context of the actual problem as well as to identify the stakeholders' actual needs and requirements. At the stage of evaluation and negotiation, it is assumed that the previous stage, requirements elicitation, has been performed effectively. The evaluation stage considers the consistency and feasibility of the requirements. Inconsistencies among requirements may exist, the chances of which will increase if the requirements have been gathered from multiple different stakeholders. Sometimes, such inconsistencies could even result in having conflicts between the requirements. Through negotiation between stakeholders and developers, conflicts and other problems are addressed and resolved. The specification and documentation phase begins with the specification process, which produces a set of precise agreed statements of the requirements, assumptions and system properties. Based on the results of the specification, the requirements specification documentation can be drafted. At the validation and verification stage, specifications must be analysed. They should be validated by stakeholders in order to check that the documented requirements accurately express their actual needs. Also, specifications should be verified in order to check their consistency and avoid conflicts and omissions. Any potential errors and flaws must be fixed during this phase and before implementation in order to save cost, effort and time.

1.3 Requirements Engineering for Model Transformations

Similar to any other software applications field, MT development also requires an appropriate RE process in order to create correct transformation applications. In this section, we introduce key characteristics of MT development that affect how RE can or should be applied. These are based on observations from existing literature and known MT practices:

- Unlike mainstream software development, MT development is usually concerned with the construction of applications to process software models, using specialised MT languages such as ATL, QVT-R and graph transformation languages. These languages are generally more declarative, and involve coding at a somewhat higher abstraction level, compared to mainstream programming languages such as Java, C++, etc., and they contain specialised constructs to select, create and modify model elements. However MT development often appears to also follow a traditional 'code and fix' process in many cases [38], and many common coding quality flaws such as code duplication and high coupling also appear in MT applications [17].

- Transformations are usually structured around transformation rules and sets of rules, together with auxilliary elements such as helper functions. This differs from the class-and-method structure typically found in object-oriented programming languages (Java or C++), and often results in less modularity. Transformation requirements often reflect this structural aspect, for example, they can be expressed in terms of how particular source model elements should be mapped to target model or text elements, for example "Each UML class should be represented as a C struct, with fields for each attribute of the class" [24].

- MT development is used in MDE to automate a particular process step, such as the generation of code or documentation from a model, or for the generation of a model of one metamodel from a model of a different metamodel. As such, the stakeholders of the MT project may be the developers of the larger MDE project, rather than customers/stakeholders of the larger project. This complicates the RE process because the MDE project developers may have misunderstandings of what MT requirements are needed in order to satisfy the goals of the wider project.

- The output model of one MT may be used as an input model to another. In such a case the expectations/assumptions made about the input model by the second transformation will need to be reflected in the requirements of the first.

In order to achieve any given goal using software (such as in model transformation), having a scheme in which its requirements have been identified is essential. According to research on MT development, requirements engineering appears to be a relatively neglected aspect of model transformation development: the emphasis in transformation development has been upon implementation in specific transformation languages [9], [26]. The failure to explicitly identify requirements may result in developed transformations which do not satisfy the needs of the transformation users. Problems may arise because implicitly-assumed requirements have not been explicitly stated, or because of insufficient understanding of explicit requirements. It might be possible to skip the requirements engineering process in small projects and jump directly into the implementation phase. However it is unlikely to be possible and effective in large and industrial projects. This paper will provide a comprehensive view of the current state of RE for MT, and identify how RE for MT can be improved and made more systematic.

An initial version of the interview study was previously published in ICMT 2016. The current paper extends the survey with three new projects, with revised and more detailed conclusions, and with a new literature survey (SLR) of the same topic. The nature of MT outputs whether intermediate artefacts or final deliverables affects stakeholder roles and requirements. When the output is user facing, direct stakeholder involvement is crucial; for intermediate outputs, requirements may come from developers. This variation highlights the need for a systematic RE approach in MT projects. In the following sections, we first review related work on requirements engineering and model transformation (Section 2). We then present our research methodology, including the interview study and systematic literature review (Section 3). Section 4 describes the MT projects included in the interview study and summarises their RE processes and outcomes. Section 5 introduces a theoretical framework for RE in MT, derived from the interview results. Section 6 presents the findings of the systematic review of the literature. Section 7 compares the two studies, Section 8 discusses threats to validity, and Section 9 concludes the article with key findings and directions for future work.

2 Related Work

There has been very limited empirical research into model-transformation development. The main related studies have been based on MDE in general, such as that of [14, 45], which used interviews as well as a questionnaire-based survey. The main aim of these studies was to capture the success and failure factors for MDE based on industry evidence. Their work conducted 22 interviews with MDE practitioners and surveyed over 400 MDE practitioners via a questionnaire. The survey found that some use of MDE is made in a wide range of companies and industry sectors, however this use tended to be based on Domain-Specific Languages (DSLs) and modelling of narrow specialised domains. Transformations were used to generate artefacts from the DSL models, however code generation was not itself a primary benefit of MDE, instead the benefits came from the ability to abstract system architectures and concepts into models. The evidence from their research suggests that transformations are often developed based on the expert knowledge of software developers, to encode and automate previously manual procedures. It is unclear from this research if a systematic approach or any relevant requirements engineering techniques (such as observation) are used for the transformation developments. A high degree of domain knowledge appears essential for the successful construction of the transformations. The MDE experience research of [33] considered in depth 4 cases of MDE application, but did not specifically consider the requirements engineering of these cases. One concern of the companies in [33] was the cost of developing transformations, a factor which could be improved by more systematic RE for MT. The survey of [29] considers the use of RE in MDE, and concludes that the use of rigorous techniques for RE in MDE is limited: the majority of surveyed cases did not have tool support for RE, and in most cases the RE process was not integrated into the

Survey	*Period*	*Cases*	*Purpose*
[3]	2005–2014	82	Transformation type, outcome
[28]	2000–2016	519	Transformation pattern use
[17]	2000–2019	503	Transformation quality

Table 1: Surveys of model transformations

MDE process. These results are consistent with our own findings for RE use in the more specialised field of MT development. A more recent general survey of MDE is [2]. This also shows widespread use of model transformations, but also continuing problems with a lack of adequate training and tool support for transformation languages, and a lack of systematic development processes.

Previous large-scale surveys of model transformations include [3], [28] and [17]. The survey [3] of concrete MT developments analyses 82 MT cases between 2005 and 2014 with regard to their type and outcomes, but does not specifically consider RE aspects. The survey of [28] analyses 519 transformation cases between 2000 and 2016, with regard to their use of transformation patterns. They find that only 44% of cases use patterns, although this percentage has increased over time. There is no consideration of the relation between transformation requirements and pattern use. The paper [17] surveys 503 transformation cases in four transformation languages, selected from transformation repositories for these languages. The time span appears to be from 2000 to 2019. They find that only 2% of these cases are industrial cases. Quality requirements for MT are formulated. Table 1 summarises the previous surveys of model transformations.

The papers [5] and [12] consider the current usage and future prospects for specialised model transformation languages, they identify that many practitioners prefer to use general-purpose languages such as Java to write transformations, due to the relatively poor knowledge and tool support available for MT languages.

There has been limited research on RE techniques for MT. The transML methodology defines an outline RE approach and methods for the RE of MT [9]. They adopt SysML and scenarios to analyse and document requirements. The papers [21], [22] define NLP-based techniques for the formalisation of transformation requirements starting from structured natural language statements. These researches are the only ones we have identified which provide tools for the RE of MT, however these tools are at the stage of research prototypes.

3 Research Methodology

This study employs two complementary qualitative research methods to investigate the role of Requirements Engineering (RE) in Model Transformation (MT) development: a semi-structured interview study with industry practitioners and a systematic literature review (SLR) analysing published transformation cases. These methods provide qualitative insights from both industry professionals and existing academic research, providing a comprehensive understanding of the topic.

3.1 Methodology of the Interview Study

The first stage of this research followed a qualitative research approach to gain a deeper understanding of RE practices in MT development. This helped identify key concepts and challenges in the field, enabling the formulation of an initial theoretical framework. The study aimed to explore transformation projects from an RE perspective, focusing on identifying whether and how RE techniques are applied in MT development and whether the absence of systematic RE processes causes issues.

To achieve this, we conducted an exploratory interview-based study involving seven experienced MT practitioners, each with eight to twenty years of experience in MT development.

The selection criteria were based on their industrial experience and published work in the field. The participants were independent from the authors, except for projects 4 and 5 (carried out by author 2). The interviewees discussed ten self-selected projects across various domains, such as embedded systems, financial applications and software re-engineering.

We employed semi-structured interviews, allowing for flexibility in responses and enabling spontaneous follow-up questions during discussions. This qualitative approach enabled the identification of underlying patterns and decision rationales behind the use of RE techniques in MT development, which are often not captured through quantitative methods. Each interview lasted between 45 and 60 minutes and was conducted by the first author. The interview prompts used in this study are available at `https://nms.kcl.ac.uk/kevin.lano/reform.pdf`.

All participants received a detailed information sheet describing the study and provided informed consent before participation. The interviews were recorded, transcribed, and thematically analyzed to extract key findings related to the role of RE in MT development.

3.2 Methodology of the Systematic Literature Review (SLR)

To complement the insights gained from the interviews, we conducted a systematic literature review (SLR) following the methodology proposed by Kitchenham [16]. An SLR is a qualitative method that provides a structured means to identify, evaluate, and interpret relevant research sources related to a specific research question, area, or phenomenon of interest.

The SLR process consisted of three main stages, as defined in [16]:

- **Planning the Review:** We identified the goal for the review and formulated specific research questions to guide the selection of relevant studies.
- **Conducting the Review:** We systematically searched relevant digital libraries, including IEEE Xplore, ACM Digital Library, ScienceDirect, Springer Link, and Google Scholar, using predefined search queries. The initial search identified 1,884 papers, which were filtered using inclusion and exclusion criteria, resulting in 262 transformation case studies for final analysis. Each study was independently labeled and reviewed by one researcher and then peer-reviewed by another to ensure accuracy. The interviews were conducted by the first author and transcribed and verified by the second author. The SLR analysis were collaboratively carried out by all authors, with peer verification performed throughout the process.
- **Reporting the Review:** The extracted data was synthesized and analyzed to identify trends, gaps, and common themes related to RE in MT development. Section 6 provides further details on the findings.

The SLR evaluated transformation cases based on multiple factors, including the presence or absence of RE processes, the types of RE techniques used, stakeholder engagement strategies, and the categorization of transformation requirements. The results were then compared with findings from the interview study to identify alignments, contradictions, and gaps between research and industry practice.

3.3 Relationship Between the Two Studies

Although the interviews and the SLR were conducted independently, they serve complementary roles in understanding the state of RE in MT development. The interview study in Section 5 provided first-hand insights into the real-world challenges faced by MT practitioners, identifying key criteria such as the type of model transformation, RE techniques used, and stakeholder involvement. These criteria were derived directly from the experiences and challenges reported by practitioners. In Section 6, the SLR was designed to validate and extend these findings by decomposing its research question into the same seven criteria as the interview framework (detailed in Section 6.5) to examine a broader range of academic and industrial studies. Specifically, in Section 6.5, we compared the criteria identified in the interviews with those found in

the literature. Out of the seven criteria analyzed, three (e.g., type of model transformation, RE technique, and stakeholder engagement) showed strong alignment between the interview findings and the SLR results. This overlap shows the potential significance and generalizability of the criteria identified in Section 5.

Several key themes from the interviews, such as the lack of systematic RE processes, challenges in stakeholder engagement, and prioritization of implementation over formal RE, were also observed in the SLR results. However, notable differences exist between academic and industrial perspectives, emphasizing gaps that future research must address.

This study serves as an initial step in formulating a theory on RE for MT development, which must be further validated through additional empirical studies. Our approach aligns with established methods of theory building in software engineering, as discussed by Sjøberg et al. [39], Wohlin [47], and Stol and Fitzgerald [42].

4 Interview Study on Transformation Development Projects

This section investigates the MT development projects from our interview study, categorising them based on their development characteristics, project types and transformation goals. The objective is to identify common RE challenges and patterns across different project types and analyze how these factors influence project success. This classification helps in understanding how different transformation development approaches align with RE best practices and stakeholder needs.

4.1 Projects of the Study

All of our interviewees were either the sole developers or the lead developers for these projects. Each project is categorised according to the MT field that it belongs to. The scale, developers time and effort for these projects will also be described. Table 2 provides an overview of these projects.

Regarding project scale, we used the categories (Large, Medium, Small) for the project size, with the following definitions:

- Large: $\geq$ 200 rules
- Medium: $\geq$ 50 rules, $<$ 200 rules
- Small: $<$ 50 rules.

The scale of the interview cases is then shown in Table 2.

In detail, the ten MT development projects considered in this study had the following goals and contexts:

P1. ***Automated generation of documentation for international standards:*** This transformation concerns the generation of standard documentation text from meta-models, to ensure consistency of the documentation. The source meta-models are large-scale: of the order of 600 meta-classes. There are one or more transformation rules for each meta-class. It can be regarded as a code-generation transformation (model-to-text).

P2. ***Reverse-engineering and re-engineering of banking systems and web-services:*** The aim of this project was to build transformations to reverse-engineer models of existing applications, and to forward-engineer these models to new platforms. The legacy systems processed by these transformations are very large: the scale of the finance system re-engineering is approximately three million LOC extracted from 100 million LOC legacy code, the scale of the web services re-engineering is approximately 15 million LOC. The re-engineering process must be done in a way that not only reveals the actual functionality of the system, but also enables further analysis according to system requirements. This project involved both reverse-engineering and code-generation transformations.

Table 2: MT projects in interview study with transformation type and scale

Project ID	Project Name	Type	Scale
P1	Automated generation of documentation for international standards	M2T	Large (approximately 600 rules)
P2	Reverse-engineering and re-engineering of banking systems and web services	M2M, M2T	Large (over 1500 transformation rules)
P3	Code generation of embedded software from DSLs	M2T	Large (about 500 transformation rules)
P4	Petri-net to state chart mapping	M2M	Small (about 20 rules)
P5	Big Data analysis of IMDb	M2M	Medium (approximately 30 rules)
P6	UML to C++ code generator	M2T	Large (a few hundred transformation rules)
P7	Reverse-engineering of a code generator	M2M, M2T	Small (described as relatively low scale in interview)
P8	Automation of railway network engineering	M2M, M2T	Large (over 200 entities and transformation rules)
P9	MDE Platform	M2T	Large (a substantial MDE platform)
P10	SOA for insurance	M2T, M2M	Medium

P3. ***Code-generation of embedded software from DSLs:*** Transformations are defined to map between embedded system DSLs, and from these DSLs to C code. These DSLs are used by embedded software developers. More than 25 different DSLs are involved. This is a code-generation case.

P4. ***Petri-net to statechart mapping:*** This model transformation maps Petri-net models to statecharts, in order to analyse the Petri-nets. It involves both refactoring and migration aspects. The transformation is intended to map large-scale models with thousands of elements. This is a semantic mapping transformation case.

P5. ***Big Data analysis of IMDb:*** The Internet Movie Database (IMDb) can be regarded as a Big Data case. It has information about the title of movies, name of actors, rating of movies and actors playing roles in which movies. The model transformation in this case computes search results for users. For instance, it enables the user to find the pair of actors with the highest average rating over all the movies in which they have both appeared. For this type of transformation, the database is regarded as a model while the transformation is focused on extracting the relevant information. This is a data analysis transformation case.

P6. ***UML to C++ code generator:*** This project involved the construction of a transformation for the generation of multi-threaded/multi-processor code from UML. The idea of this project is to generate C++ code as well as providing a run-time layer to support the generated code. This is a code-generation case.

P7. ***Reverse-engineering of a code generator:*** This MT project was an example of re-engineering of an existing transformation. In this project an existing code-generation transformation was analysed and re-engineered to improve its functionality. This project also concerned a code-generation transformation.

P8. ***Automation of railway network engineering:*** This project involved using models and transformations to support railway network design. This is a safety-critical case, with approximate value of £150,000 per year. The metamodels operated on have about 200 classes. This project involved both analysis and code-generation transformations.

P9. ***MDE Platform:*** This concerns the development of a generative software platform based on transformations and DSLs. This is a large-scale ongoing project. This project primarily involved code generation.

P10. ***SOA for insurance:*** This project concerned the generation of middleware from DSLs, for service integration. This is also a code generation case.

Thus 8 of the projects involve code generation, two are analysis projects, one is a reverse-engineering transformation and one is a semantic mapping.

4.2 Types of Software Projects

Software development projects can be classified into general several categories [44]:

Greenfield vs Brownfield Software can either be built from scratch or it can be built upon an already existing system which needs to be improved, integrated or extended. In a greenfield project, the system is completely new, therefore the developers have to entirely create the system. On the other hand, in brownfield projects, a system already exists but it has to be further developed and improved. In this case, developers work on the current system and extend its functionalities.

Customer vs Market Driven Software could be either a solution for a particular type of client in the market (customer driven) or a solution which would cover the need of a large percentage of the market (market driven). In customer-driven projects, the software is designed according to the needs of a specific type of client, whereas in market-driven projects, a larger scope of solution is considered covering more than just one particular type of client.

In-House vs Outsourced A project could be regarded as either an in-house project where it is assigned to a particular organization in order to carry out all the project's life-cycle processes or it could be outsourced where it is assigned to different companies according to different project phases. In an in-house project, one team/company will carry out all the phases in the project, whereas in an outsourced project, usually once the requirements have been identified different teams from different companies will carry out the different phases such as design, implementation, testing, etc.

Single Product vs Product Line The outcome of a project could have only one version which would satisfy the customer's need or it could have different versions each of which would cover particular needs in a large organisation. "In a single-product project, a single product version is developed for the target customer(s). In a product-line project, a product family is developed to cover multiple variants" [44].

Table 3 highlights common trends in MT project types. Most were greenfield and customer-driven, reflecting a focus on creating tailored solutions for specific organizational needs. All were in-house and followed a single-product strategy, suggesting that MT development is often tightly scoped. The inclusion of two academic projects also provides contrast with industrial practices.

