
# Use, potential, and showstoppers of models in automotive requirements engineering

Software & Systems Modeling (2019) 18:2587–2607
https://doi.org/10.1007/s10270-018-0683-4

REGULAR PAPER

# Use, potential, and showstoppers of models in automotive requirements engineering

Grischa Liebel[1] · Matthias Tichy[2] · Eric Knauss[1]

Received: 20 September 2017 / Revised: 29 March 2018 / Accepted: 15 May 2018 / Published online: 26 May 2018


**Abstract**
Several studies report that the use of model-centric methods in the automotive domain is widespread and offers several benefits. However, existing work indicates that few modelling frameworks explicitly include requirements engineering (RE), and that natural language descriptions are still the status quo in RE. Therefore, we aim to increase the understanding of current and potential future use of models in RE, with respect to the automotive domain. In this paper, we report our findings from a multiple-case study with two automotive companies, collecting interview data from 14 practitioners. Our results show that models are used for a variety of different purposes during RE in the automotive domain, e.g. to improve communication and to handle complexity. However, these models are often used in an unsystematic fashion and restricted to few experts. A more widespread use of models is prevented by various challenges, most of which align with existing work on model use in a general sense. Furthermore, our results indicate that there are many potential benefits associated with future use of models during RE. Interestingly, existing research does not align well with several of the proposed use cases, e.g. restricting the use of models to informal notations for communication purposes. Based on our findings, we recommend a stronger focus on informal modelling and on using models for multi-disciplinary environments. Additionally, we see the need for future work in the area of model use, i.e. information extraction from models by non-expert modellers.

**Keywords** Modelling · MDE · MBE · Requirements engineering · Empirical research · Case study · Automotive

## 1 Introduction

Empirical evidence suggests that Model-Based Engineering (MBE) offers benefits such as quality and productivity improvements [4,40]. Additionally, negative aspects such as insufficient tool support [4,40,41] and the use of MBE with legacy software [28,40] are reported. For the embedded domain in particular, related work reports benefits such as cost savings [29], productivity increases [1], or increases in reusability [31].

So far, research regarding the state of practice of MBE has targeted the entire software engineering cycle, from requirements engineering (RE) to software development and testing, both in terms of quantitative studies [1,18,27,28,30,31,51] and qualitative studies [4,26,29,41,56]. However, these studies typically do not investigate in detail how models are used in the respective phases of software and systems engineering. Additionally, Loniewski et al. [33] point out that only few model-driven approaches explicitly include RE.

Several studies have reported on the state of practice in RE in the embedded domain, e.g. [10,12,25,50]. According to Broy [12], one of the big challenges in the automotive industry is RE. Already in 2000 Houdek [25] states that the current way of specifying requirements in a largely textual way is insufficient. Despite these statements, Braun et al. [10] state that requirements are still elicited and specified in an ad hoc manner and typically in natural language. Sikora et al. [50] report from a case study with industry practitioners

Communicated by Dr Jeff Gray.

✉ Grischa Liebel
grischa@chalmers.se

Matthias Tichy
matthias.tichy@uni-ulm.de

Eric Knauss
eric.knauss@cse.gu.se

[1] Software Engineering Division, Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden

[2] Institute of Software Engineering, University of Ulm, Ulm, Germany

that models could be used to tackle several RE challenges in the embedded domain, e.g. the increasing complexity. Several proposed modelling frameworks that prescribe or encourage the use of models for RE have been proposed, e.g. the SPES 2020 Methodology [46] and REMsES [10], and in a few studies evaluated with practitioners, e.g. in [8] and [11]. However, the industrial uptake seems to be limited.

In summary, there are several benefits and challenges typically associated with the use of models in industrial software and systems engineering. However, the use of models for RE is despite the proposal of several frameworks and potential benefits stated by practitioners seemingly uncommon in the embedded domain. Therefore, in this paper we investigate empirically in the automotive domain how models are already used during RE, what potential practitioners see in the use of models during RE, and which roadblocks exist in this area. The aim of this paper is not to invalidate existing solution proposals that use models in RE. Instead we aim to complement existing work and deepen the understanding of how models could be used during RE and which challenges need to be tackled to do so. To reach this goal, we aim to answer the following research questions:

RQ1: How and why are models currently used during automotive RE?
RQ2: What is the perceived potential of models in automotive RE?
RQ3: Why are models not used during automotive RE?

We conducted a multiple-case study at two automotive companies, one Original Equipment Manufacturer (OEM) and one supplier, collecting qualitative data from 14 semi-structured interviews. Our findings are that models are already widely used in automotive RE, but typically only for very specific purposes or by expert users. Furthermore, many semi-formal or informal notations are used, e.g. sketches of state machines to improve understanding. There is a disagreement among our interviewees regarding the potential of models in automotive RE. While some interviewees consider it beneficial to have formal models for purposes such as requirements validation or verification, others do not believe that formalisation to such a high degree is feasible and prefer semi-formal notations, e.g. for communication purposes only. Finally, we present several reasons for the limited adoption of models in automotive RE, such as tool restrictions, too high complexity, or organisational reasons.

The data from the case study reported in this paper was used in another paper [32], focusing on the topic of communication and coordination challenges in RE. As the two analysis topics are orthogonal to each other, we decided to report them separately.

