Requirements Engineering in Model-Transformation Development: An Interview-Based Study

Sobhan Yassipour Tehrani, Steffen Zschaler, Kevin Lano

Dept of Informatics, King's College London, Strand, London, WC2R 2LS, U.K.
E-mail: {sobhan.yassipour_tehrani, steffen.zschaler, kevin.lano}@kcl.ac.uk

Abstract. Model Transformations (MT) are central building blocks of Model Driven Engineering (MDE). The size and complexity of model transformations grows as they see more wide-spread use in industry. As a result, systematic approaches to the development of high-quality and highly reliable model transformations become increasingly important. However, because little is known about the context in which model transformations are developed, it is very difficult to know what would be required from such systematic approaches. This paper provides some initial results and analysis of an interview-based study of requirements engineering (RE) in MT developments. We have interviewed industry experts in MT development, with the goal of understanding the contexts and ways in which transformations are developed and how their requirements are established. The types of stakeholders of transformations were identified, as well as their role in the transformation development. We also discovered a possible differentiation amongst the development of model transformation projects and general software development projects.

1 Introduction

Model transformations (MTs) are central to model-driven engineering (MDE) [10]. They can be used for a range of purposes, including to improve the quality of models, to refactor models, to migrate or translate models from one representation to another, and to generate code or other artifacts from models [6]. Model transformations either transform one model into another or generate text (such as code) from a model. In any case, they aim to automate repetitive development tasks, ensuring different situations are treated in a generalised manner.

As MDE is being used more intensively [4], systematic development of the transformations becomes more important [2]. However, as Selic argues [9]: "we are far from making the writing of model transformations an established and repeatable technical task". The software engineering of model transformations has only recently been considered in a systematic way, and most of this work has focussed on design and verification rather than on requirements engineering (RE).

We are interested in understanding what requirements engineering for model-transformation development should look like. 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. This will help us understand how existing RE techniques might be applied (or may have to be adapted) for the context of MT development.

In this paper, we report on the results of an exploratory interview-based study with five industry experts in model-transformation. We discuss the types of projects often seen in model-transformation development, their embedding in the context of other projects and organisations, the roles of stakeholders, and the requirements engineering techniques employed in practice, and we consider future research directions.

The remainder of this paper is structured as follows: After a brief discussion of our methodology in Sect. 2 and related work in Sect. 3, we present some of our findings from the interviews. We begin with a discussion of the types of projects identified in Sect. 4, followed by a discussion of stakeholders involved in Sect. 5. Section 6 discusses the requirements engineering techniques identified by our participants, followed by a brief analysis of project outcomes in Sect. 7. Finally, we conclude and discuss future research directions.

2 Methodology

This paper is a result of an exploratory interview-based study based on industrial model transformation projects. The aim of this study is to explore transformation projects from a requirements engineering perspective. Specifically, we are interested in finding out what requirements engineering techniques, if any, are applied in model-transformation development.

We identified five participants that are experts in the MT development field and have industrial experience. The selection was based on participants experience and the work that they have done. Our participants have between eight to twenty years of experience in MT development. We asked participants to focus their responses on self-selected recent projects. All participants had a leading role in these projects. Participants were interviewed regarding the project(s) in which they were involved (seven projects in total), and their views regarding the requirements engineering process in relation to these projects.

We conducted semi-structured interviews of approximately one hour duration. The same questions in the same order were given to all participants. The questions concerned the project context and scale, the stakeholders, the requirements engineering techniques and process used, and the project outcomes.

Our approach is, thus, qualitative investigating in depth the 'why' and 'how' of decision making for particular requirements engineering techniques and activities in model-transformation development. More information about the interview prompts can be found via the link in footnote [1].

[1] `http://www.inf.kcl.ac.uk/pg/tehrani/form.pdf`

Threats to the validity of conclusions drawn from the interviews include: (i) that the interviewees and examined cases are not representative of transformation developers and projects; (ii) that interviewees selected unrepresentative projects; (iii) that interview questions were aimed at elicitating a particular response.