5 Framework for RE in MT

In this section, we analyse the requirements engineering processes and project outcomes of the interview study projects in detail. We use this analysis to identify key factors in and distinctive aspects of the RE of model transformations. These aspects include the development context, stakeholder roles, project methodology and project outcomes. The result is an outline framework for the RE of MT. We also relate this framework to relevant previous research in RE and MDE. The analysis was conducted by the first two authors based on inspection of the interview transcripts.

Table 3: MT project types in interview study

Project ID	Brown-field	Green-field	Customer-driven	Market-driven	In-house	Out-sourced	Single-product	Product-line	Context
P1		✓	✓		✓		✓		Industrial
P2		✓	✓		✓		✓		Industrial
P3	✓		✓		✓		✓		Industrial
P4		✓	✓		✓		✓		Industrial
P5		✓	✓		✓		✓		Industrial
P6		✓	✓		✓		✓		Industrial
P7	✓		✓		✓		✓		Industrial
P8		✓	✓		✓		✓		Industrial
P9	✓			✓	✓		✓		Academic
P10		✓	✓		✓		✓		Academic

5.1 The Context of MT Development

As we noted in Section 1.3, MT development often occurs as an internal project within a wider software development project. We found that projects P2, P3, P4, P6, P7, P8, P9 and P10 from the interview cases are of this kind, although there are also cases where MT development is the main part of software development (projects P1 and P5).

Therefore, we consider it is useful to differentiate explicitly between properties of an internal transformation-development project and of the wider project that this is embedded in. For example, while most of the containing projects were brownfield projects, most of the transformation-development projects were greenfield because no previous transformation existed for the specific purpose required.

In particular, whereas the deliverables of an overall project will be directly used by the project customers, deliverables from an internal MT project may not be visible to such customers, and instead are only used by the main project developers. Thus RE processes and artefacts for the internal project will need to be distinguished from those of the containing project.

5.2 MT Project Stakeholders

In general, the term stakeholder can be defined as an individual or a group of people who either have an interest in, or an effect upon, the outcome of a given project [36]. It is essential to fully identify all the stakeholders of the project as an initial step prior to any other action, because by missing an important group of stakeholders, there is a major risk of missing a whole set of requirements of the system. In addition, a good participation of stakeholders in the software development cycle should not only result in a better understanding of the actual problem, but also help to build the system which is required to satisfy the stakeholders' needs.

The onion model of project stakeholders (*e.g.,* [1], see Figure 1) has been used to describe different types of stakeholders and their relation to the system under development. In this model, stakeholders are categorised into three different types. Operational stakeholders have a direct interaction with the system. Stakeholders in the containing business area benefit in some manner from the system. The wider environment area contains stakeholders which have an effect upon or interest in the system, but only an indirect influence.

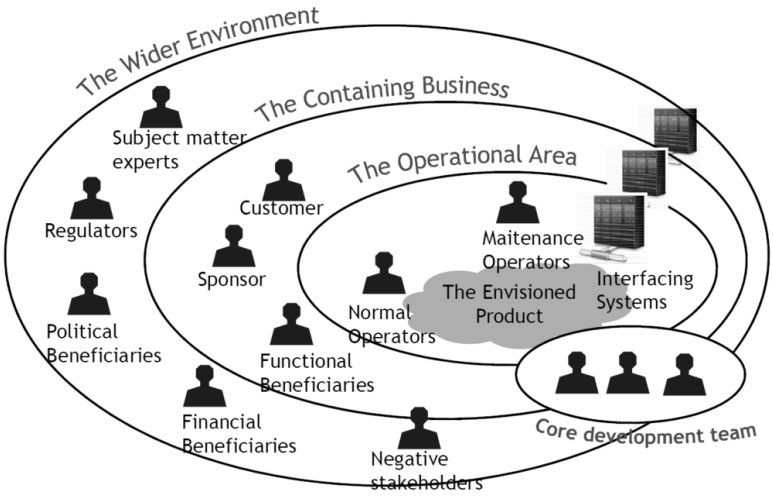

Figure 1: Onion model of stakeholder general relationships [44]

More specifically, sponsors are stakeholders that have the responsibility to pay for the development of the product. Customer(s) buy the product. The customer(s) may also be the end user(s) of the developed product. The normal operators are the people who will eventually operate and use the developed product. The maintenance operators are the people from which the maintainability requirements can be discovered. The core development team consists of developers that are in charge of developing the product. Subject matter experts could consist of "internal and external consultants, which may include domain analysts, business consultants, business analysts, or anyone else who has some specialized knowledge of the business subject" [36].

The onion model is particularly effective because it provides a clear visual representation of stakeholder relationships, highlighting their varying degrees of influence and interaction from the core (primary stakeholders) to the outer layers (secondary and tertiary stakeholders). This layered structure helps us to understand stakeholder dynamics better and identify the key players essential to a project's success. While we also considered other models, such as the power-interest grid [4] and the salience model [32], we opted for the onion model for its straightforward approach in categorising stakeholders based on their proximity and relevance to the system. Its visual clarity and simplicity are especially useful in addressing the complex interactions among stakeholders in our study, ensuring our analysis is focused and comprehensive. Choosing the onion model over other frameworks aligns with our aim to present a clear, easily understandable stakeholder analysis that aids in informed decision-making and strategic planning.

We initially used the above general onion model to classify the stakeholders in MT development, based on the interview study descriptions. For these MT project cases we can identify that the core development team consisted of the transformation developers for all of the MT projects. The customer(s) were represented by personnel who interacted with the transformation developers in order to explain the problem space and what was needed. The sponsor(s) were companies which in most cases were also specific customers. For P9, the sponsor was the software provider, who was developing the software for purchase by prospective customers (MDE users). Finally, the normal and maintenance operators consisted of the people who were going to use the result of the transformations as end users. Table 4 presents the sponsors, customers and the operators of the interview study MT projects.

As discussed earlier, the MT projects that we analysed from the interview-based study are typically embedded within wider projects. Depending on whether the output of the transformation is an intermediate product or the final software delivered to users, the roles and influence of stakeholders vary significantly. For instance, when the MT output is a final product (P6 or P10), end users directly interact with it, making their requirements critical. In contrast, when

Table 4: Interview-based study MT stakeholders

Project ID	*Sponsor and Customer*	*Normal and Maintenance Operator*
P1	Technology standards consortium	Users of the standards
P2	Financial/Telecom organisations	Users of re-engineered systems
P3	Commercial companies	Embedded software developers
P4	External customer	Users of the produced models
P5	External customer	Users searching the data
P6	Government & Defence industries	Users of C++ applications
P7	Commercial client	Users of the code generator
P8	External customer; parent company	Railway engineers and operators
P9	Own company; MDE users	Consultants, toolkit customers
P10	Insurance company	IT team of company

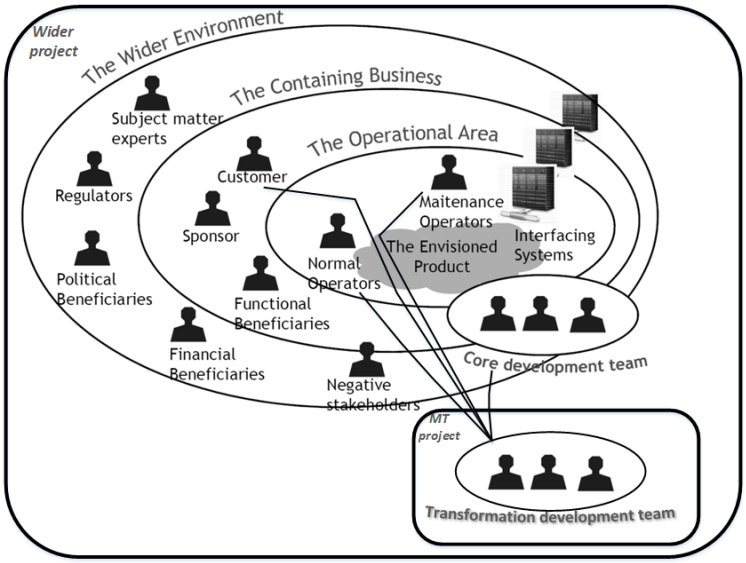

Figure 2: Onion model of MT stakeholder relationships

the transformation output is an intermediate artefact used by other tools or teams (P1 or P4), the stakeholders shift towards internal developers or analysts. This variation highlights the need for a more systematic RE process in MT development, as assumptions about stakeholder roles and requirements can otherwise be overlooked.

Similarly, the impact of other stakeholders of the containing project (*e.g.,* from the containing business or wider environment) on the transformation development can become more indirect. Understanding fully the role of these stakeholders in the context of transformation development seems important for successfully developing requirements engineering techniques for MT development and will be part of our focus for future work.

As a result of these considerations, we propose the following adapted onion model (Figure 2) to represent the typical relationships of MT developers and stakeholders for an MT development embedded in an MDE process.

Table 5: Requirements engineering factors in MT projects

Factor	Description	Examples
System Domain	Critical systems need a complete and consistent set of requirements that can be analysed in advance. For business systems, work can start with an outline of the requirements, which are then refined during development.	Projects P2, P3, P6, P8 concern critical systems.
Type of Development Process	Staged development requires all requirements upfront, whereas agile allows incremental refinement.	Projects P2, P3, P8 used an agile methodology.
Stakeholder Involvement	Limited stakeholder access can cause misunderstandings, rework, and project delays. A lack of direct communication may increase development effort, making requirement gathering difficult.	Projects P3, P6 experienced this issue.
System Reuse	Generally, the original requirements of reused systems are not available, so the RE process must reverse-engineer them from the system [41].	Project P7 experienced this issue.

5.3 Overall Requirements Engineering Process

Requirements engineering (RE) for any particular type of software development involves specialized aspects, and model transformations are not an exception. Several factors influence the RE process in MT projects, including the system domain (critical or non-critical), type of development process (such as agile methods or staged development), stakeholder involvement and system reuse. Table 5 summarizes these key factors and gives examples from the studied projects where they impacted on the RE process.

As discussed in Section 5.1 above, MT developments are often auxiliary relative to some larger software project, they provide infrastructure (e.g., code generators or data migration tools) that are required by the larger project. Access to stakeholders of the main project may therefore be limited, even if the task itself is critical (if a code generator is to be used to synthesise safety-critical application code, then the generator itself is critical). In Projects P3 and P6, stakeholder access was restricted due to organizational structures and security constraints. Project P3 faced delays and rework as key stakeholders were external, forcing developers to rely on assumptions and intermediaries, increasing the risk of misinterpreted requirements. Similarly, in Project P6, security regulations in government and defense industries limited direct communication, making requirement elicitation dependent on indirect reports and documentation. This led to incomplete or unstable requirements, resulting in costly rework. These challenges emphasize the need for structured stakeholder engagement and alternative RE techniques, such as prototyping and scenario-based validation, to address access limitations in complex projects.

The relevance of agile development for MT construction deserves further investigation. Because a wider project may depend on MT functionality being available as soon as possible, reducing time-to-delivery in such internal MT projects is desirable. In some cases (e.g., projects P2 and P8) the agile principle of incrementality, to deliver functionality quickly to clients, appeared to be beneficial, whilst in others (e.g., project P3) the agile emphasis on coding conflicted with the need to commit modelling effort to build and refine precise and stable metamodels which transformations operate on. Ongoing access to customer representatives during development is also important for effective use of agile methods, and problems with this access impacted project P3 in particular.

5.4 Requirements Ambiguity, Infeasibility, Change and Conflicts

In any software development project, infeasible, ambiguous or conflicting requirements may arise. Requirements change and evolution is also ubiquitous. Requirements may change during

Table 6: Requirements problems in MT interview study projects

Project ID	***Problem***	***Response, paraphrased from participant comments***
P1, P2, P3, P4, P6, P7, P8	Unrealistic requirements	– Implementing "what is needed" rather than what is wanted – Implementing "the underlying system" – Providing customers with an estimated cost update
P1, P2, P3, P6, P7, P8	Change of requirements	– Agile provides sufficient time via weekly deployments – Confirming the requirements at the beginning of every iteration – Charging extra for the additional requirement(s)
P1, P2, P3, P4, P5, P10	Requirements conflict	– Resolving the conflict by common sense – Trade-off amongst the conflicting requirements
P2, P3, P4, P5, P6, P7, P8	Requirements uncertainty	– Contacting the stakeholders for clarifications

the development life cycle due to stakeholder's change of mind/circumstances, or because of the introduction of new requirements additional to existing one(s). The requirements engineer may also fail to initially recognise certain requirements, and these are only added after the omissions are discovered. Based on our interview study, we identified that transformation developers often had to deal with requirements modifications, unrealistic and ambiguous requirements and with conflict amongst requirements. A typical quote from the interview responses was:

> "Don't do what you are told, but always do what is needed" (Interview study participant).

In Table 6, we list the MT developer comments from the interview study regarding common requirements problems that may occur during the MT development.

5.5 RE Process Stages for MT

According to our investigation of the interview cases, the requirements engineering process in model transformations typically starts with requirements elicitation via an initial meeting with customers. Their input is central to the process at this stage.

> "It is the process and an engagement that starts with the customer" (Interview study participant).

An online conference may be used for convenience, especially if stakeholders are physically remote from the development team. Projects P1, P2, P3 and P6 used online conferences.

Customers sometimes have only a very high-level view of what they need the transformation to achieve. For instance, a customer may only be aware of the language that his/her company want the code to be generated into or the kind of platform.

> "Stakeholders are not very technical but they know what they need to see out of the system at the end" (Interview study participant).

Therefore, transformation developers may suggest joint sessions with the stakeholders in order to gain explicit knowledge about the system. During these sessions interviews and brainstorming methods are applied to confirm the functional and non-functional requirements and specifications in more detail. Joint requirements development sessions were used in projects P2 and P10, and specific brainstorming sessions in projects P1, P2 and P6.

Customers often leave it up to the MT developers to flesh out the nature of those high-level requirements based on their expertise. The task of requirement elicitation and requirements engineering in general is done by MT developers. Not only are they in charge of implementation, but also eliciting the requirements is carried out by them as well.

> "Stakeholders give high level goals and it is for you to decide how to get there and what to use" (Interview study participant).

Therefore, initially the customer provides the developers with some high-level goals. Next, developers decompose the goals into sub requirements and once they have analysed them then they meet the customers again for a confirmation. Goal decomposition via scenarios was used in projects P4 and P5. Negotiation over requirements may take place (as in projects P2, P3, P6, P8 and P10). Once there is an initial confirmed draft of the requirements of the overall system then the implementation phase is started. During the implementation, at the end of every stage developers provide prototypes or partial implementations for stakeholders to review.

> "It starts with the customer, proof of concept, then taking some code from the customer and presenting what can be done by prototyping, by a tool which provides analysis on code" (Interview study participant).

Once the prototype is delivered to the stakeholders, they can raise an issue in case something is wrong or missing, otherwise the next stage of implementation will start. Prototypes were very popular amongst almost all the model transformation projects that we analysed, as these help both developers and stakeholders to understand about the problem space. According to a participant paraphrased quote, "A good prototype is one which is a subset of the complete system". Prototypes were used in projects P1, P2, P3, P4, P5, P6, P8, P9 and P10.

5.6 RE Techniques for MT

There are many methods and techniques proposed by the requirements engineering community, however selecting an appropriate set of requirements engineering techniques for a project is a challenging issue. Most of these methods and techniques were designed for a specific purpose and none cover the entire RE process. Researchers have classified RE techniques and categorised them according to their characteristics. For instance, Hickey *et al.* [11] proposed a selection model of elicitation techniques, and Maiden *et al.* [31] defined a framework that provides requirements acquisition methods and techniques. According to our interview and SLR study results, in MT projects, RE techniques are selected and applied primarily based on personal preference or companies policy rather than on the characteristics and context of a project.

There exist several different requirements engineering techniques from a variety of sources that can be employed during MT development. Here we present some of those that were more widely used in the MT interview projects. We have categorised RE techniques into groups of *human communication, knowledge development* and *requirements documentation.*

This classification aligns with established frameworks in the RE field, reflecting essential aspects of requirements engineering. Human communication techniques ensure effective stakeholder interaction, vital for accurately capturing requirements, as highlighted by [46]. Knowledge development, through domain analysis and prototyping, enhances understanding of the problem domain, supported by [35]. Requirements documentation ensures clarity and traceability, crucial for project continuity, as emphasized by [36]. This classification is useful because it

Table 7: RE techniques in interview study MT projects

Category	***RE Technique***	***Project ID***	***Rationale***
Human communication	Online conference	P1, P2, P3, P6	Distribution of stakeholders
			Lack of accessibility
			Convenience
	Brainstorming	P1, P2, P6	Enabling both stakeholders and developers to understand each other as well as the requirements
	Joint requirements development session	P2, P10	Resolving together any issue which is not clear
Knowledge development	Categorisation	P1, P2, P3, P4, P5, P6, P7, P10	Identifying functional and non-functional requirements
	Prototypes	P1, P2, P3, P4, P5, P6, P8, P9, P10	Receiving feedback based on the prototype
			Informing the stakeholders of the progress
	Scenarios	P4, P5	To decompose the requirements
			Identify implications
	Negotiation	P2, P3, P6, P8, P10	To prioritize the requirements
			Identify trade-offs
Requirement documentation	Diagrams	P1, P2, P4, P5, P6, P7, P8, P9, P10	Providing a general view of the system
	Textual	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10	Presenting the system formally
			Providing a contract for stakeholders

addresses different dimensions of RE, allowing flexibility in technique selection based on the project's context and needs, ensuring stakeholder engagement, structured process management, thorough knowledge integration, and precise documentation.

Ultimately each transformation is unique and the process of choosing these techniques is highly dependent on the context of the transformation and stakeholder(s). In Table 7 we summarise the RE techniques that were used in the MT development projects from the interview-based study. In the first column a general *category* is defined followed by *RE techniques* and the MT *projects* in which they were applied. In the *rationale* column, the reasons for using specific techniques are described by interviewees.