The remainder of this article is structured as follows. In Sect. 2, we present work related to empirical studies MBE uptake, RE in the embedded domain, and models during RE. The method we applied is presented in Sect. 3, together with the threats to validity. The case companies at which we conducted the study are described in Sect. 4. Results are presented in Sect. 5, followed by a discussion of the results in relation to existing work in Sect. 6. The paper is concluded in Sect. 7.

## 2 Related work

In recent years, a substantial body of knowledge regarding the use of models and of model-based engineering in industry has developed. Among the published studies, those addressing the embedded systems domain in particular are, however, scarce. For RE specifically, several processes and framework that focus on or include the use of models have been developed. In the following, we discuss these areas of research and their relation to this paper in detail. We take an inclusive approach, trying to cover a large variety of research related to our study and not limited to papers that have similar contributions to our work.

### 2.1 RE in the embedded domain

Several studies have been published during early 2000 on how RE, without a focus on models, is performed in the embedded domain, specifically studies from the automotive domain conducted by researchers associated with Daimler and the Technical University of Munich [12,24,25,54].

Broy [12] states that RE in the automotive domain is mainly performed by the OEM, with only smaller parts added by suppliers. Heumesser and Houdek [24] confirm this aspect and add that automotive specifications have to be detailed and extensive, as the suppliers typically do the implementation.

Houdek and Pohl [25] report piloting efforts of RE processes in different areas of DaimlerChrysler. The authors state that textual requirements are insufficient and that a common meta-model for requirements management and tool support for this meta-model is needed.

A study focussed on the embedded industry by Sikora et al. [50] suggests that requirements specified in natural language are not satisfactory and that the use of models during RE "would significantly improve requirement validation and verification". The study presents empirical data from a questionnaire and interviews with industry representatives participating in the SPES project. In contrast to our study, the authors focus on how RE is performed and how to address several challenges during RE, e.g. how to address high complexity during RE.

## 2.2 Modelling in industry

Several empirical studies exist that focus on the use of models in industry, e.g. [4,26,28,31,40,41,56].

In an early qualitative study at Motorola, Baker et al. [4] report on the use of MBE at Motorola over a period of 20 years. The authors report defect reductions and increases in productivity, but also a lack of tools and tool interoperability, poor performance of generated code, and a lack of scalability.

A literature review on the industrial application of MBE by Mohagheghi and Dehlen [40] reports quality improvements, but a disagreement in the literature on the effects on productivity. The authors name tool integration costs, complexity of creating models and tool maturity as drawbacks of MBE.

Experiences from three European companies with developing techniques and tools for MBE are presented by Mohagheghi et al. [41] in terms of a qualitative study. In line with previous studies, the authors report tool problems and complexity of large models as drawbacks of MBE. In terms of advantages, they list simulation and testing support as well as decision support through MBE.

In a series of studies, Hutchinson et al. study the use and adoption of MBE in industry [26–28,56]. Based on interviews at a printer company, a car company, and a telecommunications company, Hutchinson et al. [26] report that a successful adoption of MBE seems to require an iterative and progressive approach, organisational commitment, and motivated users. The authors complement their qualitative data with quantitative data in [27], and point out that MBE should be placed in the organisation context, not only focusing on technical issues. A similar argument is made in [56], where the authors report that tooling is a major challenge in MBE, but organisation issues need to be considered as well. Finally the authors investigate the use of MBE in multiple domains in [28], using more than 250 survey responses, 22 interviews, and observational studies. The authors report that significant additional training is needed for the use of MBE. From their interviews, the authors conclude that especially the people's ability to think abstractly seems to have significant impact on their ability to create models. All four publications consider software engineering in general, without a specific focus on a single industry or software engineering phase/activity.

Based on a survey of 113 practitioners by Forward and Lethbridge [18], model-centric approaches commonly suffer from model inconsistencies over time, challenging model interchange between tools, and heavyweight tools. In contrast, the authors report that a purely code-centric development can make it difficult to see the overall design and understand the behaviour of a system.

Limited to the Italian software industry, Torchiano et al. [51] report on their findings from a survey with 155 subjects. Of the 32% who do not use models, a large proportion states that modelling requires too much effort or is not useful enough.

Finally, Gorschek et al. [21] report on a survey with the aim to understand to what extent models are used in industry, not limiting themselves only to the population of subjects who use models. The authors report that the use of models is generally low in software engineering, even in areas such as embedded systems, and mainly limited to sketches for communication and coordination.

## 2.3 Modelling in the embedded domain and during RE

Regarding embedded systems, the related work on the use of MBE is substantially slimmer. We are aware of only three studies that investigate the use of MBE in the embedded systems domain, i.e. the studies by Agner et al. [1], Kirsten and Zimmermann [29], and our own study [30,31].

Agner et al. [1] survey the use of model-driven techniques in Brazilian embedded software development. Similarly to studies that investigate MBE in general, the authors report increases in productivity and quality improvements. The authors also state that there is little use of code generation.

Kirstan and Zimmermann [29] collect qualitative data through interviews on the use of MBE in the automotive domain, reporting earlier error detection, a higher degree of automation and cost savings during the initial development phases. As negative aspects, the authors report the complexity of large functional models and difficulties with tool interoperability.

Our own study [30,31] extends the body of knowledge by collecting quantitative data on the use of MBE in the European embedded systems domain. Based on 113 surveys from industry practitioners, we show that MBE is used by practitioners in a wide range of roles and phases of embedded systems engineering. Particularly, with regard to the focus of this paper, many of the participants work in the area of RE.