We tried to avoid problem (i) by requesting interviews with a wide range of MT experts. The candidates for interview were selected from our previous literature surveys of RE in MT. 12 candidates were approached, of whom 5 agreed to be interviewed. These represent a diverse range of organisations, and the projects cover a range of domains: embedded systems, finance, re-engineering, defence and business. Regarding (ii), projects with poor outcomes, such as 3 and 6, were included in addition to successful projects. Regarding (iii), the questionnaire and methodology was examined by an expert committee for ethical approval. The survey will be extended with further interview subjects and projects where possible.

3 Related Work

There has been very limited empirical research into model-transformation development. The only relevant studies have been based on MDE in general, such as that of [4, 14], which used interviews as well as a questionnaire-based survey. The main aim of this study was to capture the success and failure factors for MDE based on industry evidence. They conducted 22 interviews with MDE practitioners. 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 this survey suggests that transformations are often developed based on the expert knowledge of software developers, to encode and automate previously manual procedures. A high degree of domain knowledge appears essential for the successful construction of the transformations. The survey of [7] considered in depth four companies adopting MDE, but did not specifically consider requirements engineering. One concern of the companies in [7] was the cost of developing transformations, a factor which could be improved by more systematic RE for MT.

In our work, we focus specifically on model transformation developments, whether as part of an MDE process or as independent developments. For MT developments, we examine how RE techniques and the RE process is carried out.

4 Transformation Development Projects

In this section, we will describe the MT projects which our participants focused on in their descriptions. All of our interviewees are either the sole developers or the lead developers for these projects. Each project has been categorised according to the MT field that it belongs to. The scale, developers time and effort for some of these projects will also be described.

Seven MT development projects were considered in this study:

1. ***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 of the order of 600 meta-classes. The development effort was not available.
2. ***Reverse-engineering and re-engineering of banking systems and web-services:*** the idea of this project was to build transformations to construct models of existing applications, and to forward-engineer these models to new platforms. 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 approx 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. The development effort was not available.
3. ***Code-generation of embedded software from DSLs:*** in this project transformations are defined to map between embedded system DSLs forming C extensions, and from these DSLs to C code. These extensions are used by embedded software developers. More than 25 different DSLs are involved, and approx 30 person-years of effort.
4. ***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. Effort was three person-months.
5. ***Big Data analysis of IMDb:*** the Internet Movie Database (www.imdb.com) can be regarded as a Big Data case. It has information about the title of movies, names of actors, rating of movies and actors playing roles in which movies. In this case, a model transformation was developed to implement IMDb searches by users. Effort was 3 person-months.
6. ***UML to C++ code generator:*** this case involved the construction of a transformation for the generation of multi-threaded/multi-processor code from UML. The transformation generates C++ code as well as providing a run-time layer to support the generator. Effort was four person-years.
7. ***Reverse-engineering of a code generator:*** This MT project was an example of re-engineering of an existing transformation. In this case study an existing code-generation transformation was analysed and re-engineered to improve its functionality. Effort was four person-months.

4.1 Type of Projects

Software development projects can be classified into several types [13]:

Greenfield vs Brownfield In a greenfield type of project, the system is completely new, therefore the developers have to start from scratch and build the system from the beginning. On the other hand, in brownfield projects, a system already exists but it has to be further developed and improved.

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 types of 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 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 type of 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" [13].

According to our interviews, one of the seven projects can be regarded as a brownfield project (Project 7). Six projects were greenfield as the transformation had to be written from scratch, because either the transformation project was completely new, or because developers wanted to use their own tools and technology.

All projects were customer-driven as they were specified for particular client(s). All the projects were in-house, single-product projects. The projects were assigned to a particular company to do all the transformations, therefore there was no need of outsourcing, and only a single version of the project was developed.

MT development often occurs within a wider software development project (e.g., Projects 2, 4, 6, 7), although there are also cases where MT development is the main part of software development (e.g., Projects 1, 3, 5).

As a result, it is important to differentiate explicitly between properties of the transformation-development project and the project this was embedded in. For example, while most of containing projects were brownfield projects, most of the transformation-development projects were greenfield as no previous transformation existed for the specific purpose required.