5.7 MT Project Outcomes

Project outcomes concern the degree to which the project deliverables meet the actual requirements of stakeholders, and the amount of resources required for the development. To represent the degree of customer satisfaction we use the qualitative scale (None, Low, Moderate, High). For development effort we use the definitions:

- High: $\geq$ 2 person years
- Moderate: $\geq$ 0.5 person years

- Low: < 0.5 person years

Table 8 summarises the outcomes and key challenges experienced in each of the interview study projects. The specific outcomes and issues for each project were as follows.

Project 1: There were development problems and additional effort stemming from the complexity and size of the metamodels to be processed. The intent and rationale for certain UML/OCL constructs needed to be clarified, as these were not clear from the metamodels or from the existing documentation. The results of the developed transformation have been adopted by the customer, who had high satisfaction with the transformation.

Project 2: In this project also, development problems arose mainly from the nature of the transformation input data: large-scale legacy system code, and the effort needed to understand this code before regenerating a modernised version. There was generally good communication between the developers/analysts and the customers, and effective negotiation over requirements. There has been good acceptance of the re-engineering work by customers.

Project 3: The transformation language used (a Java-based syntax tree processor) was too procedural in style, which made analysis difficult, and in particular obstructed analysis of the semantic interaction between different transformations (code generators) which may be used together. The development process used was an agile 'implement first' process, with inadequate documentation. This led to high effort in reworking the translators when errors were discovered. The customer was unwilling to participate in any structured requirements engineering process, and some of their requirements were infeasible. For these reasons we categorise the development effort as High. Not all of the desired customer requirements could be achieved, so we give a value of Moderate for customer satisfaction here.

Project 4: Although the size of this transformation is quite small (about 20 rules), it has complex behaviour due to the (under-determined) interactions of the rules, which simultaneously refactor and translate Petri-Net models to state machines. This made it difficult to verify correctness properties such as confluence. Many development iterations (over 10) were needed. The resulting transformation did satisfy all the customer's functional requirements, but capacity requirements to handle large models were not fully achieved.

Project 5: This was a relatively small transformation (approximately 30 rules), but with large-scale data (100MB and larger). Thus we place it in the Medium size category. Many development iterations (over 10) were needed. The resulting transformation did not satisfy all of the efficiency requirements, and was not able to process the complete movie database data. Thus we have given a value of Moderate for customer satisfaction.

Project 6: This case study involved a complex and difficult code generation problem (for C++ multi-threaded and multi-processor code on multiple platforms). The semantic complexity of the target language and platforms caused the development effort to be significantly higher than for other code generators developed by the company. In addition, the complexity of the resulting generator has hindered its adoption, which has been limited. Thus we give a rating of Low for customer acceptance in this case.

Project 7: This case involved re-engineering of an existing code generator: extracting its requirements from its code and then writing a new improved generator to satisfy these requirements. The scale of the work was relatively low, and the main difficulties concerned extracting the requirements and identifying incorrect functionality in the existing translator. The new translator was accepted by the customer.

Project 8: This case involved the development of transformations to support railway network design. The scale of the models is relatively large, with over 200 entities and transformation rules. An agile methodology was followed throughout development, enabling rapid customer feedback, requirements prioritisation and fast responses to changing requirements. The project was a success and was accepted by the customer.

Project 9: This is a substantial MDE platform, which is the basis of the company's business. The scale is large. An evolutionary and incremental methodology is followed for its development. The project has been successful, with application of the tools in several commercial projects.

Project 10: This case involved the generation of middleware code from DSLs. The transformations are written partly in Java and partly in a custom MT language. It is of moderate

Table 8: Outcomes of interview study MT projects

Project	*Transformation Scale*	*Development Effort*	*Stakeholders Satisfaction*	*Key Challenges*
P1	Large	Moderate	High	Complexity of metamodels
P2	Large	High: large scale legacy code processing	High	Effort needed to understand legacy code
P3	Large	High: specifications too procedural, hard to analyze or modularize	Moderate	Poor communication, inadequate RE process, high rework effort
P4	Small	Low	High	Complex transformation semantics
P5	Medium	Low	Moderate	MT limitations in processing large data
P6	Large	High: complex and detailed semantics	Low	Lack of end-user involvement, immature MT tools
P7	Medium	Low	High	Reverse engineering issues
P8	Large	Moderate	High	Large-scale models
P9	Large	High: complex functionality	High	Needed to consider wide range of use cases and prospective customers
P10	Medium	High: complex functionality	High	Hybrid 3GL/MT transformations

scale, although substantial resources were used. The project was a success from the standpoint of customer satisfaction.

Overall, we can conclude that development problems were encountered due to the scale of the metamodels (projects P1, P6) or of the models/programs (projects P2, P5) involved in the transformations. The complexity of metamodels (project P6) and the need to manage multiple versions of metamodels (project P3) also caused problems. One conclusion that can be provisionally drawn is that large-scale transformation developments with relatively low levels of application of requirements engineering tend to have poor outcomes (projects P3, P6). In contrast, a more detailed RE process can help to counteract the impact of scale (projects P2, P8).

Challenges included technical problems with the use of MDE and model transformations (P1, P4, P5, P6), problems related to the RE process, specifically lack of stakeholder involvement (P3, P6), and problems relating to the difficulty of reverse engineering (P2 and P7).

5.8 MT Project Failures (Interview Study Cases)

As with any software project, an MT development may fully succeed, partially succeed, or fail. We consider the success and failure to consist of two aspects: (i) the resources and time consumed by the development; (ii) the level of satisfaction of stakeholders in the delivered system. A project is a complete failure if it is terminated before delivery, or if the stakeholders are not satisfied at all by the delivered system. A project is a partial failure if either it consumed excessive development effort to complete, or if there is moderate or low satisfaction in the delivered system.

From the interview study, we identified three cases of (partial) project failures. In the terms of [30], these were a *process failure* in the case of project P3, where the process cost was excessive; an *interaction failure* in the case of project P6, with relatively low uptake of the system, and an

expectation failure in the case of project P5, with the capacity of the developed transformation being inadequate to meet expectations.

The causes of these failures correlate with those described in [30]:

- Project P3: The cost of the process was due in part to poor communication with stakeholders, and to the lack of a systematic process.
- Project P6: The poor uptake seemed in part due to poor communication with the end users of the transformation and lack of knowledge of what they needed or would use.
- Project P5: The failure was in part due to lack of understanding of the IMDb (inadequate requirements elicitation), and also due to technical inadequacy of the selected MT technology to process big data.

Poor communication with stakeholders is a potential problem in many MT developments due to the fact that these developments are often internal projects embedded within larger MDE projects. The MT developers receive requirements from the MDE team, but these may contain incorrect or incomplete understanding of the complete system requirements. Techniques such as joint application design (JAD) workshops are recommended by [30] to enable active participation of end users and other stakeholders in the RE process. Similar issues have also been reported with the use of domain-specific languages (DSLs), which are often also developed in internal project contexts and tend to also have deficiencies related to lack of stakeholder involvement in their construction [19], [48].

Technological deficiencies and problems are also a factor in MT project failure: due to the relative immaturity and non-standardised nature of MT languages and tools, it may be infeasible to solve a problem (such as the IMDb case) with MT technologies, or specifications/implementations in MT languages may be difficult to manage and maintain (project P3). It may be difficult or impossible to express particular requirements using a MT language. There is a lack of use of MT *specification* languages such as ATOM [43] or $\mathcal{MT}$ [20], instead MT developers usually go directly to an implementation in a particular MT language such as ATL, QVTo, ETL, etc [9]. ATOM and ATL both define model transformations by mapping input patterns to output patterns. However, ATOM is a declarative visual language designed for formal specification and verification, while ATL is a textual language with procedural semantics, widely adopted for practical implementation. The validation and verification stage of the RE process is particularly affected by the lack of MT specification languages and tools for MT analysis and verification. Due to the immaturity of the MT field, experimental transformation techniques may also be used in production projects (project P6), resulting in high development effort.

5.9 Summary

By examining the real-world experiences of MT practitioners in substantial MT development projects, we derived the following outline RE framework for MT projects:

- The context of the transformation needs to be taken into account, i.e., whether it is a standalone MT development, or an internal project within a wider project (Section 5.1). In the latter case, the roles of stakeholders for the MT project need to be carefully considered, according to the adapted onion model of Figure 2, and sufficient access to stakeholders should be supported (Section 5.2). This is particularly important for agile developments relying upon good communication with customer representatives (Section 5.3).
- Challenges with regard to requirements changes and conflicts arise in MT projects, as in general software development projects, and need to be managed (Section 5.4). A particular problem is that MT languages and tools are quite specialised and customers may rely upon the MT experts to translate general requirements into detailed requirements suitable for MT implementation (Section 5.5).

- A range of relevant RE techniques can be used for MT developments, but as yet there is no systematic selection procedure to guide the adoption of particular techniques. In Section 5.6 we classify some RE techniques which have been found to be useful in practice. A thorough RE process would probably use at least one technique from each category (projects P1, P2, P3, P6, P10 satisfy this criteria).
- Challenges affecting project outcomes and possibly leading to project failures include stakeholder communication issues, inadequate process management, and technical problems with MT technologies – especially scalability and maturity issues (Sections 5.7, 5.8).

6 Structured Literature Review (SLR)

To investigate further the issue of RE in MT, and to provide an alternative source of information, we carried out a literature review of published MT case studies covering the last twenty years (2004–2023). This provides a broader range of information on the prevalence and application of RE in MT developments, compared to the interview study. We provide a complete replication package to enable the easy reproduction of our systematic review. The prepared package that has been made publicly available[1] includes a spreadsheet containing a list of filtered articles, the classification of primary studies and extracted data.

6.1 Research Question for SLR

The main question which we consider for the SLR is Q1:

> "What requirements engineering processes and techniques, if any, have been applied in model transformation development?".

The purpose of this question is to investigate the recent and current role of RE in MT. We aim to collect the current available knowledge regarding MT developments and the role of RE in these developments. It will also be used to identify any potential gaps in research and practice, and guide the proposal of solutions and suggestions for further investigations and future work. We have defined the structure of the SLR according to the PICOC criteria [29]:

- Population: Research papers presenting MT developments and case studies.
- Intervention: RE process and techniques used within MT developments.
- Comparison: Analysis of the current state of RE in MT.
- Outcome: Identification of issues that need addressing in RE for MT.
- Context: MT engineering and development.

6.2 Source Selection

The selection process of relevant and appropriate papers was done via two different paths. An automatic searching method was used based on the main sources of scientific paper databases such as: IEEE Xplore, ACM Digital Library, Science Direct, Google Scholar, Wiley Online Library, and Springer Link were investigated. Moreover a manual search method was also used in the following representative journals and conferences such as: Transformation Tool Contest (TTC), Model Driven Engineering Languages and Systems (MODELS), International Conference on Model Transformation (ICMT), European Conference on Modelling Foundations and Applications (ECMFA), International Journal on Software and Systems Modeling (SoSyM). One of the main advantages of applying the manual search was due to the fact that it allows us to carry out a more in-depth study of some particular works based on specific topics and areas;

[1] `https://tinyurl.com/bdfecw7t`

Table 9: Search terms for selecting relevant research studies

Group	Term
A	Requirements engineering
	Requirements engineering technique
B	Model Driven Architecture (MDA)
	Model Driven Engineering (MDE)
	Model Driven
	Model Based
	Model

also it can be used as a verification method in order to verify the correctness of the automatic search method. In accordance with the objectives of this study, two search terms were selected. Each term has its own set of keywords, and every selected paper must include at least one of these keywords. Table 9 presents the list of selected search terms in this research.

The following search string is defined to identify the largest number of studies in this domain.

$$\text{Search String} = \text{“Transformation”} \wedge (\text{A} \wedge \text{B})$$

We validated the search string using trial searches on Google Scholar to determine if it discovered certain expected papers.

Backwards and forwards snowballing was applied to the initial selection of papers in order to achieve completeness in selection.

6.3 Inclusion and Exclusion Criteria

Inclusion and exclusion criteria are considered in this survey for appropriate selection of primary studies.

- IC1: Papers published between January 2004 and June 2023
- IC2: Publications that describe the techniques/processes for specifying requirements of MT
- IC3: Publications in peer-reviewed journals, conferences, and workshops
- IC4: Publication in English
- EC1: Publications not written in English
- EC2: Publications before 2004
- EC3: Summary, survey, or review publications
- EC4: Non peer-reviewed publications
- EC5: Publications not focusing on MT
- EC6: Books, web sites, PhD theses, pamphlets, tutorials, duplicate papers, and white papers.

In this research, we assessed the abstracts, titles, and keywords of papers based on our inclusion and exclusion criteria. Additionally, in cases where the relevance of a paper was unclear, we conducted a comprehensive examination of the entire paper.

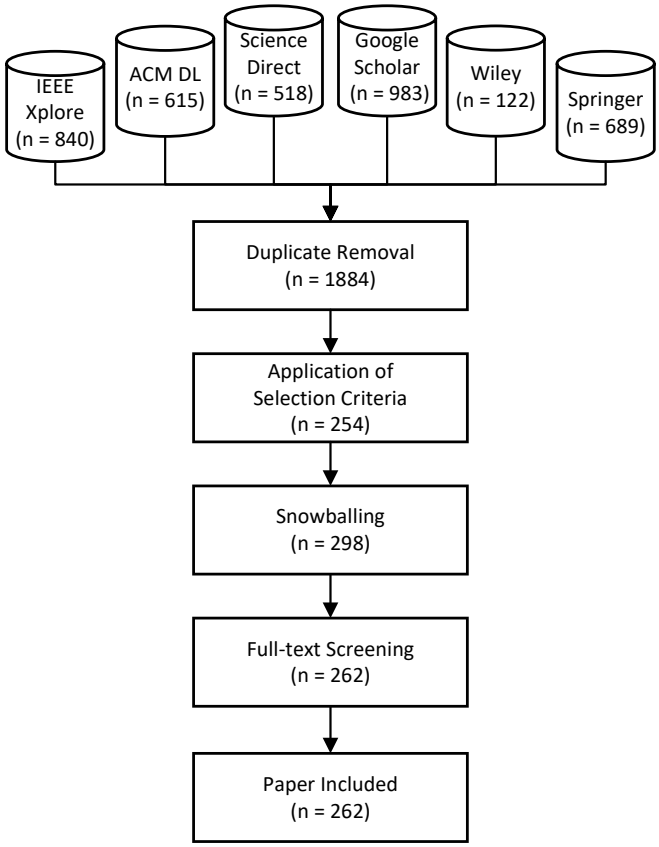

Figure 3: Overview of search and selection process

6.4 Applying the SLR Process

We conducted our study selection process iteratively to gather relevant primary studies from the identified databases. This process involved an initial automatic database search and a subsequent iterative snowballing-based search. We executed the predefined search queries in the selection process for each repository, with the aim of identifying relevant articles from the chosen data sources. The initial pool of studies, comprising 1884 articles, was generated after collecting all the relevant papers and eliminating duplicates. To further enhance the precision of article selections, we applied inclusion and exclusion criteria. This activity resulted in the identification of 254 articles that met all inclusion criteria and did not violate any exclusion criteria. Snowballing-based search: To guarantee thorough coverage of the study's extensive scope, in the ultimate phase of the search process, we employed backward snowballing to examine the references of the remaining articles and forward snowballing to search for articles that cited the existing ones, thus broadening our search horizon. By implementing this process, 44 additional articles were identified. We reviewed the full texts of these articles and arrived at a total of 298 of relevant articles selected in accordance with the predefined inclusion and exclusion criteria. After a manual screening process that takes into account the title, abstract, introduction, and conclusion, our final pool consisted of 262 papers. Moreover, through checking the quality of all the papers the same number of papers are preserved (262). Figure 3 outlines the process applied in this research.

6.5 Information Extraction

We extracted the data and information from the selected SLR papers according to our research question Q1.

The research question is broken down further into the following specific criteria, based on the framework elements of Section 4 and Section 5:

1. *Category of transformation (mirrors Section 4.1)*: Transformations can be classified as: refactorings, migrations, refinements, code generations (model-to-text), semantic mappings, bx (bidirectional), text-to-model, data analysis, reactive (defining the response to events), abstractions and other reverse-engineering transformations.

2. *Transformation development process (aligns with Section 5.3)*: We consider the overall methodology of the MT projects, e.g., if an agile or MDE process is followed.

3. *Type of projects (corresponds to Section 4.2)*: The classification of the MT software development project into greenfield vs brownfield, customer vs market driven, academic versus industrial, in-house vs outsourced, single product vs product line, as defined in Section 4.2.

4. *Stakeholders (matches Section 5.2)*: We are interested to identify the type of stakeholders in MT development projects for this SLR, such as end users or software developers, and how the MT project developers engaged with stakeholders.

5. *Requirements engineering techniques and process (reflects Section 5.6)*: We identify which RE techniques are used in cases, and whether a systematic and well-organised process is followed for RE.

6. *Requirements category and expression (ties to Section 5.5)*: We are interested in the categories of requirements (functional, non-functional, local, global) which are considered in MT requirements engineering, and the degree to which requirements are explicitly expressed.

7. *MT project outcomes (aligns with Section 5.7)*: The degree to which requirements were achieved, and the degree to which stakeholder expectations were satisfied.

6.5.1 Category of Transformation

Considering the SLR cases, the result for the transformation category (criterion 1) shown in Figure 4 is that about 28.24% of the SLR transformation case studies are refinements, 25.19% are migrations, 19.47% are refactorings, and 18.32% are code generations. This distribution reveals that no single transformation type dominates the field, indicating balanced research attention across different MT applications.

6.5.2 Transformation Development Process

The result for the process of the transformation development project (criterion 2) in the SLR cases was not very explicit as not many developers explained the scale of the transformation and the time and effort that they consumed regarding the development in the analysed papers. Where size information was available, there was a predominance of small cases (66%) or medium-sized cases (28%). The cases were predominately academic (92%), meaning that most publications were written by university-affiliated authors. However, 95% of the projects were customer-driven in the sense that they aimed to solve a practical or externally motivated problem (e.g., code generation for a specific DSL or system). Importantly, in many of these academic cases, the customer or stakeholder role was either internal to the research team or inferred rather than directly involved. Regarding the development approach used (Figure 5), 61% of projects had no defined development process, 8% used an iterative/incremental approach, 13% used an MDE methodology for MT development, and 3% used formal methods.