Recently, Haghighatkhah et al. [22] reviewed the literature on automotive SE. The authors find 33 papers covering RE in the automotive domain. Based on that work, they argue that the automotive industry "should adopt model-based RE." [22]. However, they back this argument using opinion papers only.

## 2.4 Frameworks for using models during RE

Regarding RE in particular, several frameworks and processes have been proposed that advocate or prescribe the use of models, mainly by researchers associated with Technical

University of Munich and the Ruhr Institute for Software Technology.

Pohl et al. [46] introduce the SPES 2020 Methodology for the development of embedded systems, which prescribes the use of models during all development phases. During RE in particular, the framework suggests a separation between *solution-independent* and *solution-oriented* diagrams. Examples of the used models include context models, goal models, and scenario models. The authors state that while model-based approaches are adopted in embedded systems engineering, they "often fail due to the lack of sufficiently powerful modelling theories and missing integration of theories, methods, and tools.".

With the aim to bridge software and systems engineering, Böhm et al. [7] map the SPES framework to existing standards, i.e. to ISO/IEC 15288 and ISO/IEC 12207. The authors describe which views and artefact types are related to which elements in the standards.

Practical experiences with SPES are reported in [8,11]. In [8], Böhm et al. present their experiences with SPES in an industrial project at Siemens. The authors apply SPES to a mature, already running train control system, using a specification of "high quality". Findings are that "the high quality of input documents, and cooperation with product experts were considered the most influential success factors". Brings et al. [11] discuss experiences of using SPES in the area of cyber-physical systems. The authors report that they "identified problems resulting from an increased number of dependencies." and "the need to cope with redundancies caused by properties which are system as well as context properties in a structured manner".

Apart from the SPES framework, there exist several proposed processes and frameworks for requirements modelling, e.g. [6,9,10,17,53].

Vogelsang et al. [53] propose a process to model requirements and architecture in parallel, starting from informal use cases. In particular, the authors propose to create textual use cases, textual scenarios and Message Sequence Charts (MSCs). The process is evaluated with 15 master students, concluding that it is feasible for detecting inconsistencies early.

Brandstetter et al. [9] present a process to perform early validation of requirements by means of simulation, using the control software of a desalination plant as an industrial case. Assumptions and properties of the system are captured in context models. Furthermore, textual validation use cases are created and refined into MSCs. Finally, the models are made executable, e.g. by implementation in Simulink. Experiences of using the approach in an industrial case are discussed, but details on the execution of the use case are largely missing.

Resulting from a research project with academic and industrial partners, Braun et al. [10] propose the use of model-based documentation. For RE, these include goal models, scenario models and function models. Behaviour models are included under design models. To our knowledge, the approach has only been evaluated within a research project [45] and not in terms of an empirical study.

The interchangeable use of controlled natural language and formal models by means of a bidirectional graph transformation is proposed by Fockel and Holtmann [17]. However, empirical data on the application of the approach is to our knowledge not existing.

Berenbach et al. [6] list several requirements they consider essential for a requirements modelling language, such as distinction between process and use case modelling. The authors argue that using UML for requirements modelling has proven to be frustrating. They then proceed to propose a language that fulfils their requirements, the Unified Requirements Modelling Language (URML), based on a UML profile. URML is piloted in one commercial project at Siemens, showing that the proposed concepts are useful.

Outside the research scope, the Handbook of Requirements Modelling According to the IREB Standard [52] lists several options of doing requirements models. No empirical data is used to argue why models should be used or to evaluate the feasibility of the proposal. Arguably, this is outside the scope of such a handbook.

Finally, there is substantial research in the area of structured language or controlled natural language (CNL), mainly in the area of program verification, e.g. [2,15]. While most of the research in this area focuses on specifications on implementation level, there are also approaches to express requirements in CNLs, e.g. [14,19,34]. The reasons for introducing CNLs are varied, e.g. requirements verification [14], inconsistency detection [19], or change impact analysis [34].

While not explicitly model-driven, models also play a central role in Méndez Fernández's work on artefact-oriented RE [36–39].

In [37], Méndez Fernández et al. propose that in RE, there should be a focus on the results, the artefacts that are produced during RE instead of the process. The authors state that "the establishment of domain specific concept models has become a wide accepted technique, mostly arising from the benefit of enabling seamless modelling". As a practical contribution, the authors propose a meta-model for artefact orientation, created from experiences in two industrial collaborations. The applicability of this model is then further studied in [36].

In [39], Méndez Fernández and Wieringa propose a specific way to use artefact-oriented RE using an empirical cycle, similar to the design science cycle. As a part of this method, the authors state that "The first step for companies towards good RE is to establish an RE reference model, [..]".

Finally, Méndez Fernández et al. [38] study how RE is executed in 12 successful[1] real-life projects at CapGemini.

[1] Released and deployed systems, used in production.

In line with their notion of artefact orientation, the authors analyse which artefacts are produced and why, "instead of exclusively focusing on how it was produced". The results show that domain and environment models are present in all 12 projects. It is important to note that behaviour models are considered design artefacts by the authors, not requirements artefacts.