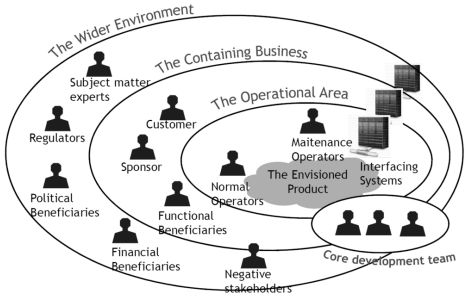

Fig. 1. Onion model of stakeholder general relationship [1]

5 Stakeholders

In general, the term stakeholder can be defined as an individual or an organisation/group of people who is either affected by or has an effect on the outcome of a given project [8]. 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. A good participation of stakeholders in the software development cycle not only would result in a better understanding of the actual problem, but also help to build that which is required according to the stakeholders' needs. The onion model of project stakeholders (*e.g.,* [1], see Fig. 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 somehow benefit from the system. The wider environment area contains stakeholders which have an effect on or interest in the system, but only an in-direct influence.

More specifically, sponsors are stakeholders that have the responsibility to pay for the developed product. Customer(s) buy the product. Sometimes it can be the case where the customer is also the end user 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, may include domain analysts, business consultants, business analysts, or anyone else who has some specialized knowledge of the business subject" [8].

We have adapted the onion model to classify the stakeholders in MT development based on our participants' descriptions. We can identify that the core development team consisted of the transformation developers for all of the MT projects. The customer(s) consisted of the committee that were interacting with the transformation developers in order to explain the problem space and what is needed. The sponsor(s) were the companies which were represented by the

Table 1. Stakeholders of model transformation projects

Case	*Sponsor and Customer*	*Normal and Maintenance Operator*
1	Technology standards consortium	Users of the standards
2	Financial/Telecom organisations	Users of re-engineered systems
3	Commercial companies	Embedded software developers
4	External customer	Users of the output model
5	External customer	Users searching the data
6	Government & Defence industries	Users of C++ application
7	Commercial client	Users of the code generator

customers, and do not interact with MT developers directly. Finally, the normal and maintenance operator consisted of the people who were going to use the result of the transformations as end users. Table 1 presents the sponsors, customers and the operators of the MT projects.

As discussed earlier, the MT projects that we analysed are typically embedded within wider projects. As a result, the role of stakeholders of the wider project was changed according to the embedded MT project. For example, in one case (project 2) the members of the core development team of the wider project turned into the customers interacting with transformation developers for technical issues. Therefore, the transformation developers were facing two types of customer for this project: one to explain the general requirements of the overall system and one to deal with more detailed requirements and technical difficulties of the transformation.

Similarly, the impact of other stakeholders of the containing project (*e.g.,* from the containing business or wider environment) on the transformation development has 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. For example, the indirect nature of contact with the stakeholders of the enclosing development project is likely to impact on the use of RE techniques that require stakeholder interaction. Figure 2 is a first attempt at showing some of the relationships amongst the MT developers and general stakeholders in a generalised onion model.

6 Requirements Engineering Process

In this section, we will discuss our findings regarding the requirements engineering process applied in the projects we discussed in our interviews. We start by discussing the overall RE process used, before focusing on requirements elicitation and cataloguing typical RE techniques employed.

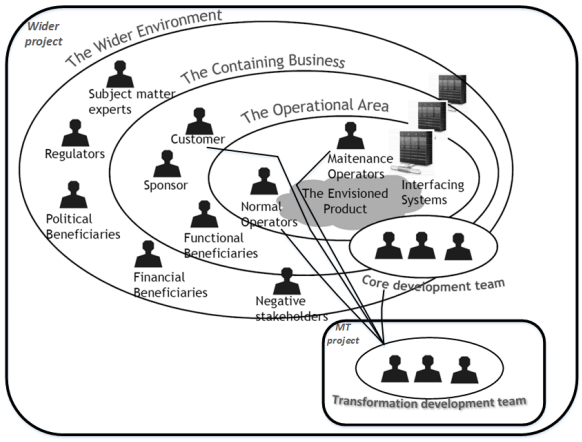

Fig. 2. Onion model of MT stakeholder relationship

6.1 Overall Requirements Engineering Process

Requirements engineering for any type of software development is specialized and model transformation is not an exception. There are some key issues which cause this uniqueness:

Type of system. 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 that are then refined during development.