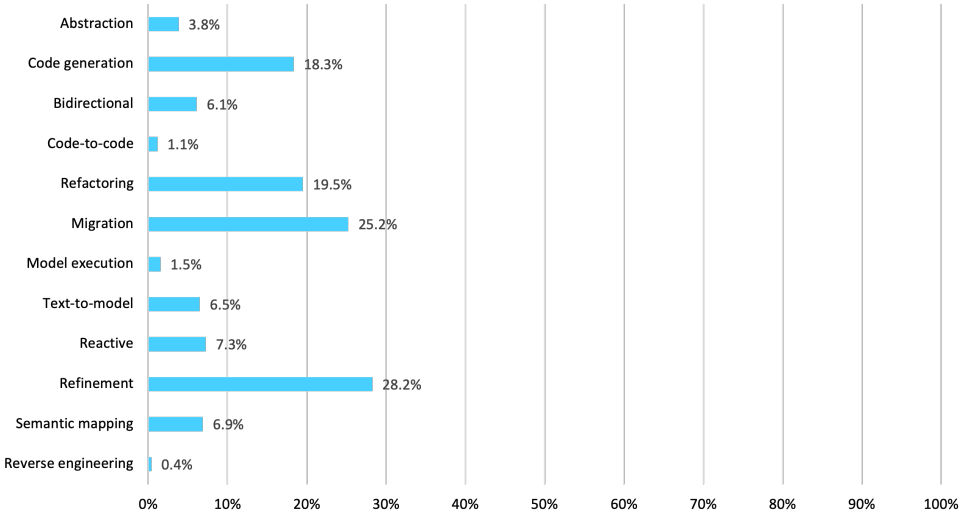

Figure 4: MT categories of SLR MT cases

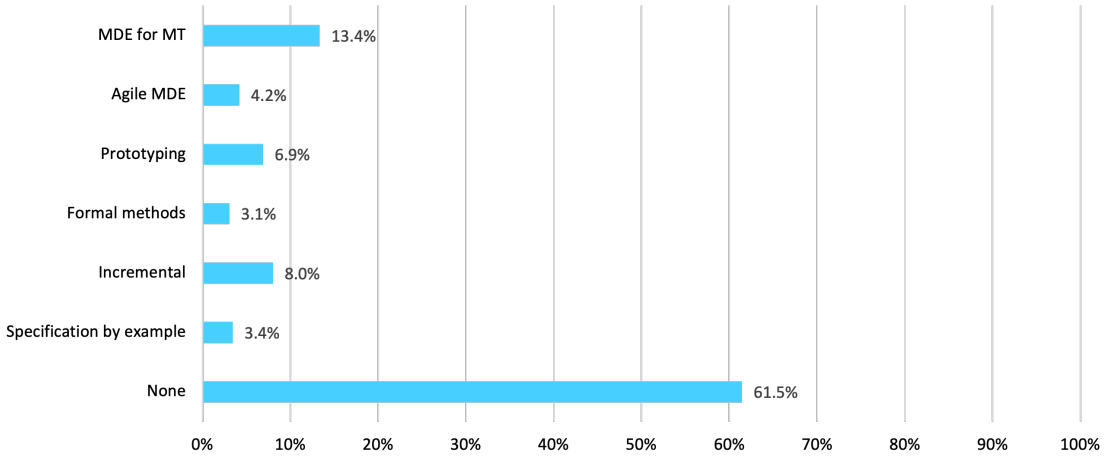

Figure 5: Software development methodologies in SLR MT projects

Table 10: MT development types in SLR cases

Case	Brown-field	Green-field	Customer-driven	Market-driven	In-house	Out-sourced	Single-product	Product-line	Academic	Industrial
Count	11	251	250	12	254	8	257	5	241	21
Percentage	4.20%	95.80%	95.42%	4.58%	96.95%	3.05%	98.09%	1.91%	91.98%	8.02%

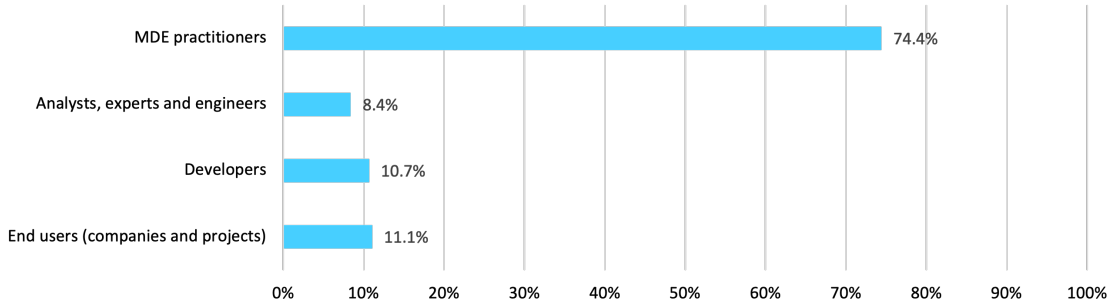

Figure 6: SLR cases: Stakeholders

6.5.3 Type of Projects

The results for the type of development project (criterion 3) in the SLR cases show that 95.80% of the MT development cases were greenfield, while only 4.20% were refinements or extensions of existing transformations. About 95.42% of the cases were customer-driven and proposed solutions for a particular type of client, while 4.58% of the cases targeted a wider range of clients by considering the different possible users who could benefit from the software development. Most of the cases (96.95%) carried out all the phases of the MT development lifecycle within a single team, whereas 8 cases (3.05%) followed coordination among teams from different organisations. Nearly all MT development cases (98.09%) developed a single product version, while the remaining 1.91% released two or three versions. Furthermore, about 91.98% of the MT development cases were academic, and only 8.02% were industrial. Table 10 summarises this classification of the MT development types in our SLR.

6.5.4 Stakeholders

Regarding the stakeholders (criterion 4) of the SLR cases, the main stakeholders of the analysed papers consisted of: 74.43% MDE practitioners, 8.4% domain experts and analysts, 10.69% system developers, and 11.07% end users (Figure 6). The high proportion of MDE practitioners as stakeholders validates the model we proposed in Figure 2 for the stakeholders of MT projects.

About 44.66% of the SLR cases used online forums and email as a means of communication between the developer(s) and the stakeholder(s). In many cases (40.08%) there seemed to be no attempt made to reach stakeholders, instead the transformation developers made their own assumptions about the needs of the stakeholders. Direct stakeholder participation in development or direct stakeholder consultation took place in only 6.87% of cases. Figure 7 illustrates the methods employed to interact with the stakeholders.

6.5.5 Requirements Engineering Techniques and Process

Figure 8 depicts the techniques utilized to determine the requirements, revealing a heavy reliance on problem descriptions and statements (68.32% and 61.83%) compared to other methods. This suggests most studies adopted a top-down approach where requirements were derived from predefined problem definitions rather than through iterative discovery or stakeholder collaboration.

For the SLR study cases, analysis of the application of RE processes and techniques (criterion 5) throughout the development showed that 69.85% of cases applied some requirements

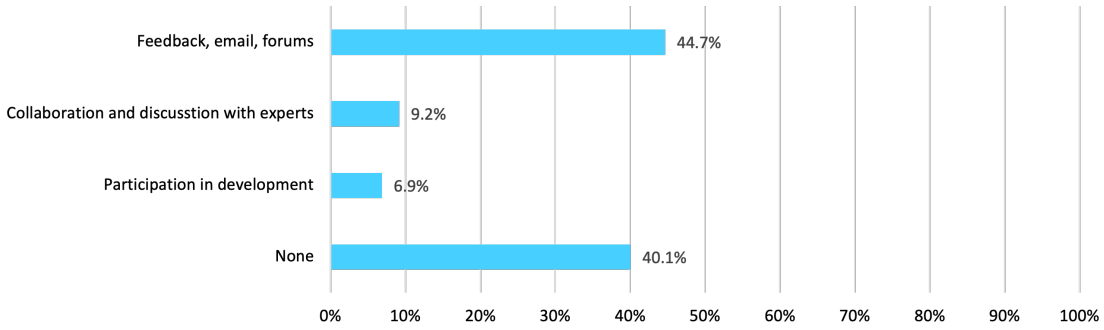

Figure 7: SLR cases: Interaction with stakeholders

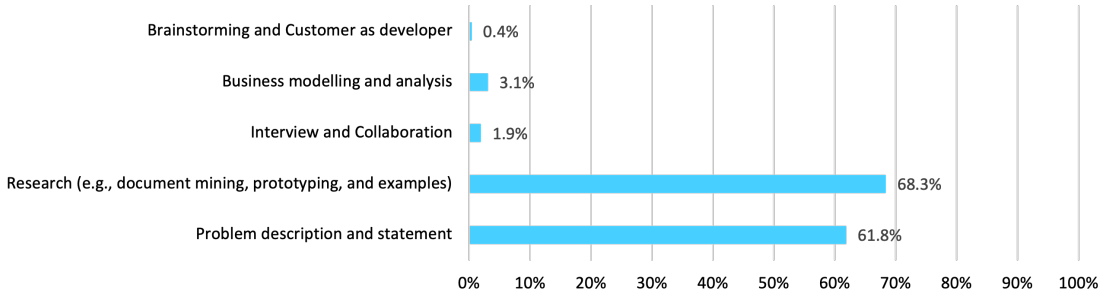

Figure 8: SLR cases: Finding out what was required

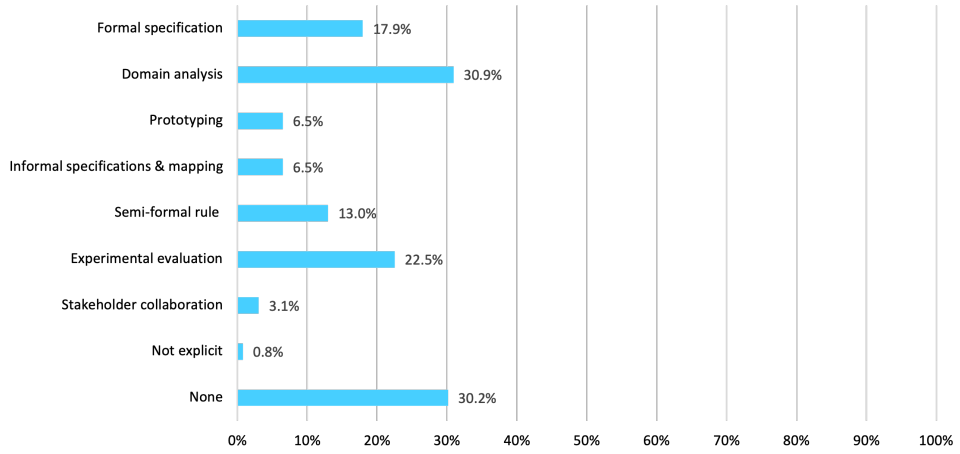

Figure 9: Requirements engineering techniques in SLR MT projects

engineering technique during the development life cycle such as: Domain analysis (30.92%), semi-formal or formal rule specifications (not in an MT language) 30.92%, prototyping 6.49%, experimental evaluation (interviews, questionnaire, observation, survey) 22.52%. Information about the RE technique used by 30.15% of the the cases was not available, according to their presentation of the cases (Figure 9). Around 52% used UML class diagrams for documentation, 11% used no diagrams, and 5.73% used graphs (Figure 10).

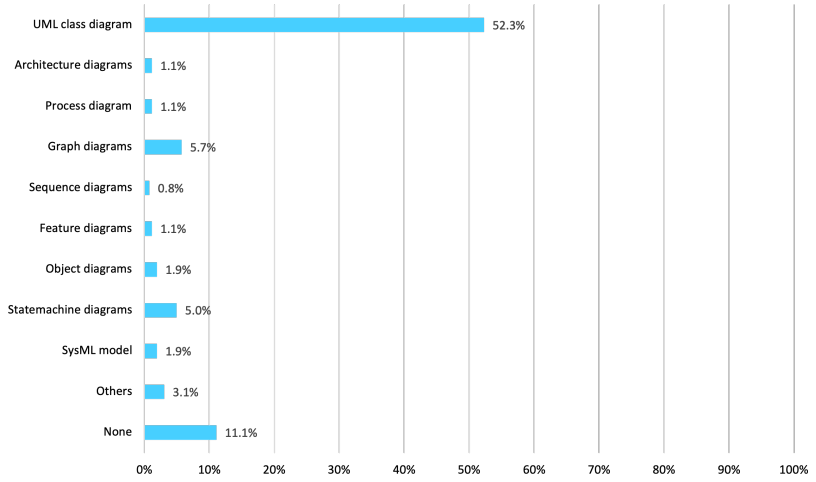

Figure 10: Diagrams in SLR MT projects

6.5.6 Requirements Category and Expression

In the SLR we endeavoured to find out what types of requirements arose most often in MT development, and how these differed from the requirements of other (non-MT) applications (criterion 6).

Figures 11, 12 and 13 show the classifications of the requirements for the SLR MT cases. In these cases, requirements could generally be classified as either *functional* and *non-functional*. In the SLR, in contrast to the interview study cases, we found that functional MT requirements were often divided into *local* and *global* functional requirements: local requirements express how individual model elements or small groups of related model elements should be mapped to elements of another model, or should be refactored in-place. In the case of bidirectional transformations (bx), local correspondence requirements express how elements of one model should correspond to elements of the other. Global requirements concern properties of source or target models considered as a whole. Syntactic correctness is the property that the transformation produces models which conform to the constraints of the target language. Semantic correctness expresses that necessary semantic properties of the target model relative to the source model are satisfied. Semantic preservation states that the semantics of the source model are preserved in the target. Completeness states that all cases that may occur in the source model can be processed by the transformation.

We found that the most common types of transformation requirement explicitly considered in the SLR cases are: Local mapping (39.31% of SLR cases), Efficiency (34.35% of cases), Semantic correctness (28.63%), Syntactic correctness (25.57%) and Completeness (18.32%).

6.5.7 MT Project Outcomes

This section discusses the outcomes of the SLR cases (criterion 7). Figure 14 shows the degree to which the MT case requirements were achieved by the implementation: for 27.1% of cases the achievement appears to be low, based on the information reported in the publication. Since negative results are less likely to be reported, this percentage is likely to be higher in actual practice. Unfortunately, reasons for project failure were not consistently reported. The most likely outcome is moderate success (53.44% of cases), with only 15.65% highly successful. Regarding stakeholders satisfaction with the results (Figure 15), there is only definite information in 24.81% of the surveyed cases. Of these, 1.53% report low satisfaction, 18.7% moderate satisfaction and 4.20% high satisfaction. Overall, the picture obtained from the SLR cases shows incomplete achievement of requirements and only moderate success in achieving stakeholder satisfaction. These results may derive in part from the continuing technological problems and

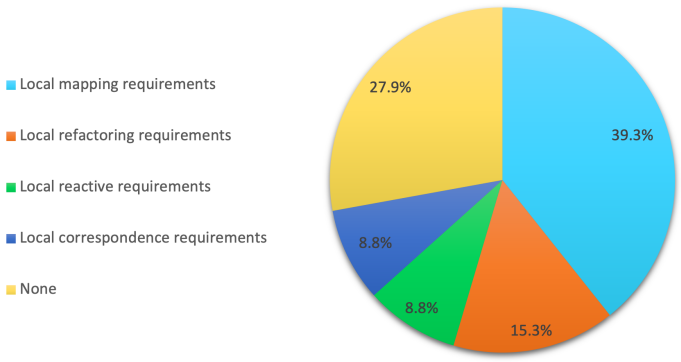

Figure 11: SLR MT project local functional requirements

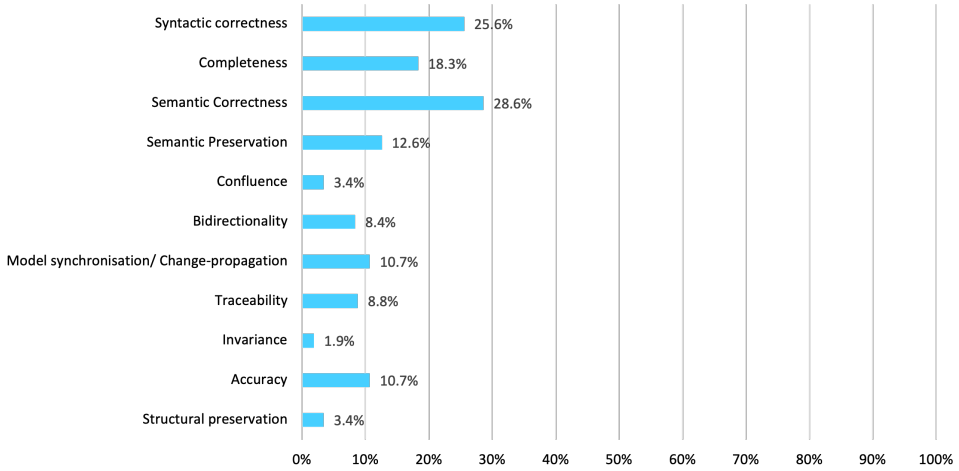

Figure 12: SLR MT project global functional requirements

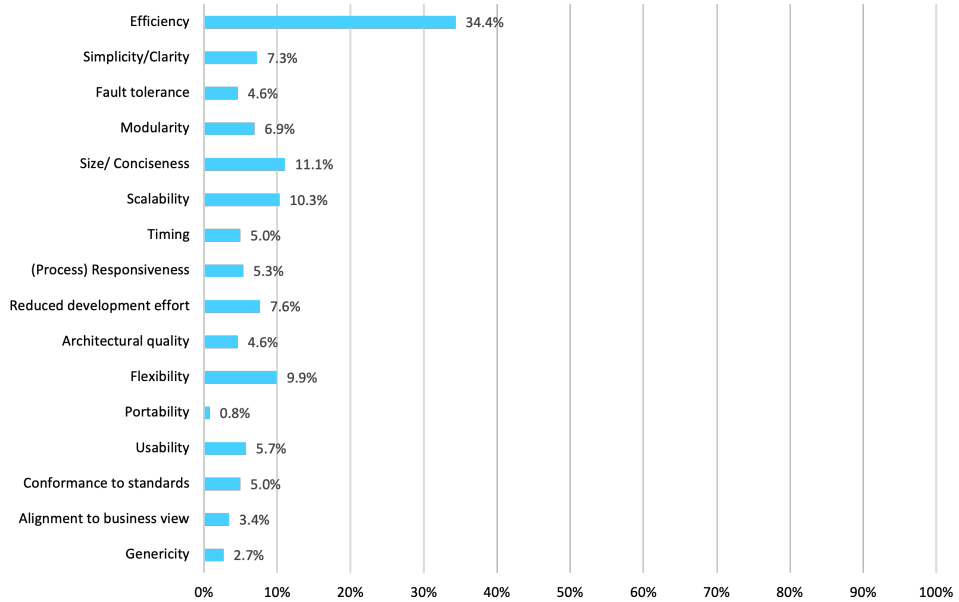

Figure 13: SLR MT project non-functional requirements

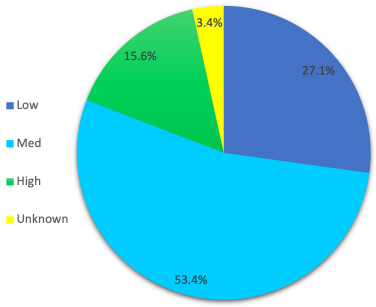

Figure 14: Requirements achievement in SLR projects

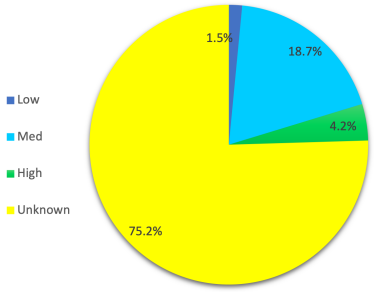

Figure 15: Stakeholder's satisfaction in SLR projects

deficiencies that have been identified in MT languages and tools [5], [12].