Focusing on verification only, models have been shown to enhance the testing process in what has been coined Model-Based Testing [5]. However, it is also common in this area of research to focus on solution proposals, while evaluation or validation research is scarce [5].

### 2.5 TwinPeaks and architecture models

Nuseibeh [43] established in 2001 the notion of the TwinPeaks model, in which requirements and architecture are treated as equal and intertwined. He proposes to develop requirements and architecture iteratively, as they would inform each other. Based on this paper, the TwinPeaks Workshop series was established, in which several publications propose the use of models. For instance, authors propose the use of architectural trace models [23] or formalised requirements in Simulink [48] to bridge the gap between requirements and architecture.

Outside of the TwinPeaks Workshop series, several authors have made the argument that requirements and architecture should or cannot be separated [3,16,44]. Paech et al. [44] argue that functional and non-functional requirements, as well as architecture need to be considered together, as they typically influence each other. Similarly, Avgeriou et al. [3] state that "Although research in both requirements engineering and software architecture is quite active, it is in their combination that understanding is most needed and actively sought." Finally, Eliasson et al. [16] provide experiences from the automotive industry, stating that, in practice RE does not always come before design or architecture.

Therefore, while we focus on RE in this paper, it is important to consider that practitioners might see a strong connection between RE and architecture, or even consider them the same thing to some extent. Furthermore, due to this strong connection, challenges or solution proposals from the area of architecture might apply to the area of automotive RE as well.

### 2.6 Summary

In summary, there is a large body of knowledge of how models are used in industry and solution proposals of how models can be used during RE, including some empirical evaluations. However, there is a lack of empirical data on how exactly models are already used during RE in the embedded domain, what perceived benefits and drawbacks of using models for RE are, and which prospective use practitioners see for models in RE.

To our knowledge, the only paper that empirically investigates how RE challenges in the embedded domain can be solved using models is [50]. While a number of surveys on the use of models in industry exist, the nature of surveys is to provide an overview instead of an in-depth analysis [47]. In this paper, we complement this work in the sense that we (a) provide additional, in-depth qualitative data from two cases in the automotive domain, and (b) focus on the use of models, not on how to address existing RE challenges, as it is the case in [50].

## 3 Research methodology

As outlined in the previous section, there is a gap of knowledge with respect to understanding the role of models in automotive RE. Hence, the objective of this study is to reduce this gap by gaining insight into the current practice and the potential of using models in automotive RE. From this objective, we derive our three research questions, targeting the current state (RQ1), the perceived potential (RQ2), and the obstacles (RQ3) of modelling during RE.

We used a multiple-case study design, following a four-step process consisting of study design, data collection, data analysis, and reporting. The first three steps are depicted in Fig. 1 and discussed in more detail in the following.

Case studies are suitable when a real-life phenomenon is investigated in its context, investigating few instances [47]. Case studies aim to provide an in-depth analysis of an existing situation in its context. As such, case studies differ from

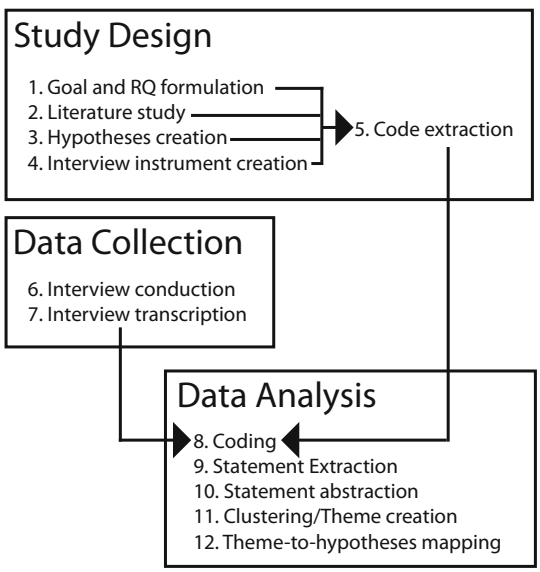


**Fig. 1** Case study process

**Table 1** Hypotheses from related work

| Nr. | Description | RQs | References |
|---|---|---|---|
| **H1** | Requirements are mainly specified in natural language | 1 | [45] |
| **H2** | Models are already used in automotive RE | 1 | [30,31] |
| **H3** | Using models during Requirements Engineering could significantly improve requirements validation and verification | 2 | [50] |
| **H4** | There is insufficient tool support for modelling requirements | 3 | [4,30,40,41] |
| **H5** | Using UML for modelling requirements is deficient and frustrating | 3 | [6,20] |
| **H6** | Requirements Engineers lack guidance on how to use models within Requirements Engineering | 3 | [45,50] |
| **H7** | The main problem in automotive RE is finding the right level of abstraction | 3 | [45] |

controlled experiments in the sense that there is a lack of control in favour of realism, and from surveys in the sense that a case study provides an in-depth analysis instead of an overview picture [47]. Given that the related work heavily focuses on surveys, we therefore chose the case study strategy to complement the existing body of work.

### 3.1 Study design

After defining the goals and RQs, and studying the related work regarding models in RE (Steps 1 and 2 in Fig. 1), we selected two automotive companies as our cases. We decided to select one OEM (Company A) and one supplier (Company B) from the automotive domain as representatives for two typical cases within this domain. The cases were selected based on existing industrial contacts. The companies are described in detail in Sect. 4.