Type of development process. Plan-based processes require all requirements to be available at the start of the project, whereas in an agile approach, requirements are developed incrementally.

The environment where the system will be deployed. In some cases, users and other stakeholders are available to provide information about the requirements; in others they are not. These require different approaches to RE to get a starting point for implementation.

The extent to which other systems are reused in a system being developed. Generally, requirements for the reused systems are not available. Thus, the RE process needs to reverse engineer these requirements from the existing system [12].

Kotonya *et al* [11] have proposed a process model for the RE process. It is widely accepted by researchers and professional experts. In this study, we used this model as our template to investigate the MT projects. 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 step in the RE process is the act of obtaining detailed knowledge regarding the domain of the current problem, the organization/company con-

fronting 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 identifies inconsistencies and conflicts between requirements. The likelihood of such conflicts will increase if the requirements have been gathered from multiple and different stakeholders. Negotiation with stakeholders takes place to resolve conflicts and potentially infeasible requirements. The specification and documentation phase of the RE process begins with the specification process, which makes precise a set of agreed statements by all relevant sides of the project such as: requirements, assumptions, and system properties. Based on the specification, the requirements documentation can be drafted. At the validation and verification stage, the specifications are analysed. They should be validated by stakeholders to ensure that they satisfy their actual needs. Also, the specification should be verified in order to check its consistency and to avoid conflicts and omissions. Any potential error and flaw must be fixed during this phase and before the actual development in order to save cost, effort and time.

Table 2 shows the requirements engineering processes that were used in the examined MT development projects. Every MT project has been divided into four stages (*elicitation, evaluation, specification* and *validation*) regarding the requirements engineering process. The detailed RE process used was as follows in each project:

Project 1: Document mining, prototyping and interviews were used to obtain requirements. Daily meetings or conference calls with the stakeholders were used to resolve issues. Conflict resolution was used during evaluation, and a UML and QVT/OCL specification was defined. This was validated by inspection.

Project 2: Brainstorming and interviews were used to elicit requirements and decide on the project scope and priorities, together with exploratory prototyping to show the customer what the MT developers intended to develop. An agile process with frequent customer liason was used. During evaluation there were joint requirements development sessions, and negotiation over unrealistic, conflicting or impractical requirements. Impact analysis was used. Semi-formal specifications were used. Testing was used for validation.

Project 3: Brainstorming and prototyping were used, but no formal RE technique was applied. Requirements were categorised and prioritised. Communication with the stakeholders via screencasts were used to resolve issues. An agile process was followed, and implementation was commenced at an early stage. Informal specifications were constructed, and testing used for validation.

Project 4: Document mining of the existing text requirements was used, together with exploratory prototyping to understand the requirements. The requirements were decomposed into separate mapping and refactoring scenarios, expressed in concrete grammar sketches, and then formalised in a UML/OCL specification. This was validated and refined by inspection and testing.

Project 5: Document mining of the existing text requirements was used, together with exploratory prototyping to understand the requirements. The requirements were categorised and prioritised. The functional requirements were decomposed into separate mapping scenarios. Client feedback via email and a forum enabled the refinement of these scenarios. The transformation was formalised in a UML/OCL specification. This was validated and refined by inspection and testing.

Project 6: Interviews and exploratory prototyping were used to elicit the requirements, followed by goal decomposition and then confirmation with the clients. The transformation was specified in UML. Testing was used for validation.

Project 7: Reverse-engineering of the existing transformation was used to obtain requirements for the revised transformation. In some cases it was difficult to identify if these were correct, and discussion with the customer was necessary. A logical specification of the new transformation was defined, which supported formal proof of correctness.