7 Comparison of Findings: Interview Study and SLR

This paper has utilized two complementary research methods: semi-structured interviews with model transformation (MT) practitioners and a systematic literature review (SLR) analyzing published MT case studies. While both approaches investigate the role of requirements engineering (RE) in MT development, they offer distinct perspectives that contribute to a more comprehensive understanding of the field.

Findings from the interview study and SLR exhibit strong alignment in key areas. Both sources indicate that RE practices in MT development are informal and lack structured methodologies. Additionally, stakeholder engagement challenges are evident in both studies. There was a wider range of transformation categories in the SLR compared to the interview cases, and a more even distribution of categories. Interview participants frequently cited limited access to stakeholders, a constraint also reflected in the SLR, where many studies inferred transformation requirements without direct consultation with stakeholders. Furthermore, both studies highlight a primary focus on implementation rather than formal RE processes, with most MT projects prioritizing coding and transformation development. The studies also indicate a preference for agile or informal requirements gathering techniques, particularly prototyping and iterative refinements, which aligns with SLR findings that example-driven specifications are more commonly used than formal requirement definitions.

Despite these similarities, several notable differences are identified. A key distinction lies in the populations covered by the two studies, rather than in their research aims. Both the interview study and the SLR sought to investigate RE practices in MT development, but while the interviews focused on industrial practitioners, the SLR predominantly identified academic case studies, which is a reflection of what is available in the published literature, rather than a deliberate focus. This reflects a gap in how RE is applied in practice versus how it is often represented in academic studies. While many of these are customer-driven in motivation, the customers are often idealized or abstracted, that is, they are not real-world stakeholders engaged in the RE process, but rather hypothetical roles or proxy users defined by the researchers. Figure 6 shows that MDE practitioners are the primary stakeholders, which typically refers to transformation developers or researchers who implement and evaluate the MT, not necessarily external users or paying customers. This highlights the gap between academic design assumptions and industrial stakeholder dynamics. Another divergence is observed in stakeholder engagement. In the interview study, stakeholders are primarily MT developers and project managers embedded within larger model-driven engineering (MDE) teams, whereas in the SLR, many studies assume idealized or hypothetical stakeholders, often overlooking practical constraints. Additionally, differences arise in requirements categorization. Findings from the interview study indicate that

MT developers frequently deal with implicit, evolving, or unstated requirements, whereas the SLR cases categorize requirements explicitly into functional, non-functional, local and global classifications.

The comparison highlights several key areas that require further investigation. First, there is a clear need to bridge the research-practice divide. Although academic studies are often expected to include well-defined requirements, our findings show that many published MT projects lack evidence of systematic RE techniques or software development methodologies (see Figures 5 and 9). This suggests a gap between academic expectations and actual practice, even in scholarly case studies. Second, the findings emphasize the necessity for standardized RE practices in MT. The absence of formalized RE methodologies in industry suggests that practical, lightweight RE frameworks tailored to MT development should be explored. Lastly, enhancing stakeholder collaboration remains a critical challenge. Adopting structured approaches, such as joint application development (JAD) workshops or stakeholder-driven requirement elicitation techniques, could address communication barriers and improve the RE process in MT development. The integration of insights from both research methods is essential to developing a robust, structured RE framework that aligns with both academic research and industrial practice in MT development.

8 Threats to Validity

This section discusses potential threats that might affect the validity of our interview study and SLR.

Threats to the validity of conclusions drawn from the interviews include: (i) the possibility that the interviewees and examined cases are not representative of transformation developers and projects; (ii) the risk that interviewees selected unrepresentative projects; (iii) the potential for interview questions to elicit a particular response; (iv) the chance that interviewees presented inaccurate information, particularly by exaggerating project success; and (v) potential response bias, where participants may provide socially desirable answers rather than fully objective responses.

To mitigate (i), some literature review was conducted to identify experts with relevant experience in model transformation development. This ensured representation across diverse domains such as embedded systems, finance, re-engineering, transport, defense, and business. Of 15 candidates approached, 7 agreed to participate, covering a wide range of transformation projects. Regarding (ii) and (iv), the study includes both successful and problematic projects, such as projects 3 and 6, to provide a balanced assessment. To address (iii), the interview questionnaire and methodology were reviewed and approved by an expert committee for ethical compliance.

To mitigate (v) response bias, participants were informed that their responses would be anonymized and used solely for research purposes. Additionally, open-ended questions were employed to allow participants to express their experiences without leading prompts.

A further consideration is that projects 4 and 5 were carried out by one of the authors, introducing a potential risk of bias. To reduce this risk, responses from these projects were reviewed independently by multiple researchers to ensure objectivity, and their findings were not disproportionately weighted in deriving overall conclusions. While these measures help mitigate bias, some subjectivity may remain, and future studies could incorporate external validation.

For the SLR study, a limitation is that case study papers may not fully describe the RE processes applied during transformation development. In many cases, only high-level information is provided about transformation details, making it necessary to assume that if RE or development processes were not mentioned, they were likely not used. Additionally, factors such as publication bias, data extraction errors, and misunderstandings may affect validity [16]. To mitigate this, credible digital libraries were used for source selection, and a double-review process was applied, where each selected paper was independently assessed by two researchers to minimize inaccuracies. Additionally, while the study focused primarily on peer-reviewed literature in the SLR, the exclusion of grey literature is a limitation. Grey literature, such as industry reports and unpublished case studies, may provide additional insights into RE practices in MT development.

However, it was excluded due to concerns regarding limited accessibility, and potential bias toward successful projects. Future research should explore methods to systematically incorporate grey literature while ensuring rigor and reliability.

9 Conclusions

This study explored the role of requirements engineering (RE) in model transformation (MT) development through two complementary research methods: a semi-structured interview study with industry practitioners and a systematic literature review (SLR) of published transformation cases. While conducted independently, the interview study identified key challenges in industrial RE practices, which informed the focus areas of the SLR.

Findings from both studies highlight three major aspects of RE in MT development: (1) the informal nature of RE practices and the predominant focus on implementation over structured RE processes, (2) challenges in stakeholder engagement and limited access to domain experts, and (3) the difference between research and industrial practice in how RE is applied. Most MT projects are developed in greenfield settings, yet they are often embedded within larger model-driven engineering (MDE) projects, which affects stakeholder identification and communication. The difference between research and practice may arise from MT being a relatively young technology in industrial practice and its highly variable application domains.

One of the most significant challenges in RE for MT development is the lack of a systematic RE process. Although some RE techniques—such as prototyping, scenario analysis, and example-based generalization—are used, they are often applied in an ad-hoc manner based on personal experience rather than a well-defined methodology. This lack of structure is further compounded by restricted access to stakeholders, limiting the applicability of traditional RE techniques such as interviews and brainstorming. Additionally, the absence of clear guidelines for adapting RE techniques to MT-specific needs hinders practitioners from effectively managing transformation requirements.

Despite similarities in the issues identified by both studies, key differences emerge between them. The interview study reflects real-world industrial constraints, including evolving requirements and time pressures, whereas the SLR presents a more structured, research-driven perspective, often assuming well-defined requirements. Additionally, while industrial practitioners frequently work with implicit and evolving requirements, expressed from a stakeholder perspective, the SLR cases predominantly use fixed sets of requirements, categorized into functional, non-functional, local and global classifications, with a more technical orientation. This contrast highlights a fundamental gap between academic research and industrial needs.

The study identified four key obstacles to effective RE in MT development:

- Restricted access to stakeholders, which limits the applicability of structured RE techniques.
- The lack of standardized RE methods tailored to MT, leading to ad-hoc practices.
- An absence of systematic processes for capturing and managing transformation requirements.
- A gap between research and industry, where structured RE techniques proposed in academia are not readily adopted in practice.

These findings emphasize the need for lightweight, practical RE frameworks that balance the flexibility required in industry with the structure provided in academic methodologies. The results also highlight the importance of strategies to manage requirements changes and conflicts, as well as a better understanding of the unique nature of MT requirements, including both local and global functional and non-functional aspects.

To address potential response bias in interviews, several measures were taken. Participants were informed that their responses would be anonymized to encourage honesty, and open-ended

questions were used to minimize leading prompts. Furthermore, the study included practitioners from diverse domains to reduce the risk of a single perspective dominating the findings.

Future work should focus on bridging the gap between academic and industrial practice by integrating agile and formal RE techniques in a way that is both practical and adaptable to industrial constraints. In parallel, we have started work on defining a more systematic process for requirements engineering in the context of MT development [49]. By combining insights from both the interview study and SLR, this research lays the groundwork for refining RE methodologies in MT development, ultimately fostering better alignment between academic advancements and industrial practices.

References

[1] Ian F. Alexander. A taxonomy of stakeholders: Human roles in system development. *International Journal of Technology and Human Interaction (IJTHI)*, 1(1):23–59, 2005.

[2] H. Alfraihi and K. Lano. Trends and insights into the use of model-driven engineering: a survey. In *SAM/MODELS*, 2023.

[3] E. Batot, H. Sahraoui, E. Syriani, P. Molins, and W. Sbuoi. Systematic mapping study of model transformations for concrete problems. In *4th International Conference on Model-Driven Engineering and Software Development, 176-183, 2016, Rome, Italy*, 2016.

[4] J. M. Bryson. What to do when stakeholders matter: Stakeholder identification and analysis techniques. *Public Management Review*, 6(1):21–53, 2004.

[5] Loli Burgueño, Jordi Cabot, and Sébastien Gérard. The future of model transformation languages: An open community discussion. *Journal of Object Technology*, 18(3), 2019.

[6] Fabian Buttner, Marina Egea, Esther Guerra, and Juan De Lara. Checking model transformation refinement. In *International Conference on Theory and Practice of Model Transformations*, pages 158–173. Springer, 2013.

[7] Alcino Cunha, Ana Garis, and Daniel Riesco. Translating between Alloy specifications and UML class diagrams annotated with OCL. *Software & Systems Modeling*, 14(1):5–25, 2015.

[8] S. Greiner, T. Buchmann, and B. Westfechtel. Bidirectional transformations with QVT-R: a case study in round-trip engineering UML class models and Java source code. In *Modelsward 2016*, pages 15–27. INSTICC, SCITEPRESS, 2016.

[9] Esther Guerra, Juan de Lara, Dimitrios S Kolovos, Richard F Paige, and Osmar Marchi dos Santos. transML: a family of languages to model model transformations. In *Model Driven Engineering Languages and Systems*, pages 106–120. Springer, 2010.

[10] Zef Hemel, Lennart CL Kats, Danny M Groenewegen, and Eelco Visser. Code generation by model transformation: a case study in transformation modularity. *Software and Systems Modeling*, 9(3):375–402, 2010.

[11] Ann M Hickey and Alan M Davis. Requirements elicitation and elicitation technique selection: model for two knowledge-intensive software development processes. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, pages 10–pp. IEEE, 2003.

[12] S. Hoppner, Y. Haas, M. Tichy, and K. Juhnke. Advantages and disadvantages of (dedicated) model transformation languages. *Empirical Software Engineering*, 27, 2022.

[13] Tassilo Horn, Christian Krause, and Matthias Tichy. The TTC 2014 movie database case. *TTC 2014*, page 93, 2014.

[14] John Hutchinson, Jon Whittle, Mark Rouncefield, and Steinar Kristoffersen. Empirical assessment of MDE in industry. In *Proceedings of the 33rd International Conference on Software Engineering*, pages 471–480. ACM, 2011.

[15] F. Jouault, F. Allilaire, J. Bezivin, and I. Kurtev. ATL: a model transformation tool. *Science of Computer Programming*, 72(1):31–39, 2008.

[16] Barbara Kitchenham. Procedures for performing systematic reviews. In *Keele University technical report*, 2004.

[17] Shekoufeh Kolahdouz-Rahimi, Kevin Lano, Mohammadreza Sharbaf, Meysam Karimi, and Hessa Alfraihi. A comparison of quality flaws and technical debt in model transformation specifications. *Journal of Systems and Software*, 169, 2020.

[18] D. Kolovos, R. Paige, and F. Polack. The Epsilon transformation language. In *ICMT*, 2008.

[19] T. Kosar, S. Bohra, and M. Mernik. Domain-specific languages: A systematic mapping study. *IST*, 71:77–91, 2016.

[20] K. Lano, S. Fang, and S. Kolahdouz-Rahimi. Tl: an abstract specification language for bidirectional transformations. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, MODELS '20, New York, NY, USA, 2020. Association for Computing Machinery.

[21] K. Lano, S. Fang, M. Umar, and S. Yassipour-Tehrani. Enhancing model transformation synthesis using natural language processing. In *MDE Intelligence workshop, MODELS 2020*, 2020.

[22] K. Lano, S. Kolahdouz-Rahimi, and S. Fang. Model Transformation Development using Automated Requirements Analysis, Metamodel Matching and Transformation By-Example. *ACM TOSEM*, 31(2):1–71, 2021.

[23] K. Lano and H. Siala. Using MDE to automate software language translation. *Automated Software Engineering*, 31, 2024.

[24] K. Lano, S. Yassipour-Tehrani, H. Alfraihi, and S. Kolahdouz-Rahimi. Translating from UML-RSDS OCL to ANSI C. In *OCL 2017, STAF 2017*, pages 317–330, 2017.

[25] Kevin Lano and S Kolahdouz-Rahimi. Constraint-based specification of model transformations. *Journal of Systems and Software*, 86(2):412–436, 2013.

[26] Kevin Lano and Shekoufeh Kolahdouz-Rahimi. Model-driven development of model transformations. In Jordi Cabot and Eelco Visser, editors, *Theory and Practice of Model Transformations*, pages 47–61, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[27] Kevin Lano and Shekoufeh Kolahdouz Rahimi. Case study: Class diagram restructuring. In *TTC 2013*. arXiv:1309.0369, 2013.

[28] Kevin Lano, Sobhan Yassipour-Tehrani, Shekoufeh Kolahdouz-Rahimi, and Mohammadreza Sharbaf. A survey of model transformation design patterns in practice. *Journal of Systems and Software*, 2018.

[29] Grzegorz Loniewski, Emilio Insfran, and Silvia Abrahão. A systematic review of the use of requirements engineering techniques in model-driven development. In *International Conference on Model Driven Engineering Languages and Systems*, pages 213–227. Springer, 2010.

[30] L. Macaulay. Requirements for requirements engineering techniques. In *ICRE '96*, pages 157–164. IEEE press, 1996.

[31] NAM Maiden and Gordon Rugg. ACRE: selecting methods for requirements acquisition. *Software Engineering Journal*, 11(3):183–192, 1996.

[32] R. K. Mitchell, B. R. Agle, and D. J. Wood. Toward a theory of stakeholder identification and salience: Defining the principle of who and what really counts. *Academy of Management Review*, 22(4):853–886, 1997.

[33] Parastoo Mohagheghi, Wasif Gilani, Alin Stefanescu, and Miguel A Fernandez. An empirical study of the state of the practice and acceptance of model-driven engineering in four industrial cases. *Empirical Software Engineering*, 18(1):89–116, 2013.

[34] OMG. MOF2 Query/View/Transformation specification, v1.3, 2016.

[35] Klaus Pohl. *Requirements Engineering: Fundamentals, Principles, and Techniques.* Springer, 2010.

[36] Suzanne Robertson and James Robertson. *Mastering the Requirements Process (2nd Edition).* Addison-Wesley Professional, 2006.

[37] Louis M Rose, Dimitrios S Kolovos, Richard F Paige, and FA Polack. Model migration case for TTC 2010. *Transformation Tool Contest 2010 1-2 July 2010, Malaga, Spain*, page 1, 2010.

[38] Bran Selic. What will it take? a view on adoption of model-based methods in practice. *Software & Systems Modeling*, 11(4):513–526, 2012.

[39] Dag I. K. Sjoberg, Tore Dyba, Bente C. D. Anda, and Jo E. Hannay. *Building Theories in Software Engineering Guide to Advanced Empirical Software Engineering, Chapter: 12.* Springer-Verlag, 2008.

[40] Ian Sommerville and Gerald Kotonya. *Requirements engineering: processes and techniques.* John Wiley & Sons, Inc., 1998.

[41] Professor Ian Sommerville. private communication, 2015.