Our case study is exploratory in nature [47]. Hence, it uses an inductive approach without a specified theory at the beginning of the study and is instead guided by seven hypotheses (Table 1), which we derived from an initial literature review. These hypotheses present a limited set of statements made in relation to our research questions (Step 3 in Fig. 1) and allow for the exploration of the study topic in an exploratory manner, without limiting the scope.

We used a semi-structured interview guide for data collection, consisting of 5 demographic questions and 19 questions targeting the research questions. We created the interview guide based on our hypotheses, maintaining links between the interview questions and the hypotheses (Step 4 in Fig. 1). The interview guide is listed in "Appendix A", followed by a table showing how our hypotheses are linked to interview questions in "Appendix B".

Note that during the interviews, we did neither define exactly what we understood as a model, nor what comprises RE and different requirements abstraction levels according to our understanding. This was done deliberately to allow for a wide coverage of topics and insights in the interviewees' own understanding of these terms, especially as RE, architecture and design-related activities are often intertwined in the automotive industry [16].

Finally, we created a number of a-priori codes based on the first four steps (see "Appendix C" for the complete list). These a-priori codes are on a high level of abstraction to allow for a broad categorisation of the interview data in the first round of coding.

To ensure quality in our study, we followed several of the guidelines proposed by Runeson et al. [47]. One researcher not involved in the actual study reviewed the study design and suggested improvements. Furthermore, the interview instrument was discussed and refined in several iterations among the authors. Throughout the study, we maintained a case study protocol to keep track of the progress and important decisions.

### 3.2 Data collection

We selected 15 interviewees through a contact person at each company, based on their own contacts within the companies. In Company A, we aimed for a variety in roles, as all our contacts were located in the same department, focusing on embedded software engineering (EmbSE). We therefore selected 4 requirements engineers, 3 software engineers and 2 verification engineers, in order to include interviewees who write requirements as well as interviewees who receive requirements. In Company B, we covered two different areas, namely systems engineering (SysE) and application software engineering (AppSE). Therefore, we tried to select interviewees who had an overview of the areas and insights into RE. That is, all of them had either extensive experience and/or specialist roles at the case company. With the help of our contact person, we selected 3 employees from SysE and 3 from AppSE.

A summary of the areas and the interviewees' work experience is summarised in Table 2. Work experience denotes the experience in years in the interviewees' area at the case company. Note that interview A5 was conducted with two interviewees with 3 and 10 years of experience.

All interviews were conducted by the first author of this paper (Step 6 in Fig. 1), but the second author assisted in several of them. The interview time ranged between 37 and 72 minutes.

All interviews were transcribed verbatim, anonymised and sent out to the interviewees for review (Step 7 in Fig. 1).

**Table 2** Interviewees with experience in years

| Area | Interviewees |
|---|---|
| Embedded software engineering | A1(15), A2(4), A3(15), A4(3.5), A5(3, 10), A6(7), A7(10), A8(< 1) |
| Systems engineering | B1(3.5), B2(1.5), B6(4.5) |
| Application software engineering | B3(36), B4(27), B5(4) |

## 3.3 Data analysis

We used a mix of qualitative content analysis [35] and the 8-step coding procedure proposed by Creswell [13] for data analysis.

In the first step, we assigned a-priori codes from our study design to all transcripts (Step 8 in Fig. 1), using the open-source qualitative analysis tool TAMSAnalyzer [55]. During this coding procedure, we applied instances of the 24 codes to the transcripts, where each unit of classification [35] was assigned one or many code instances. Essentially, using this procedure, each unit of classification made by an interviewee is enhanced with keywords (the codes). A unit of classification is in our case a sentence, part of a sentence or multiple sentences that can be summarised in a single statement up to one sentence. For example, if an interviewee talked about how a notation is used to create models of a requirement, we would assign the codes *Modelling* and *Notation*.

In the next step (Step 9 in Fig. 1), we extracted *relevant* units of classification using a search function. In this study, we decided that all units of classification that were either annotated using the code *Modelling* or the code *Notation* are relevant. The extraction yielded 154 units of classification from Company A and 124 from Company B.

We then summarised each unit of classification into an abstract statement, a summarised description of the unit (Step 10 in Fig. 1). This step was followed by inductive category development [35], grouping the statements into clusters or themes (Step 11 in Fig. 1). This process was done in iterations, discussing the outcome among the study authors.

For presentation in this paper, we only selected those themes that were supported by more than one interviewee, in order to ensure data triangulation. Finally, we mapped the resulting themes to our initial hypotheses and RQs.

## 3.4 Validity threats

In the following, we describe the threats to validity of this study and measures we took to avoid or reduce them We use the categorisation by Runeson et al. [47] into *Construct Validity, Internal Validity, External Validity and Reliability* .

### 3.4.1 Construct validity

Construct validity describes whether the constructs used in the study, e.g. terms and definitions, are interpreted in the same way by the researchers, the interviewees and other people involved in the study.

The interview guide was designed by the first author of this paper, which he then discussed with the second author and one additional researcher not involved in the study. Afterwards, our industry contacts at both case companies received the questionnaire for another round of review.

Overall, we tried to avoid ambiguities with respect to terms and definitions, e.g. by adding definitions to the interview guide. Similarly, we tried to identify and improve questions that could be considered suggestive.