Table 2. Requirements engineering techniques in MT projects

Case	***Elicitation***	***Evaluation***	***Specification***	***Validation***
1	document mining, prototyping, interviews	informal conflict resolution	UML/OCL	inspection
2	brainstorming, interviews exploratory prototyping	impact analysis, negotiation	UML, graphs	testing
3	informal techniques, prototyping	negotiation	informal	testing
4	exploratory prototyping	scenario analysis	UML/OCL	testing, inspection, proof
5	exploratory prototyping	scenario analysis	UML/OCL	testing, inspection
6	exploratory prototyping	goal decomposition negotiation	UML/metamodelling	testing
7	reverse-engineering	goal decomposition	formal/logic	proof

Requirements change is a common occurrence during project development. This can be due to stakeholder's change of mind/circumstances or the introduction of some additional requirements to the existing one(s). Based on our study, we realised that transformation developers experienced similar events where they had to deal with requirements modifications, unrealistic requirements and conflict amongst the requirements.

> "Never do what you are told, and always do what is needed" (Study participant).

In Table 3, we have identified MT developer's responses when confronted with common problems that may occur during the MT development. As can be seen,

these revision activities generally require stakeholder interaction, or understanding of their real needs, and hence may be more difficult for MT projects where the project is embedded within a larger MDE project.

Table 3. Requirements revision in MT projects

Project	***Problem***	***Reaction, paraphrased from participant comments***
1, 2, 3, 4, 6, 7	Unrealistic requirements	-Implementing "what is needed" rather than what is wanted -Implementing "the underlying system"
1, 2, 3, 6, 7	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)
1, 2, 3, 4, 5	Requirements Conflict	-Resolving the conflict by common sense -Trade-off amongst the conflict requirements
2, 3, 4, 5, 6, 7	Requirements uncertainty	-Contacting the stakeholders for clarifications

6.2 Requirements Elicitation

According to our investigation, the requirements elicitation process in MT development often begins with 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 customer" (Study participant).

Customers often only have 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" (Study participant).

Therefore, transformation developers can suggest joint sessions with the stakeholders to be explicit 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.

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 developers. Not only are they in charge of implementation, but also eliciting the requirements are done by them as well.

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

> "It starts with customer, proof of concept than taking some code from the customer and presenting what can be done by prototyping, by a tool which provide analysis on code" (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 the model transformation projects that we analysed, as these help both developers and stakeholders to understand the problem space.

6.3 RE Techniques

There are several 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 could cover the entire RE process. Researchers have classified RE techniques and categorised them according to their characteristics. For instance, Hickey *et al* [3] proposed a selection model of elicitation techniques, Maiden *et al.* [5] came up with a framework that provide requirements acquisition's method and techniques. According to our study, in MT projects, RE techniques are selected and applied mainly based on personal preference, or on a company policy, rather than on the characteristics and specifications 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 projects. We have categorised RE techniques into groups of *human communication, process technique, knowledge development* and *requirements documentation.* Table 4 summarises the RE techniques that were used in the MT development projects. 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 selection criteria of the techniques are described by interviewees.

7 Outcome

In evaluating the outcomes of the MT projects, the development effort and problems encountered are considered, together with the degree to which the delivered transformation achieved the customer expectations. We use a qualitative five point scale (Very Low, Low, Moderate, High, Very High) for both factors based on the transformation size, business value and customer satisfaction. Table 5 summarises the outcomes of the different MT projects.

Table 4. RE techniques in MT projects

Category	***RE Technique***	**Project**	***Rationale***
Human communication	Online conference	1, 2, 3, 6	- Distribution of stakeholders - Lack of accessibility - Conveniency
	Brainstorming	1, 2, 3, 6	- Clarifying both stakeholders and developers to understand each other as well as the requirements
Process techniques	Joint requirements development session	2	- Resolving any possible issue which is not clear
	Categorisation	1, 2, 3, 4 5, 6, 7	- Identifying functional and non-functional requirements
Knowledge development	Prototype	1, 2, 3, 4, 5, 6, 7	- Receiving feedback based on the prototype -Informing the stakeholders from the progress
	Negotiation	2, 3, 6	- To prioritize the requirements - Trade-off
Requirement documentation	Diagram	1, 2, 3, 4, 5, 6, 7	- Providing a general view of the system
	Documentation	1, 2, 3, 4, 5, 6, 7	- Presenting the system formally - Providing a guidline for stakeholders