[42] Klaas-Jan Stol and Brian Fitzgerald. Theory-oriented software engineering. *Science of Computer Programming*, 101:79–98, 2015.

[43] Eugene Syriani, Jeff Gray, and Hans Vangheluwe. *Modeling a Model Transformation Language*, pages 211–237. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[44] Axel van Lamsweerde. *Requirements Engineering: From System Goals to UML Models to Software Specifications.* John Wiley & Sons, 9 Jan 2009.

[45] Jon Whittle, John Hutchinson, and Mark Rouncefield. The state of practice in model-driven engineering. *Software, IEEE*, 31(3):79–85, 2014.

[46] Karl E. Wiegers and Joy Beatty. *Software Requirements.* Microsoft Press, 2013.

[47] Claes Wohlin, Darja Asmite, and Nils Brede Moe. A general theory of software engineering: Balancing human, social and organizational capitals. *The Journal of Systems and Software*, 109:229–242, 2015.

[48] A. Wortmann, O. Barais, B. Combemale, and M. Wimmer. Modeling languages in Industry 4.0: an extended systematic mapping study. *SoSyM*, 19:67–94, 2020.

[49] Sobhan Yassipour Tehrani and Kevin Charles Lano. Model transformation applications from requirements engineering perspective. In *The 10th International Conference on Software Engineering Advances*, 2015.

Surveyed Papers

[P1] Vahdat Abdelzad, Hamoud Aljamaan, Opeyemi Adesina, Miguel A. Garzon, and Timothy C. Lethbridge. A model-driven solution for financial data representation expressed in FIXML. *CEUR Workshop Proceedings*, 1305:65–70, 2014.

[P2] Mathieu Acher, Philippe Lahire, Sabine Moisan, and Jean Paul Rigault. Tackling high variability in video surveillance systems through a model transformation approach. *Proceedings - International Conference on Software Engineering*, 82(7):44–49, 2009.

[P3] Nouha Adadi, Mohammed Berrada, Mohamed Halim, and Driss Chenouni. Modeling and implementation of web services composition based on MARDS. *International Journal of Advanced Trends in Computer Science and Engineering*, 8(6):3381–3388, 2019.

[P4] C. Amelunxen and A. Schürr. Formalising model transformation rules for UML/MOF 2. *IET Software*, 2(3):204–222, 2008.

[P5] Kyriakos Anastasakis, Behzad Bordbar, Geri Georg, and Indrakshi Ray. On challenges of model transformation from UML to Alloy. *Software and Systems Modeling*, 9(1):69–86, 2010.

[P6] Pedro Azevedo and Ricardo Jorge C de Oliveira. The SDMLib Solution to the TTC 2017 State Elimination Case. *CEUR Workshop Proceedings*, 2031:31–36, 2017.

[P7] Imran S Bajwa and Mark G Lee. Transformation rules for translating business rules to ocl constraints. In *Modelling Foundations and Applications: 7th European Conference, ECMFA 2011, Birmingham, UK, June 6-9, 2011 Proceedings 7*, pages 132–143. Springer, 2011.

[P8] Francesco Basciani, Daniele Di Pompeo, Davide Di Ruscio, Ludovico Iovino, and Alfonso Pierantonio. Integrating semantic reasoning in information loss-based transformation chain rankers. *Proceedings of the ACM Symposium on Applied Computing*, pages 1494–1503, 2021.

[P9] Francesco Basciani, Davide Di Ruscio, Mattia D'Emidio, Daniele Frigioni, Alfonso Pierantonio, and Ludovico Iovino. A tool for automatically selecting optimal model transformation chains. *21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, MODELS-Companion 2018*, pages 2–6, 2018.

[P10] Valentin Besnard, Frédéric Jouault, and Théo Le CalvarMassimo Tisi. The TTC 2018 Social Media Case, by ATL and AOF. *CEUR Workshop Proceedings*, 2216, 2018.

[P11] Enrico Biermann, Claudia Ermel, and Stefan Jurack. Modeling the "ecore to genmodel" transformation with emf henshin. *Science of Computer Programming*, 82:1138–1152, 2013.

[P12] Kristopher Born. Transformation of Finite State Automata to Regular Expressions using Henshin. *CEUR Workshop Proceedings*, 2031:25–30, 2017.

[P13] Kristopher Born, Stefan Schulz, Daniel Strüber, and Stefan John. Solving the Class Responsibility Assignment Case with Henshin and a Genetic Algorithm. *CEUR Workshop Proceedings*, 1758:45–54, 2016.

[P14] Artur Boronat. YAMTL Solution to the TTC 2018 Social Media Case. *CEUR Workshop Proceedings*, 2310:45–49, 2018.

[P15] Artur Boronat. YAMTL solution to the TTC 2019 bibtextodocbook case. *CEUR Workshop Proceedings*, 2550:73–77, 2019.

[P16] Artur Boronat. YAMTL Solution to the TTC 2019 TT2BDD Case. *CEUR Workshop Proceedings*, 2414, 2019.

[P17] Artur Boronat. EMF-Syncer solution to TTC'20 round-trip migration case. *CEUR Workshop Proceedings*, 2694, 2020.

[P18] Artur Boronat. YAMTL solution for the TTC 2021 Laboratory Workflows case. *CEUR Workshop Proceedings*, 3089, 2021.

[P19] Younes Boubekeur, Prabhsimran Singh, and Gunter Mussbacher. A DSL and model transformations to specify learning corpora for modeling assistants. *Proceedings - ACM/IEEE 25th International Conference on Model Driven Engineering Languages and Systems, MODELS 2022: Companion Proceedings*, pages 95–102, 2022.

[P20] Di Penta M Garzotto F Pierantonio M Braico M. Agile Model-Driven Engineering in Mechatronic Systems - An Industrial Case Study. *CEUR Workshop Proceedings*, 1431:170–175, 2015.

[P21] Di Penta M Garzotto F Pierantonio M Braico M. Applying mde to the (semi-) automatic development of model transformations. *CEUR Workshop Proceedings*, 1431:236–241, 2015.

[P22] Di Penta M Garzotto F Pierantonio M Braico M. Case study: Bpmn to bpel model transformation. *CEUR Workshop Proceedings*, 1431:104–109, 2015.

[P23] Di Penta M Garzotto F Pierantonio M Braico M. Developing bp-driven web applications through the use of mde techniques. *CEUR Workshop Proceedings*, 1431:194–199, 2015.

[P24] Di Penta M Garzotto F Pierantonio M Braico M. Lifting transformational models of product lines: A case study. *CEUR Workshop Proceedings*, 1431:128–133, 2015.

[P25] Di Penta M Garzotto F Pierantonio M Braico M. Source transformation of c++ codes for compatibility with operator overloading. *CEUR Workshop Proceedings*, 1431:146–151, 2015.

[P26] Di Penta M Garzotto F Pierantonio M Braico M. Towards rule-based detection of design patterns in model transformations. *CEUR Workshop Proceedings*, 1431:206–211, 2015.

[P27] Di Penta M Garzotto F Pierantonio M Braico M. Trajectory tracking by tp model transformation: case study of a benchmark problem. *CEUR Workshop Proceedings*, 1431:200–205, 2015.

[P28] Di Penta M Garzotto F Pierantonio M Braico M. An nmf solution for the flowgraphs case study at the ttc 2013. *Transformation Tool Contest*, 2013.

[P29] Di Penta M Garzotto F Pierantonio M Braico M. An nmf solution for the petri nets to state charts case study at the ttc 2013. *Transformation Tool Contest*, 2013.

[P30] Di Penta M Garzotto F Pierantonio M Braico M. An nmf solution to the java refactoring case at the ttc 2015. *CEUR Workshop Proceedings*, 1431:21–26, 2015.

[P31] Di Penta M Garzotto F Pierantonio M Braico M. An nmf solution to the train benchmark case at the ttc 2015. *CEUR Workshop Proceedings*, 1431:27–32, 2015.

[P32] Mary Brown. Migrating uml activity models with cope. *ACM Transactions on Software Engineering and Methodology*, 18(3):1–20, 2019.

[P33] Thomas Buchmann. A BXtendDSL Solution to the TTC2023 Asymmetric and Directed Bidirectional Transformation for Container Orchestrations Case. *CEUR Workshop Proceedings*, 3345, 2023.

[P34] Thomas Buchmann. A BXtendDSL Solution to the TTC2023 Incremental MTL vs. GPLs Case. *CEUR Workshop Proceedings*, 3345, 2023.

[P35] Thomas Buchmann and Felix Schwägerl. Using meta-code generation to realize higher-order model transformations. *ICSOFT 2013 - Proceedings of the 8th International Joint Conference on Software Technologies*, 10(3):536–541, 2013.

[P36] Runde M Buck T. Refactoring java programs using spoon. *CEUR Workshop Proceedings*, 1431:98–103, 2015.

[P37] Alexandru Burdusel and Steen Zschaler. Model optimisation for feature-class allocation using MDEOptimiser: A TTC 2016 submission. *CEUR Workshop Proceedings*, 1758:33–38, 2016.

[P38] Luna M M Cabot J. Analyzing flowgraphs with atl. *Transformation Tool Contest*, 2011.

[P39] Luna M M Cabot J. Solving the flowgraphs case with eclectic. *Transformation Tool Contest*, 2011.

[P40] Javier Luis Cánovas Izquierdo and Jesús García Molina. Extracting models from source code in software modernization. *Software and Systems Modeling*, 13(2):713–734, 2014.

[P41] Bassim Chabibi, Abdelilah Douche, Adil Anwar, and Mahmoud Nassar. Integrating SysML with simulation environments (Simulink) by model transformation approach. *Proceedings - 25th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE 2016*, pages 148–150, 2016.

[P42] Kai Chen, Janos Sztipanovits, Sherif Abdelwalhed, and Ethan Jackson. Semantic anchoring with model transformations. In *Model Driven Architecture–Foundations and Applications: First European Conference, ECMDA-FA 2005, Nuremberg, Germany, November 7-10, 2005. Proceedings 1*, pages 115–129. Springer, 2005.

[P43] Howard Chivers and Richard F Paige. Xround: A reversible template language and its application in model-based security analysis. *Information and Software Technology*, 51(5):876–893, 2009.

[P44] Antonio Cicchetti, Bart Meyers, and Manuel Wimmer. Abstract and Concrete Syntax Migration of Instance Models. *Software and Systems Modeling*, 16(3):1–7, 2017.

[P45] Federico Ciccozzi, Antonio Cicchetti, Toni Siljamäki, and Jenis Kavadiya. Automating test cases generation: From xtUML system models to QML test models. *Proceedings - 7th International Workshop on Model-based Methodologies for Pervasive and Embedded Software, MOMPES 2010*, 18(1):9–16, 2010.

[P46] Demuth M Eichelberger H Lotz T Cichowski A. A solution to the java refactoring case study using emoflon. *CEUR Workshop Proceedings*, 1431:9–14, 2015.

[P47] Alessandro Colantoni, Luca Berardinelli, and Manuel Wimmer. DevOpsML: Towards modeling DevOps processes and platforms. *Proceedings - 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS-C 2020 - Companion Proceedings*, pages 480–489, 2020.

[P48] Sebastian Copei, Adrian Kunz, and Albert Zuendorf. The Fulib solution to the TTC 2021 laboratory workflow case. *CEUR Workshop Proceedings*, 3089, 2022.

[P49] Sebastian Copei and Albert Zuendorf. The Fulib solution to the TTC 2020 migration case. *CEUR Workshop Proceedings*, 3089, 2022.

[P50] Xiaofeng Cui. Co-design of the business and software architectures: A systems engineering and model-driven method. *Proceedings - 11th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD2010*, 11(3-4):209–214, 2010.

[P51] Alcino Cunha, Ana Garis, and Daniel Riesco. Translating between Alloy specifications and UML class diagrams annotated with OCL. *Software and Systems Modeling*, 14(1):5–25, 2015.

[P52] Jácome Cunha, João P. Fernandes, Jorge Mendes, Hugo Pacheco, and João Saraiva. Bidirectional transformation of model-driven spreadsheets. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7307 LNCS(1):105–120, 2012.

[P53] Jácome Cunha, João Saraiva, and Joost Visser. From spreadsheets to relational databases and back. *Proceedings of the 2009 ACM SIGPLAN Symposium on Partial Evaluation and Program Manipulation, PEPM'09*, 10(3):179–188, 2009.

[P54] António Miguel Rosado Da Cruz and João Pascoal Faria. A metamodel-based approach for automatic user interface generation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6394 LNCS(PART 1):256–270, 2010.

[P55] Karim Dahman, François Charoy, and Claude Godart. Generation of component-based architecture from business processes: model-driven engineering for SOA. *IEEE European Conference on Web Services*, 81(8):1407–1425, 2008.

[P56] Li Dan. QVT based model transformation from sequence diagram to CSP. *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS*, 5(2):349–354, 2010.

[P57] Gwendal Daniel, Frederic Jouault, Gerson Sunye, and Jordi Cabot. Gremlin-ATL: A scalable model transformation framework. *ASE 2017 - Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering*, pages 462–472, 2017.

[P58] Valeria De Castro, Juan Manuel Vara Mesa, Elisa Herrmann, and Esperanza Marcos. From real computational independent models to information system models: An MDE approach. *CEUR Workshop Proceedings*, 389(8):46–60, 2008.

[P59] Amir Hossein Ghamarian Maarten de Mol and Arend Rensink Eduardo Zambon. Solving the Topology Analysis Case Study with GROOVE. *Transformation Tool Contest 2010*, 229:119, 2010.

[P60] Zekai Demirezen, Marjan Mernik, Jeff Gray, and Barrett Bryant. Verification of DSMLs using graph transformation: A case study with alloy. *ACM International Conference Proceeding Series*, 413(8):987–1002, 2009.

[P61] Joachim Denil, Pieter J. Mosterman, and Hans Vangheluwe. Rule-based model transformation for, and in simulink. *Simulation Series*, 46(4):24–31, 2014.

[P62] Antinisca Di Marco and Stefano Pace. Model-driven approach to Agilla Agent generation. *International Wireless Communications and Mobile Computing Conference*, 16(2):1482–1487, 2013.

[P63] Davide Di Ruscio, Juergen Etzlstorfer, Ludovico Iovino, Alfonso Pierantonio, and Wieland Schwinger. A feature-based approach for variability exploration and resolution in model transformation migration. In *European Conference on Modelling Foundations and Applications*, pages 71–89. Springer, 2017.

[P64] Oscar Díaz, Gorka Puente, Javier Luis Cánovas Izquierdo, and Jesús García Molina. Harvesting models from web 2.0 databases. *Software and Systems Modeling*, 12(1):15–34, 2013.

[P65] Cohen E Diskin Z. Case study: Class diagram restructuring. *Transformation Tool Contest*, 2006.

[P66] Jane Doe. Model migration with mola. *ACM Transactions on Software Engineering and Methodology*, 21(4):21–42, 2022.

[P67] John Doe. Migrating activity diagrams with epsilon flock. *IEEE Transactions on Software Engineering*, 50(1):1–10, 2023.

[P68] Michel Dos Santos Soares and Jos Vrancken. A metamodeling approach to transform uml 2.0 sequence diagrams to petri nets. *Proceedings of the IASTED International Conference on Software Engineering, SE 2008*, 20(2):159–164, 2008.

[P69] de Lara J Eick S. M. FUML activity diagrams with rag-controlled rewriting: A racr solution of the ttc 2015 model execution case". *CEUR Workshop Proceedings*, 1431:63–74, 2015.

[P70] Christoph Eickhoff, Tobias George, Stefan Lindel, and Albert Zündorf. The SDMLib solution to the FIXML case for TTC2014. *CEUR Workshop Proceedings*, 1305:22–26, 2014.

[P71] Christoph Eickhoff, Lennert Raesch, and Albert Zündorf. The SDMLib solution to the Class Responsibility Assignment Case for TTC2016. *CEUR Workshop Proceedings*, 1758:27–32, 2016.

[P72] Simon-Lennert Raesch Eickhoff, Christoph and Albert Zündorf. The EMFeR Solution to the TTC 2018 Software Mapping Case. *CEUR Workshop Proceedings*, 2230, 2018.

[P73] Gregor Engels, Anneke Kleppe, Arend Rensink, Maria Semenyak, Christian Soltenborn, and Heike Wehrheim. From uml activities to taal-towards behaviour-preserving model transformations. In *Model Driven Architecture–Foundations and Applications: 4th European Conference, ECMDA-FA 2008, Berlin, Germany, June 9-13, 2008. Proceedings 4*, pages 94–109. Springer, 2008.

[P74] Hüseyin Ergin and Eugene Syriani. AToMPM solution for the IMDB case study. *CEUR Workshop Proceedings*, 1305:134–138, 2014.

[P75] et al. De Lara J. Deriving multi-agent system behavior. *Requirements Engineering*, 8(2):109–128, 2002.

[P76] et al. Kuhn A. A model-driven software engineering approach for hmis in process industries. *Journal of Systems and Software*, 81(8):1386–1406, 2008.

[P77] Andreas Fuhr, Tassilo Horn, Volker Riediger, and Andreas Winter. Model-driven software migration into service-oriented architectures. *Computer Science - Research and Development*, 28(1):65–84, 2013.

[P78] Mathias Funk, Alexander Nyen, and Horst Lichter. From UML to ANSI-C: An eclipse-based code generation framework. *ICSOFT 2008 - Proceedings of the 3rd International Conference on Software and Data Technologies*, SE(GSDCA/M/-):12–19, 2008.

[P79] Kelly Garcés, Juan M. Vara, Frédéric Jouault, and Esperanza Marcos. Adapting transformations to metamodel changes via external transformation composition. *Software and Systems Modeling*, 13(2):789–806, 2014.

[P80] Antonio García-Domínguez. Hawk solutions to the TTC 2018 Social Media Case. *CEUR Workshop Proceedings*, 2216, 2018.

[P81] Antonio Garcia-Dominguez. The Epsilon Solution to the OCL2PSQL Case. *CEUR Workshop Proceedings*, 3089, 2022.