Interviewees participated on a voluntary basis and did not receive any form of compensation. They were granted anonymity and had the option to review their transcript before it was shared or discussed with the contact persons (*Member checking*). Hence, we expected that they answered all questions truthfully and did not avoid certain topics.

### 3.4.2 Internal validity

Internal validity reflects whether all causal relationships have been examined or if unknown factors remain that might affect the analysis outcome. Generally, internal validity is high for case studies, since there is no artificial environment and the real-life context is taken into account.

In order to avoid single interviewees having a too large impact, we used data *triangulation*, using only statements expressed by multiple interviewees. At Company A, we interviewed multiple roles and multiple people in each role in order to get a cross-cutting picture of the problems. While we did not have multiple interviewees for each role in Company B, we still used data triangulation in order to extract common problems.

In order to ensure continuity in the data collection process, all interviews were conducted by the first author using the same interview guide. However, due to the semi-structured nature of the interviews, different follow-up questions were asked depending on the context. This is a potential threat to internal validity, since the question order might have influenced the answers. We try to lower this threat through member checking, as interviewees got the chance to re-read their answers and comment/clarify.

The selection of interviewees poses a threat to internal validity, as one contact person at each company made the selection. To address this, participation was voluntary and

invitations were sent to a larger sample of potential interviewees.

Finally, the three areas of EmbSE, SysE, and AppSE from which we recruited could have an impact on the results. For example, the recent change to an agile process in the case of AppSE can clearly be seen as an influence factor. This might have lead our interviewees in that area to focus on specific parts of their development process that are affected most by the change. Given that organisations of similar size as our case companies are in constant change, we accept this threat and see it as a means to obtain data from diverse situations, similar to what Yin [57] describes as *revelatory cases*.

#### 3.4.3 External validity

External validity concerns to which degree the results of the study can be generalised to a broader context, beyond the studied cases. The external validity of case studies is low by design. Therefore, we cannot claim that our findings are indeed general to the automotive domain (or beyond). Instead, a case study offers detailed information of the studied topic in a real-life context and a high internal validity (in contrast to, e.g. surveys). As most existing empirical studies targeted towards MBE are surveys, we decided to complement this existing body of work using a case study design, accepting the low external validity in exchange for a high internal validity.

To some extent, the external validity is increased through the use of triangulation, member checking, and the use of hypotheses from related work (as these are in many cases based on empirical surveys). Therefore, while we cannot claim generality, we expect that parts of our findings extent to similar contexts, i.e. parts of automotive projects in which the context resembles our two cases.

#### 3.4.4 Reliability

Reliability describes to what other researchers would arrive at the same conclusions when replicating the study with the same design.

With respect to the case study design, we aimed at reducing reliability threats as much as possible by reviewing the design and the interview instrument. Similarly, we discussed the code set used for analysis of the interview data among the three first authors. Finally, the interview instrument is appended to this paper to allow for replication.

The interviews were transcribed verbatim by the first author in order to avoid subjective judgement. However, the abstraction and categorisation of the coded statements is to some extent subjective. We tried to address this by discussing the resulting analysis with our contact persons at both companies.

**Comp. A, Embedded software development (EmbSE)**
- Input: Vague high-level requirements
- Requirements breakdown to very detailed level
- Specification handover to suppliers and test organisation
- Requirements evolve and are rarely written from scratch

**Comp. B, Systems Engineering (SysE)**
- Input: User Requirements Specification (URS)
- Process related to waterfall
- Creation of System Requirements Specification (SRS) based on URS

**Comp. B, Application Software Engineering (AppSE)**
- Input: URS, similar to SysE
- Agile process, recently introduced
- Based on URS, creation of business stories, epics, features and user stories

**Fig. 2** Summary of the case companies' RE-related processes

## 4 Case companies

We conducted the presented case study at two case companies. Their processes related to RE are summarised in Fig. 2. In the following, the cases are described in more detail.

### 4.1 Company A

Company A is a global automotive OEM based in Sweden. In this company, we performed all interviews in a single department, where a large part of the embedded software development for vehicles takes place.

Within the studied department, high-level requirements for each project usually come from product planning, which is outside of the department. These requirements are usually user-oriented and often vague, which is partially intentional in order to leave room for creativity during implementation.

In the department, the high-level requirements are broken down into smaller parts and assigned to employees whose functions are affected. The employees break down the requirements into logical components. These are on a very detailed level, often close to pseudo-code. The resulting component specifications are then handed over to in-house development or used as a contracting document for external suppliers.

As soon as the requirements are broken down, the department's test organisation is starting to prepare the verification activities in parallel with development. This includes, e.g. to write test cases or prepare models of the environment for testing purposes. The models and tests are developed independently of the source code for the actual software.

The overall vehicle specification is usually kept and evolved throughout projects. That is, specifications are mod-

ified over multiple projects and not written entirely from scratch.

### 4.2 Company B

Company B is an Austria-based automotive supplier developing power train systems, as well as simulation and test bed systems. The company has both market-driven projects and customer projects, where the requirements come directly from an OEM. The process is different for the two different parts of the company that were studied, *SysE* and *AppSE*.

In *SysE*, the development follows a process related to the waterfall model, with smaller differences between the departments and units. Generally, projects start with a User Requirements Specification (URS) coming in from different sources. The URS is analysed in the company by the customer relations unit and a System Requirements Specification (SRS) is elicited to meet the URS requirements. The level of detail and the quality of the received URS varies depending on the customer. However, the customers generally already have experience with the products offered by the company and know their own use cases.