Project	***Transformation scale***	***Development cost***	***Customer satisfaction***
1	High	Moderate	High
2	Very High	Moderate	High
3	High	High: specifications too procedural, hard to analyse or modularise	Moderate
4	Low	Moderate	High
5	Moderate	Moderate	Moderate
6	High	High: complex and detailed semantics	Moderate
7	Moderate	Moderate	High

Table 5. Outcomes of MT projects

Of particular note are Project 2, which was the largest of the case studies in scale, with over 1500 transformation rules, and very large scale source data. This project also had the most systematic RE process, with good communication between the developers/analysts and the customers, and effective negotiation over requirements. There has been good acceptance of the project results by the customers, so we classify this as High satisfaction.

In contrast, Project 3 was also of large scale, but 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. There was a lack of systematic RE processes, and this led to high costs in reworking the translators when errors were discovered. The customer was unwilling to participate in any structured requirements engineering process.

Whilst Project 6 had a more systematic RE process than project 3, the semantic complexity of the target language and platforms caused the development effort and costs to be significantly higher than for other code generators. The complexity of the resulting generator has hindered its adoption, which has been limited. Thus, we give a rating of Moderate for customer acceptance in this case.

8 Conclusions and Future Work

In this paper, we have reported on the results of an exploratory study of requirements engineering for model-transformation development. We have reported on our initial findings from five semi-structured interviews with industrial experts in the field. Clearly, more research is needed, but some interesting points have already emerged from this study and are worth closer attention: First, we have been able to identify that model-transformation projects are typically individual projects that are embedded in wider software-development projects. We have briefly commented on how this impacts the identification of and communication with stakeholders in the transformation development. The projects we have discussed are almost exclusively greenfield projects, which is different from the wider software-development reality. This may be because model transformations are still a relatively young technology in industrial practice.

The interaction between the needs of the wider project and the highly technical nature of model-transformation development seems to have an impact on the requirements elicitation process in particular. We have seen that while prototyping and example-based generalisation seem to play an important role in understanding the requirements on model transformations, no more systematic process seems to be followed. Although developers apply some requirements engineering techniques in transformation projects this is often based on their experience and common sense as there is no specific requirements engineering process designed for model transformation development. At the moment, the focus of transformation development is mainly on the specification and implementation stages and the development team is responsible for all development process activities including the requirements engineering process.

More understanding of the context in which transformations are developed is required and we will, consequently, continue our empirical work in this area. In parallel, we have started work on defining a more systematic process for requirements engineering in the context of MT development [15].

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. 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.
3. 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.
4. 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.
5. NAM Maiden and Gordon Rugg. Acre: selecting methods for requirements acquisition. *Software Engineering Journal*, 11(3):183–192, 1996.
6. Tom Mens and Pieter Van Gorp. A taxonomy of model transformation. *Electronic Notes in Theoretical Computer Science*, 152:125–142, 2006.
7. 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.
8. Suzanne Robertson and James Robertson. *Mastering the Requirements Process (2Nd Edition)*. Addison-Wesley Professional, 2006.
9. Bran Selic. What will it take? a view on adoption of model-based methods in practice. *Software & Systems Modeling*, 11(4):513–526, 2012.
10. Shane Sendall and Wojtek Kozaczynski. Model transformation: The heart and soul of model-driven software development. *IEEE Software*, 20(5):42–45, September.
11. Ian Sommerville and Gerald Kotonya. *Requirements engineering: processes and techniques*. John Wiley & Sons, Inc., 1998.
12. Professor Ian Sommerville. private communication, 2015.
13. Axel van Lamsweerde. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. John Wiley & Sons, 9 Jan 2009.
14. Jon Whittle, John Hutchinson, and Mark Rouncefield. The state of practice in model-driven engineering. *Software, IEEE*, 31(3):79–85, 2014.
15. Sobhan Yassipour Tehrani and Kevin Charles Lano. Model transformation applications from requirements engineering perspective. In *The 10th International Conference on Software Engineering Advances*, 2015.