[P82] Antonio García-Domínguez. The Epsilon Solution to the KMEHR to FHIR Case. *CEUR Workshop Proceedings*, 3421, 2023.

[P83] Christopher Gerking and Christian Heinzemann. Solving the movie database case with QVTo. *CEUR Workshop Proceedings*, 1305:98–102, 2014.

[P84] Marzieh Ghorbani, Mohammadreza Sharbaf, and Bahman Zamani. Incremental Model Transformation with Epsilon in Model-Driven Engineering. *Acta Informatica Pragensia*, 11(2):179–204, 2022.

[P85] Tudor Girba, Jean-Marie Favre, and Stéphane Ducasse. Using meta-model transformation to model software evolution. *Electronic Notes in Theoretical Computer Science*, 6(4):377–400, 2005.

[P86] Johannes Mey Götz, Sebastian, René Schöne, and Uwe Aßmann. A JastAdd- and ILP-based Solution to the Software-Selection and Hardware-Mapping-Problem at the TTC 2018. *CEUR Workshop Proceedings*, 2329, 2018.

[P87] Roy Grønmo, Michael C Jaeger, and Hjørdis Hoff. Transformations between uml and owl-s. In *European Conference on Model Driven Architecture-Foundations and Applications*, pages 269–283. Springer, 2005.

[P88] Roy Grønmo and Birger Møller-Pedersen. From sequence diagrams to state machines by graph transformation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6142 LNCS(4):93–107, 2010.

[P89] Saida Haidrar, Adil Anwar, and Ounsa Roudies. On the use of model transformation for requirements trace models generation. *2017 International Conference on Wireless Technologies, Embedded and Intelligent Systems, WITS 2017*, 2017.

[P90] Ábel Hegedüs, Zoltán Ujhelyi, Gábor Bergmann, and Ákos Horváth. Ecore to Genmodel case study solution using the Viatra2 framework. *Ecore2Genmodel*, (i):1–10, 2010.

[P91] Marwa Hentati, Lassaad Ben Ammar, Abdelwaheb Trabelsi, and Adel Mahfoudhi. An approach for incorporating the usability optimization process into the model transformation. *Advances in Intelligent Systems and Computing*, 557:779–888, 2017.

[P92] Frank Hermann, Susann Gottmann, Nico Nachtigall, Hartmut Ehrig, Benjamin Braatz, Gianluigi Morelli, Alain Pierre, Thomas Engel, and Claudia Ermel. Triple graph grammars in the large for translating satellite procedures. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8568 LNCS(1):122–137, 2014.

[P93] Frank Hermann, Nico Nachtigall, Benjamin Braatz, Susann Gottmann, and Thomas Engel. Solving the FIXML2Code-case study with Henshin TGG. *CEUR Workshop Proceedings*, 1305:32–46, 2014.

[P94] Frank Hilken, Lars Hamann, and Martin Gogolla. Transformation of UML and OCL models into filmstrip models. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8568 LNCS(2):170–185, 2014.

[P95] Georg Hinkel. An NMF solution to the class responsibility assignment case. *CEUR Workshop Proceedings*, 1758(3):15–20, 2016.

[P96] Georg Hinkel. An NMF solution to the Families to Persons case at the TTC 2017. *CEUR Workshop Proceedings*, 2031:7–12, 2017.

[P97] Georg Hinkel. An NMF solution to the Smart Grid Case at the TTC 2017. *CEUR Workshop Proceedings*, 2031:1–6, 2017.

[P98] Georg Hinkel. An NMF solution to the State Elimination case at the TTC 2017. *CEUR Workshop Proceedings*, 2031:13–18, 2017.

[P99] Georg Hinkel. An NMF solution to the TTC 2018 Social Media Case. *CEUR Workshop Proceedings*, 2216, 2018.

[P100] Georg Hinkel. An NMF solution to the TTC 2019 Live Case. *CEUR Workshop Proceedings*, 2563, 2019.

[P101] Georg Hinkel. An NMF solution to the TTC 2019 Truth Tables to Binary Decision Diagrams Case. *CEUR Workshop Proceedings*, 2414, 2019.

[P102] Georg Hinkel. An NMF solution to the TTC 2020 roundtrip engineering case. *CEUR Workshop Proceedings*, 3089, 2022.

[P103] Georg Hinkel. An NMF solution to the TTC 2021 OCL to SQL case. *CEUR Workshop Proceedings*, 3089, 2022.

[P104] Georg Hinkel. An NMF solution to the TTC2021 incremental recompilation of laboratory workflows case. *CEUR Workshop Proceedings*, 3089, 2022.

[P105] Georg Hinkel. An NMF Solutions to the TTC2023 Containers to MiniYAML Case. *CEUR Workshop Proceedings*, 3345, 2023.

[P106] Georg Hinkel. Two NMF Solutions to the TTC2023 Incremental Class to Relational Case. *CEUR Workshop Proceedings*, 3345, 2023.

[P107] Ta'id Holmes and Uwe Zdun. Refactoring architecture models for compliance with custom requirements. *Proceedings - 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS 2018*, pages 267–277, 2018.

[P108] Tassilo Horn. Solving the class diagram restructuring transformation case with funnyqt. *Transformation Tool Contest*, 2013.

[P109] Tassilo Horn. Solving the ttc 2013 flowgraphs case with funnyqt. *Transformation Tool Contest*, 2013.

[P110] Tassilo Horn. Reusable model transformations. *CEUR Workshop Proceedings*, 1305:125–130, 2014.

[P111] Tassilo Horn. Solving the petri-nets to statecharts transformation case with funnyqt. *CEUR Workshop Proceedings*, 1305:53–58, 2014.

[P112] Tassilo Horn. Solving the TTC FIXML case with funny QT. *CEUR Workshop Proceedings*, 1305:7–21, 2014.

[P113] Tassilo Horn. Solving the TTC Movie Database Case with FunnyQT. *CEUR Workshop Proceedings*, 1305:125–133, 2014.

[P114] Tassilo Horn. Solving the ttc java refactoring case with funnyqt. *CEUR Workshop Proceedings*, 1431:81–85, 2015.

[P115] Tassilo Horn. Solving the ttc model execution case with funnyqt. *CEUR Workshop Proceedings*, 1431:75–80, 2015.

[P116] Tassilo Horn. Solving the ttc train benchmark case with funnyqt. *CEUR Workshop Proceedings*, 1431:57–62, 2015.

[P117] Tassilo Horn. Solving the TTC Families to Persons Case with FunnyQT. *CEUR Workshop Proceedings*, 2031:43–48, 2017.

[P118] Koster J Horn T. A solution to the flowgraphs case study using triple graph grammars and emoflon. *Transformation Tool Contest*, 2013.

[P119] Samaneh HoseinDoost, Meysam Karimi, Shekoufeh Kolahdouz Rahimi, and Bahman Zamani. Solving the Quality-based Software-Selection and Hardware-Mapping Problem with ACO. *CEUR Workshop Proceedings*, 2230, 2018.

[P120] Horacio Hoyos, Jaime Chavarriaga, and Paola Gómez. Solving the FIXML case study using epsilon and Java. *CEUR Workshop Proceedings*, 1305:87–92, 2014.

[P121] Akram Idani, German Vega, and Michael Leuschel. Applying formal reasoning to model transformation: The Meeduse solution. *CEUR Workshop Proceedings*, 2550:33–44, 2019.

[P122] Pablo Inostroza and Tijs van Der Storm. The ttc 2014 movie database case: Rascal solution. *CEUR Workshop Proceedings*, 1305:107–112, 2014.

[P123] Pablo Inostroza and Tijs Van Der Storm. The TTC 2014 FIXML case: Rascal solution. *CEUR Workshop Proceedings*, 1305:47–51, 2014.

[P124] Muhammad Zohaib Iqbal, Andrea Arcuri, and Lionel Briand. Environment modeling and simulation for automated testing of soft real-time embedded software. *Software and Systems Modeling*, 14(1):483–524, 2015.

[P125] Edgar Jakumeit. Solving the TTC 2014 movie database case with GrGen.NET. *CEUR Workshop Proceedings*, 1305:125–133, 2014.

[P126] Edgar Jakumeit. Solving the TTC 2014 movie database case with GrGen.NET. *CEUR Workshop Proceedings*, 1305:125–133, 2014.

[P127] Liuyue Jiang, Nguyen Khoi Tran, and Muhammad Ali Babar. Mod2Dash: A Framework for Model-Driven Dashboards Generation. *Proceedings of the ACM on Human-Computer Interaction*, 6(EICS), 2022.

[P128] Álvaro Jiménez, David Granada, Verónica Bollati, and Juan M. Vara. Using ATL to support Model-Driven development of RubyTL model transformations. *CEUR Workshop Proceedings*, 742(4):35–48, 2011.

[P129] Leif Arne Johnsen, Fernando Mac, and Adrian Rutle. Solving the Class Responsibility Assignment using Java and ATL. *CEUR Workshop Proceedings*, 9(1):20–23, 2016.

[P130] Michael Johnson. Topology analysis of car platoons merge with fujabart timedstorychartsÑa case study. *Software and Systems Modeling*, 13(4):451–470, 2014.

[P131] Peter Johnson. Uml model migration with pete. *Journal of Software Engineering and Research*, 7(2):1–10, 2018.

[P132] David Jones. A grgen .net solution of the model migration case for the transformation tool contest 2010. *Software and Systems Modeling*, 19(4):501–520, 2020.

[P133] Robbert Jongeling, Johan Fredriksson, Jan Carlson, Federico Ciccozzi, and Antonio Cicchetti. Structural consistency between a system model and its implementation: a design science study in industry. *be published*, 2022.

[P134] Frédéric Jouault and Théo Le Calvar. (Ab)using Incremental ATL on the TTC 2021 Incremental Laboratory Workflow Benchmark. *CEUR Workshop Proceedings*, 3089, 2021.

[P135] Frédéric Jouault, Theo Le Calvar, and Matthew Coyle. Asymmetric and directed bidirectional transformation for container orchestrations with YAMTL and EMF-Syncer. *Transformation Tool Contest (TTC)*, 2023.

[P136] Frédéric Jouault and Nicolas Pouillard. Cheptre Solution to the TTC 2023 Incremental Class2Relational Case. *Transformation Tool Contest (TTC)*, 2023.

[P137] Frédéric Jouault, Théo Le Calvar, and Matthew Coyle. Incremental ATL Solution to the TTC 2023 KMEHR to FHIR Case. *CEUR Workshop Proceedings*, 3421, 2023.

[P138] Gerd Kainz, Christian Buckl, Stephan Sommer, and Alois Knoll. Model-to-metamodel transformation for the development of component-based systems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6395 LNCS(PART 2):391–405, 2010.

[P139] Gernot Kappel, Martin Rungger, Gabor Csaba, Lszl Vlgyesi, and Paul Greenfield. Domain-specific discrete event modelling and simulation using graph transformation. *Journal of Visual Languages and Computing*, 18(4-5):263–285, 2007.

[P140] Amogh Kavimandan and Aniruddha Gokhale. Automated middleware QoS configuration techniques using model transformations. *Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOC*, 61(3):20–27, 2007.

[P141] Wael Kessentini and Vahid Alizadeh. Interactive metamodel/model co-evolution using unsupervised learning and multi-objective search. *Proceedings - 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS 2020*, pages 68–78, 2020.

[P142] Hasselmann M Klaren H Khasse R. Uml2alloy: A challenging model transformation. *CEUR Workshop Proceedings*, 1431:116–121, 2015.

[P143] Mnnich M Khne O. The petri-nets to statecharts transformation case. *Transformation Tool Contest*, 2002.

[P144] Soon Kyeong Kim, Toby Myers, Marc Florian Wendland, and Peter A. Lindsay. Execution of natural language requirements using state machines synthesised from Behavior Trees. *Journal of Systems and Software*, 85(11):2652–2664, 2012.

[P145] Mathias Kleiner, Marcos Didonet Del Fabro, and Davi De Queiroz Santos. Transformation as Search. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 17(4):54–69, 2013.

[P146] Manninger F Kolb M. Java refactoring case: a viatra solution. *CEUR Workshop Proceedings*, 1431:86–91, 2015.

[P147] Filip Krikava and Philippe Collet. Solving the TTC'14 FIXML case study with SIGMA. *CEUR Workshop Proceedings*, 1305:76–86, 2014.

[P148] Collet P Krikava F. Solving the ttcÕ15 train benchmark case study with sigma. *CEUR Workshop Proceedings*, 1431:33–38, 2015.

[P149] Daniel Strber Kristopher Born Stefan Schulz and Stefan John. Solving the TTC'16 Class Responsibility Assignment Case Study with SIGMA and Multi-Objective Genetic Algorithms. *CEUR Workshop Proceedings*, 1758:55–60, 2016.

[P150] Mirco Kuhlmann and Martin Gogolla. From UML and OCL to relational logic and back. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7590 LNCS(4):415–431, 2012.

[P151] Géza Kulcsár, Erhan Leblebici, and Anthony Anjorin. A solution to the FIXML case study using triple graph grammars and eMoflon. *CEUR Workshop Proceedings*, 1305:71–75, 2014.

[P152] K. Lano, S. Fang, and S. Kolahdouz-Rahimi. TL: An abstract specification language for bidirectional transformations. *Proceedings - 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS-C 2020 - Companion Proceedings*, pages 538–547, 2020.

[P153] K. Lano, S. Fang, M. A. Umar, and S. Yassipour-Tehrani. Enhancing model transformation synthesis using natural language processing. *Proceedings - 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS-C 2020 - Companion Proceedings*, pages 277–286, 2020.

[P154] K. Lano, S. Kolahdouz-Rahimi, and S. Fang. Model Transformation Development Using Automated Requirements Analysis, Metamodel Matching, and Transformation by Example. *ACM Transactions on Software Engineering and Methodology*, 31(2), 2022.

[P155] K. Lano, S. Yassipour-Tehrani, and S. Kolahdouz-Rahimi. Solving the class responsibility assignment case with UML-RSDS. *CEUR Workshop Proceedings*, 1758(3):9–14, 2016.

[P156] K. Lano, S. Yassipour-Tehrani, and K. Maroukian. Case study: FIXML to Java, C and C++. *CEUR Workshop Proceedings*, 1305:2–6, 2014.

[P157] Kevin Lano, T. Clark, and S. Kolahdouz-Rahimi. A framework for model transformation verification. *Formal Aspects of Computing*, 27(1):193–235, 2015.

[P158] Kevin Lano and Alireza Rouhi. KMEHR to FHIR case solution with UML-RSDS. *CEUR Workshop Proceedings*, 3089, 2021.

[P159] Kolahdouz Rahimi S Lano K. Solving the petri-nets to statecharts transformation case with uml-rsds. *Transformation Tool Contest*, 2013.

[P160] Le Gall P Jzquel J.-M. Le Mogudes P. A solution to the ttcÕ15 model execution case using the gemoc studio. *CEUR Workshop Proceedings*, 1431:15–20, 2015.

[P161] Ana León, Maribel Yasmina Santos, Alberto García, Juan Carlos Casamayor, and Oscar Pastor. Model-to-Model Transformation: From UML Class Diagrams to Labeled Property Graphs. *Business and Information Systems Engineering*, 2023.

[P162] Dan Li, Danning Li, Xiaoshan Li, and Volker Stolz. FIXML to Java, C and C++ transformations with QVTR-XSLT. *CEUR Workshop Proceedings*, 1305:27–31, 2014.

[P163] Carsten Lohmann, Joel Greenyer, Juanjuan Jiang, and Tarja Systä. Applying triple graph grammars for pattern-based workflow model transformations. *J. Object Technol.*, 6(9):253–273, 2007.

[P164] Tiziano Lombardi, Vittorio Cortellessa, Alfonso Pierantonio, and In Model. Co-evolution of metamodel and generators: Higher-order templating to the rescue. *J. Object Technol.*, 20(3):7–1, 2021.

[P165] Florian Lugou, Letitia W. Li, Ludovic Apvrille, and Rabéa Ameur-Boulifa. SysML models and model transformation for security. *MODELSWARD 2016 - Proceedings of the 4th International Conference on Model-Driven Engineering and Software Development*, pages 331–338, 2016.

[P166] István Madari, László Angyal, and László Lengyel. Traceability-based incremental model synchronization. *WSEAS Transactions on Computers*, 8(10):1691–1700, 2009.

[P167] A. P. Magalhães, A. Andrade, and R. S. Maciel. A model driven transformation development process for model to model transformation. *ACM International Conference Proceeding Series*, pages 3–12, 2016.

[P168] Salvador Martínez, Jokin García, and Jordi Cabot. Runtime Support for Rule-Based Access-Control Evaluation through Model-Transformation. *SLE 2016 - Proceedings of the 2016 ACM SIGPLAN International Conference on Software Language Engineering, co-located with SPLASH 2016*, 28(4):57–69, 2016.

[P169] Mendonca P da Silva A R Guizzardi G Martins M. Improving the application of agile model-based development: Experiences from case studies. *CEUR Workshop Proceedings*, 1431:110–115, 2015.

[P170] S. Mbarki and M. Erramdani. Model-driven transformations: From analysis to MVC 2 web model. *International Review on Computers and Software*, 4(5):612–620, 2009.

[P171] D. A. Meedeniya and Indika Perera. Model based software design: Tool support for scripting in immersive environments. *International Conference on Industrial and Information Systems, ICIIS 2013 - Conference Proceedings*, 20(4):248–253, 2013.

[P172] Johannes Mey, René Schöne, Christopher Werner, and Uwe Aβmann. Transforming truth tables to binary decision diagrams using relational reference attribute grammars. *CEUR Workshop Proceedings*, 2550:27–32, 2019.

[P173] Milan Milanović, Dragan Gašević, Adrian Giurca, Gerd Wagner, Sergey Lukichev, and Vladan Devedžić. Model transformations to bridge concrete and abstract syntax of web rule languages. *Computer Science and Information Systems*, 6(2):47–85, 2009.

[P174] Antonio Moreno-Delgado and Francisco Durán. The movie database case: Solutions using maude and the maude-based e-motions tool. *CEUR Workshop Proceedings*, 1305:116–124, 2014.