In *AppSE*, a waterfall process was followed for a long time. It has recently been replaced by an agile process following the SAFe framework [49]. The interviewees were positive towards this newly introduced process, as far as they could already judge it. Especially the fact that communication had improved significantly was mentioned several times. However, it also means that experiences from this new process are preliminary.

An incoming URS is first handled by the customer relations unit and then handed over to product management, if the current product range does not already fulfil the request. Product management has the task to understand the problem of the customer, with the help of domain experts and fit it to the company's current product range. In particular, an incoming URS is often covering multiple products, as the customers only describe their problems and needs, which do not necessarily adhere to single products. From the received URS, an SRS is derived, usually together with the stakeholders. For customer projects, the SRS is then used as a contracting document.

Initially, so-called Business Stories are specified and then refined or broken down. In the first step, requirements are broken down into Epics. The next level of granularity is the Feature, which should fit into one iteration of 10 weeks. Features are picked by the development teams, consisting of developers and the Development Owner. The teams then take ownership of the Features and break them down into User Stories, which can be implemented in a single two-week sprint. Acceptance criteria are written by the receiving party for all levels of granularity.

## 5 Results

In the following we will present our results with respect to the three research questions. Whenever we use quotes, we identify the interviewee and the area according to Table 2.

### 5.1 RQ1: How and why are models currently used in automotive RE?

We break down our findings regarding the current use of models in RE (RQ1) into the nature of existing models in RE (summarised in Table 3) and the purpose of these models (summarised in Table 4).

#### 5.1.1 Nature of models in RE

We find numerous different models that are existing today, reflecting the heterogeneous nature of large automotive projects and the variety of our two case companies and three application areas. An overview over the main themes we extracted from the interview data is given in Table 3.

Supporting the findings of many publications in the area of embedded systems, e.g. [24,25,50], multiple interviewees from both companies stated that **textual requirements are the norm (T1.1.1)** in most areas and that only few models exist. On a high level of abstraction, and outside of software development, these textual requirements are typically stored in Word or Powerpoint documents. On lower levels of abstraction, closer to architecture and software development,

**Table 3** Model nature during RE: support for different themes by area

| ID | Model Nature | EmbSys | SysE | AppSE |
|---|---|---|---|---|
| T1.1.1 | Mainly textual requirements | 5/8 | 2/3 | 3/3 |
| T1.1.2 | Emerging specification structure | 5/8 | 1/3 | 1/3 |
| T1.1.3 | Templates | 2/8 | 1/3 | 3/3 |
| T1.1.4 | Sketches of behaviour | 5/8 | 1/3 | 3/3 |
| T1.1.5 | Formal structural models | 3/8 | 2/3 | 0/3 |

**Table 4** Purposes of models during RE: support for different themes by area

| ID | Purpose | EmbSys | SysE | AppSE |
|---|---|---|---|---|
| T1.2.1 | V&V | 5/8 | 0/3 | 0/3 |
| T1.2.2 | Communication | 1/8 | 1/3 | 1/3 |
| T1.2.3 | Guidance and streamlining | 3/8 | 2/3 | 3/3 |
| T1.2.4 | Handling complexity | 5/8 | 0/3 | 2/3 |

specific tools are used that allow for advanced features, such as versioning or traceability.

While the requirements themselves might be textual, several interviewees stated that the **entire specification is an emerging structural model (T1.1.2)** due to existing trace links. That is, the model is not actively modelled by someone, but emerges through creation of traces. In the case of Company A, a systems engineering tool is used that supports different element types, e.g. requirements and logical design components, and relationships between them, e.g. a requirement is fulfilled by a design component, in the form of a customisable meta-model. Similarly, in Company B, there was an ongoing initiative at the time of the interviews to move to model-based documentation in the form of SysML. In SysML, requirements are a specific type that contains a textual description, i.e. while the requirements themselves remain textual, they can be linked via defined relationships.

As a step towards formalisation, interviewees mentioned the **use of templates (T1.1.3)**, keywords, or sentence patterns in order to describe requirements. However, in most cases the interviewees stated that these templates are either not enforced or had already been abandoned. The reason for abandoning these templates were often stated to be that their meaning deteriorated over time or that they were simply not followed.

> [..] there were a couple of terms specified that were not followed and, due to that, (it) has broken down in what it actually means, [..] it has broken down in principle.—A7, EmbSE

In both companies, interviewees mentioned that **sketches of behaviour are common (T1.1.4)**. For example, one interviewee at Company A stated:

> And sometimes when we have pictures, it's mostly like state diagrams and so on. So you say 'OK, A, B, C, and what are the transitions between that?'.—A4 EmbSE

Similarly, several interviewees at Company B also stated that sketches of state machines or flow diagrams are not uncommon. In particular, two interviewees from SysE stated that diagrams of the control flow or other behaviour diagrams close to implementation are widespread.

**Formal structural models describing the system (T1.1.5)** are used in both companies, e.g. to define the logical architecture with software and hardware components, and their connections. While these models are primarily architecture models, they are often connected to RE, e.g. as a basis for structuring the requirements specification or as building blocks that are used to define behaviour requirements.

Finally, multiple interviewees in both companies stated that **some formal models describing requirements exist**, e.g. in the form of UML models. However, these are typically used by experts or proponents of modelling only.