[P175] Jean-Marie Mottu, Benoit Baudry, and Yves Le Traon. Mutation analysis testing for model transformations. In *Model Driven Architecture–Foundations and Applications: Second European Conference, ECMDA-FA 2006, Bilbao, Spain, July 10-13, 2006. Proceedings 2*, pages 376–390. Springer, 2006.

[P176] Olaf Muliawan, Pieter Van Gorp, Anne Keller, and Dirk Janssens. Executing a standard-compliant transformation model on a non-standard platform. *Proceedings of the 6th International Conference on Model Driven Engineering Techniques and Applications*, pages 327–338, 2008.

[P177] Gunter Mussbacher, Jörg Kienzle, and Daniel Amyot. Transformation of aspect-oriented requirements specifications for reactive systems into aspect-oriented design specifications. *Model-Driven Requirements Engineering Workshop, MoDRE 2011*, 142(2):39–47, 2011.

[P178] Andrs Szabolcs Nagy and Gbor Szrnyas. Class Responsiblity Assignment Case: a Viatra-DSE Solution. *CEUR Workshop Proceedings*, 1758:7–19, 2016.

[P179] Felix Neumann. An agile driven architecture modernization to a model-driven development solution. *CEUR Workshop Proceedings*, 1305:149–154, 2014.

[P180] Felix Neumann. Transformation tool soccer worldcup (ttc 2014 live contest case). *CEUR Workshop Proceedings*, 1305:119–124, 2014.

[P181] Felix Neumann and Felix von Wendel. Movie database case: An emf-incquery solution. *CEUR Workshop Proceedings*, 1305:77–82, 2014.

[P182] von Wendel F Neumann F. A component model for model transformations. *CEUR Workshop Proceedings*, 1431:224–229, 2015.

[P183] von Wendel F Neumann F. A uml/ocl framework for the analysis of graph transformation rules. *CEUR Workshop Proceedings*, 1431:122–127, 2015.

[P184] von Wendel F Neumann F. Code generation by model transformation: a case study in transformation modularity. *CEUR Workshop Proceedings*, 1431:134–139, 2015.

[P185] von Wendel F Neumann F. Model migration with gretl. *CEUR Workshop Proceedings*, 1431:242–247, 2015.

[P186] von Wendel F Neumann F. Model transformation for service-oriented web applications development. *CEUR Workshop Proceedings*, 1431:218–223, 2015.

[P187] von Wendel F Neumann F. Model transformations for round-trip engineering in control deployment co-design. *CEUR Workshop Proceedings*, 1431:230–235, 2015.

[P188] von Wendel F Neumann F. Pn2sc case study: An emf-incquery solution. *CEUR Workshop Proceedings*, 1431:3–8, 2015.

[P189] von Wendel F Neumann F. Refactoring object constraint language specifications. *CEUR Workshop Proceedings*, 1431:182–187, 2015.

[P190] von Wendel F Neumann F. Refactoring ocl annotated uml class diagrams. *CEUR Workshop Proceedings*, 1431:176–181, 2015.

[P191] von Wendel F Neumann F. Semantics of ocl specified with qvt. *CEUR Workshop Proceedings*, 1431:164–169, 2015.

[P192] von Wendel F Neumann F. Streaming model transformations by complex event processing. *CEUR Workshop Proceedings*, 1431:212–217, 2015.

[P193] von Wendel F Neumann F. Synthesis of test scenarios using uml activity diagrams. *CEUR Workshop Proceedings*, 1431:188–193, 2015.

[P194] von Wendel F Neumann F. The algorithm of transformation from uml sequence diagrams to the topological functioning model. *CEUR Workshop Proceedings*, 1431:158–163, 2015.

[P195] von Wendel F Neumann F. The atl/emftvm solution to the train benchmark case for ttc2015. *CEUR Workshop Proceedings*, 1431:92–97, 2015.

[P196] von Wendel F Neumann F. The sdmlib solution to the java refactoring case for ttc2015. *CEUR Workshop Proceedings*, 1431:51–56, 2015.

[P197] von Wendel F Neumann F. The sdmlib solution to the model execution case for ttc2015. *CEUR Workshop Proceedings*, 1431:45–50, 2015.

[P198] von Wendel F Neumann F. Train benchmark case: an emf-incquery solution. *CEUR Workshop Proceedings*, 1431:39–44, 2015.

[P199] von Wendel F Neumann F. Transformation from petri nets model to programmable logic controller using one-to-one mapping technique. *CEUR Workshop Proceedings*, 1431:152–157, 2015.

[P200] von Wendel F Neumann F. Using model transformation to refactor use case models based on antipatterns. *CEUR Workshop Proceedings*, 1431:140–145, 2015.

[P201] Hoang Lam Nguyen, Nebras Nassar, Timo Kehrer, and Lars Grunske. MoFuzz: A Fuzzer Suite for Testing Model-Driven Software Engineering Tools - Summary. *Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI)*, P-310:81–82, 2020.

[P202] Oyindamola Olajubu, Suraj Ajit, Mark Johnson, Scott Thomson, Mark Edwards, and Scott Turner. Automated test case generation from high-level logic requirements using model transformation techniques. *2017 9th Computer Science and Electronic Engineering Conference, CEEC 2017 - Proceedings*, pages 178–182, 2017.

[P203] Jon Oldevik, Tor Neple, Roy Grønmo, Jan Aagedal, and Arne-J Berre. Toward standardised model to text transformations. In *European Conference on Model Driven Architecture-Foundations and Applications*, pages 239–253. Springer, 2005.

[P204] Gøran K Olsen and Jon Oldevik. Scenarios of traceability in model to text transformations. In *European Conference on Model Driven Architecture-Foundations and Applications*, pages 144–156. Springer, 2007.

[P205] Óscar Pastor. Generating user interfaces from conceptual models: A modeltransformation based approach. *Computer-Aided Design of User Interfaces V - Proceedings of the 6th International Conference on Computer-Aided Design of User Interfaces, CADUI 2006*, 18(4):1–14, 2007.

[P206] Sven Peldszus, Jens Bürger, and Daniel Strüber. Detecting and Preventing Power Outages in a Smart Grid using eMoflon. *CEUR Workshop Proceedings*, 2026, 2017.

[P207] Ricardo Pérez-Castillo, Ignacio García-Rodríguez De Guzmán, and Mario Piattini. Implementing business process recovery patterns through QVT transformations. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6142 LNCS(1):168–183, 2010.

[P208] Tirdad Rahmani, Daniel Oberle, and Marco Dahms. An adjustable transformation from owl to ecore. *International Conference on Model Driven Engineering Languages and Systems*, 6(2):133–149, 2010.

[P209] Struyf T Rensink A. Class diagram restructuring with groove. *Transformation Tool Contest*, 2006.

[P210] Emanuel Montero Reyno and José Carsí Cubel. Automatic prototyping in model-driven game development. *Computers in Entertainment*, 7(2):1–23, 2009.

[P211] Kolovos D Paige R F Polack F A C Rose L. M. Model migration case for TTC 2010. *Transformation Tool Contest*, 2010.

[P212] Adrian Rutle, Ludovico Iovino, Harald König, and Zinovy Diskin. Automatic transformation co-evolution using traceability models and graph transformation. In *Modelling Foundations and Applications: 14th European Conference, ECMFA 2018, Held as Part of STAF 2018, Toulouse, France, June 26-28, 2018, Proceedings 14*, pages 80–96. Springer, 2018.

[P213] Eman Saleh, Amr Kamel, and Aly Fahmy. A model driven engineering design approach for developing multi-platform user interfaces. *WSEAS Transactions on Computers*, 9(5):536–545, 2010.

[P214] Samin Salemi, Ali Selamat, and Marek Penhaker. A model transformation framework to increase OCL usability. *Journal of King Saud University - Computer and Information Sciences*, 28(1):13–26, 2016.

[P215] Leila Samimi-Dehkordi, Bahman Zamani, and Shekoufeh Kolahdouz Rahimi. Solving the Families to Persons Case using EVL+Strace. *CEUR Workshop Proceedings*, 2031:49–56, 2017.

[P216] Jesús Sánchez Cuadrado, Esther Guerra, and Juan de Lara. Towards the systematic construction of domain-specific transformation languages. In *Modelling Foundations and Applications: 10th European Conference, ECMFA 2014, Held as Part of STAF 2014, York, UK, July 21-25, 2014. Proceedings 10*, pages 196–212. Springer, 2014.

[P217] João Pedro Santos, Ana Moreira, João Araújo, and Miguel Goulão. Increasing quality in scenario modelling with model-driven development. *Proceedings - 7th International Conference on the Quality of Information and Communications Technology, QUATIC 2010*, 11(3):204–209, 2010.

[P218] A. Schauerhuber, M. Wimmer, E. Kapsammer, W. Schwinger, and W. Retschitzegger. Bridging WebML to model-driven engineering: From document type definitions to meta object facility. *IET Software*, 1(3):81–97, 2007.

[P219] René Schöne and Johannes Mey. A JastAdd-based Solution to the TTC 2018 Social Media Case. *CEUR Workshop Proceedings*, 2216, 2018.

[P220] Simon Schwichtenberg, Christian Gerth, Zille Huma, and Gregor Engels. Normalizing heterogeneous service description models with generated QVT transformations. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8569 LNCS(4):180–195, 2014.

[P221] Gehan MK Selim, Shige Wang, James R Cordy, and Juergen Dingel. Model transformations for migrating legacy models: an industrial case study. In *Modelling Foundations and Applications: 8th European Conference, ECMFA 2012, Kgs. Lyngby, Denmark, July 2-5, 2012. Proceedings 8*, pages 90–101. Springer, 2012.

[P222] Gehan M.K. Selim, Shige Wang, James R. Cordy, and Juergen Dingel. Model transformations for migrating legacy deployment models in the automotive industry. *Software and Systems Modeling*, 14(1):365–381, 2015.

[P223] Matt Selway, Georg Grossman, Markus Stumptner, Kerryn R. Owen, and Richard M. Dexter. Integration of visual contracts and model transformation for enhanced MDE development. *Proceedings - 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS 2018*, pages 415–425, 2018.

[P224] M. Sharbaf, S. Kolahdouz-Rahimi, and B. Zamani. Solving the state elimination case study using epsilon. In *CEUR Workshop Proceedings*, volume 2026, 2017.

[P225] Ahmed Siabdelhadi, Abdelhafid Chadli, Hadda Cherroun, Abdelkader Ouared, and Houari Sahraoui. MoTrans-BDI: Leveraging the Beliefs-Desires-Intentions agent architecture for collaborative model transformation by example. *Journal of Computer Languages*, 74, 2023.

[P226] Florian Sihler, Matthias Tichy, and Jakob Pietron. One-way model transformations in the context of the technology-roadmapping tool iris. *Ratio*, 1:1–8.

[P227] Fábio Levy Siqueira and Paulo Sérgio Muniz Silva. Transforming an enterprise model into a use case model in business process systems. *Journal of Systems and Software*, 96(1):152–171, 2014.

[P228] Fabio Levy Siqueira, Paulo Sérgio Muniz Silva, and Paulo Śrgio Muniz Silva. Transforming an enterprise model into a use case model using existing heuristics. *2011 Model-Driven Requirements Engineering Workshop, MoDRE 2011*, 14(5):21–30, 2011.

[P229] Kim Smith. Uml1.4 to 2.1 activity diagram model migration with fujabaÑa case study. *Journal of Systems and Software*, 126:101–120, 2021.

[P230] Hui Song, Gang Huang, Franck Chauvel, Wei Zhang, Yanchun Sun, Weizhong Shao, and Hong Mei. Instant and incremental QVT transformation for runtime models. *Model Driven Engineering Languages and Systems: 14th International Conference*, 13(1-2):7–20, 2011.

[P231] Nikolaos Spanoudakis and Pavlos Moraitis. Modular JADE agents design and implementation using ASEME. *Proceedings - 2010 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2010*, 2(1):221–228, 2010.

[P232] Jim Steel and Robin Drogemuller. Domain-specific model transformation in building quantity take-off. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6981 LNCS(3):198–212, 2011.

[P233] Friedrich Steimann. Constraint-based model refactoring. *International Conference on Model Driven Engineering Languages and Systems*, 19(3):177–200, 2011.

[P234] Ashley Sterritt and Vinny Cahill. Customisable model transformations based on non-functional requirements. *Proceedings - 2008 IEEE Congress on Services, SERVICES 2008*, PART 1(4):329–336, 2008.

[P235] Daniel Strüber. Supporting round-trip data migration for web APIs with Henshin. *CEUR Workshop Proceedings*, 3089, 2022.

[P236] Eugene Syriani and Hüseyin Ergin. Operational semantics of UML activity diagram: An application in project management. *IEEE International Workshop on Model-Driven Requirements Engineering, MoDRE 2012 - Proceedings*, 50(1-2):1–8, 2012.

[P237] Eugene Syriani and Hans Vangheluwe. Programmed graph rewriting with time for simulation-based design. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5063 LNCS(1):91–106, 2008.

[P238] Eugene Syriani and Hans Vangheluwe. A modular timed graph transformation language for simulation-based design. *Software and Systems Modeling*, 12(2):387–414, 2013.

[P239] F. Taghizadeh and S. R. Taghizadeh. A graph transformation-based approach for applying MDA to SOA. *4th International Conference on Frontier of Computer Science and Technology, FCST 2009*, 9(4):446–451, 2009.

[P240] Ann Taylor. Abstract topology analysis of the join phase of the merge protocol. *ACM Transactions on Software Engineering and Methodology*, 24(4):1–25, 2015.

[P241] Michele Tucci. Model-driven round-trip software dependability engineering. *21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, MODELS-Companion 2018*, pages 186–191, 2018.

[P242] M. F. Van Amstel, M. G.J. Van Den Brand, Z. Protić, and T. Verhoeff. Transforming process algebra models into UML state machines: Bridging a semantic gap? *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5063 LNCS(2):61–75, 2008.

[P243] Jeroen Van Den Bos and Tijs Van Der Storm. Domain-specific optimization in digital forensics. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7307 LNCS(1):121–136, 2012.

[P244] Dániel Varró, Gábor Bergmann, Ábel Hegedüs, Ákos Horváth, István Ráth, and Zoltán Ujhelyi. Road to a reactive and incremental model transformation platform: three generations of the VIATRA framework. *Software and Systems Modeling*, 15(3):609–629, 2016.

[P245] Maximiliano Vela, Yngve Lamo, Fazle Rabbi, and Fernando Macías. A heuristic approach for resolving the Class Responsibility Assignment Case. *CEUR Workshop Proceedings*, 1758(4):21–26, 2016.

[P246] Jens von Pilgrim. Ecore2GenModel with Mitra and GEF3D. *Transformation Tool Contest 2010*, 21(1):166, 2012.

[P247] Dennis Wagelaar, Ludovico Iovino, Davide Di Ruscio, and Alfonso Pierantonio. Translational semantics of a co-evolution specific language with the EMF transformation virtual machine. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7307 LNCS(4):192–207, 2012.

[P248] Dennis Wagelaar, Théo Le Calvar, and Frédéric Jouault. Truth tables to binary decision diagrams in modern ATL. *CEUR Workshop Proceedings*, 2550:51–58, 2019.

[P249] Yichuan Wang, Lee Ann Kung, William Yu Chung Wang, and Casey G. Cegielski. An integrated big data analytics-enabled transformation model: Application to health care. *Information and Management*, 55(1):64–79, 2018.

[P250] Nils Weidmann, Shubhangi Salunkhe, Anthony Anjorin, Enes Yigitbas, and Gregor Engels. Automating model transformations for railway systems engineering. *J. Object Technol.*, 20(3):10–1, 2021.

[P251] Christopher Werner, Rico Bergmann, Johannes Mey, René Schöne, and Uwe Aßmann. Transforming Truth Tables to Binary Decision Diagrams

using the Role-based Synchronization Approach. *CEUR Workshop Proceedings*, 2550:27–32, 2019.

[P252] M. Wimmer, A. Kusel, J. Schoenboeck, W. Retschitzegger, W. Schwinger, and G. Kappel. On using inplace transformations for model co-evolution. *CEUR Workshop Proceedings*, 711(3):65–78, 2010.

[P253] Manuel Wimmer. A semi-automatic approach for bridging DSMLs with UML. *International Journal of Web Information Systems*, 5(3):372–404, 2009.

[P254] Sabine Winetzhammer, Thomas Buchmann, and Bernhard Westfechtel. MODGRAPH-A Transformation Engine for EMF Model Transformations. *International Conference on Software and Data Technologies*, 17(3):1–40, 2008.

[P255] Sophie Wood, Nicholas Matragkas, Dimitris Kolovos, Richard Paige, and Simos Gerasimou. Supporting robotic software migration using static analysis and model-driven engineering. *Proceedings - 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS 2020*, pages 154–164, 2020.

[P256] Yingfei Xiong, Hui Song, Zhenjiang Hu, and Masato Takeichi. Synchronizing concurrent model updates based on bidirectional transformation. *Software and Systems Modeling*, 12(1):89–104, 2013.

[P257] 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*, 93(4):42–68, 2014.

[P258] Tao Yue and Shaukat Ali. Bridging the gap between requirements and aspect state machines to support non-functional testing: Industrial case studies. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7349 LNCS(3):133–145, 2012.

[P259] Haiping Zha, Wil M.P. van der Aalst, Jianmin Wang, Lijie Wen, and Jiaguang Sun. Verifying workflow processes: A transformation-based approach. *Software and Systems Modeling*, 10(2):253–264, 2011.

[P260] Steffen Zschaler and Sobhan Yassipour Tehrani. Mapping FIXML to OO with aspectual code generators. *CEUR Workshop Proceedings*, 1305:52–64, 2014.

[P261] Albert Zündorf. The Fulib Solution to the TTC 2019 Truth Tables to Binary Decision Diagram Case. *CEUR Workshop Proceedings*, 2414, 2019.

[P262] Albert Zündorf and Alexander Weidt. The SDMLib Solution to the TTC 2017 Families 2 Persons Case. *CEUR Workshop Proceedings*, 2031:19–24, 2017.