> But it (the model) is not fully used. This is here the problem. Some use it, some gurus.—B4, AppSE

An exception is SysE, in which formal diagrams are commonly drawn for safety-critical applications.

Beyond the current practice, there are **ongoing attempts to introduce formal models for RE** at both companies. At Company A, there are initiatives to entirely abolish textual specifications on lower levels of requirements abstraction in favour of Simulink. At Company B, there are piloting efforts to introduce model-based systems engineering using SysML in parts of the company.

#### 5.1.2 Purpose of models in RE

Similar to the variety of different models existing in the two case companies, our interviewees named multiple different purposes to use models during RE. An overview over the main themes we extracted from the interview data is given in Table 4.

At Company A, requirements models were named by five different people as a **support to verification activities (T1.2.1)** later on, e.g. in terms of sketches of behaviour.

> [..] it's used for test and verification. I mean those pictures, it's not just for writing down a specification and have it for later use when we do a new model, it's used later on also to verify that we have gotten the right things.—A5, EmbSE

For semi-formal and informal notations, interviewees mentioned models as a tool to **support communication (T1.2.2)**.

> Our focus is clearly on cross-domain communication. Definitely.—B2, SysE

Additionally, several interviewees brought up formalisation as a tool to **guide and streamline RE (T1.2.3)**. For example, in Company B one interviewee mentioned a defined vocabulary and process as a way to guide engineers.

> [..] it helps [..] that you concentrate on 'What is the function I want to fulfil? What is the behaviour [..]?' And that contributes to a lot of thoughts that you would never have if you would simply write it down in a Word document.—B1, SysE

Finally, several interviewees mentioned that different kinds of models are currently used as a way to **handle increasing complexity (T1.2.4)**. This is especially true for the structural models close to architecture, e.g. interface and signal descriptions that can be used to generate skeleton code or documentation. However, also tracing was mentioned several times as a means to understand changes, support product variants, or support verification activities. On higher levels

**Table 5** Potential of models during RE: support for different themes by area

| ID | Potential | EmbSys | SysE | AppSE |
|---|---|---|---|---|
| T2.1 | Analyses/execution | 3/8 | 1/3 | 2/3 |
| T2.2 | Communication | 0/8 | 2/3 | 3/3 |
| T2.3 | Manage evolution | 3/8 | 1/3 | 1/3 |
| T2.4 | Provide context | 2/8 | 0/3 | 0/3 |
| T2.5 | Standardisation | 1/8 | 3/3 | 2/3 |

of abstraction, use case models were brought up specifically in Company B as a way to understand the existing system functionality and allow for easier impact analysis when new feature or change requests are incoming from customers.

## 5.2 RQ2: What is the perceived potential of models in automotive RE?

We identified five themes with respect to the use of models in automotive RE in our data. These are depicted in Table 5. Several of these overlap with the current use of models at the case companies. In the following, we describe these themes in detail.

Note that, in our data, when we asked about the potential of models in RE, interviewees referred to current use, past use, and potential future use of models. Therefore, by *perceived potential* of models, we refer to all three cases.

### 5.2.1 Formal analysis and execution (T2.1)

In theory, MBE enables practitioners to increase quality of the resulting system by automating repetitive tasks, e.g. through code generation, or by running a number of analyses on existing models, e.g. through model checking.

Interestingly, only three interviewees in Company B and none in Company A raised formal analysis of requirements models as a potential use case. One interviewee mentioned that code generation should be used to generate skeleton code of interface models (see Sect. 5.2.5). Two interviewees in Company B stated that it would be beneficial to have use case models in order to automatically check for existing behaviour.

> What does the product contain already? What do we need to develop newly? Here it would be cool if we had the existing products as a model.—B3, AppSE

In Company A, the importance of models for early validation with customers and other stakeholders was raised repeatedly. Three interviewees stated that they would like to use executable models to simulate parts of the system behaviour. This could then be used to demonstrate early concepts or to establish a common understanding in a group of engineers or stakeholders.

> I think information management is key. [..] I do believe in using models for testing. To have things that you can show people, that you can work with. That you can already try the specification or the model.—A2, EmbSE

One interviewee expressed the belief that the company should go directly from high-level textual requirements to executable models.

> I think we should have (high-level requirements) in whatever Word document that is needed and then we go directly to a model where we can execute the model and simulate the behaviour and the functionality.—A3, EmbSE

One of the reasons for having executable requirements models is, according to another interviewee at Company A, that faults can easily be injected and the resulting behaviour assessed.

### 5.2.2 Communication (T2.2)

In contrast to formal analyses of models, all but one interviewee in Company B stated that they would like to use requirements models for communication purposes only, especially among people with different backgrounds.

> On cross-discipline level. [..] There I think it makes most sense, as more and more experts and disciplines emerge [..], but the connections, that's the point.—B2, SysE

While models that are used for communication could possibly also be reused for formal analyses, one interviewee stressed the importance of informal models for communication.

> Well, models. When you say you work with pictures, with sketches, graphics, that definitely helps. But flow diagrams, when you are that far then usually everybody understood it already. The problems come before that.—B5, AppSE

Interestingly, none of the interviewees in Company A pointed out models as a means to communicate only.

### 5.2.3 Manage system evolution (T2.3)

Evolution was mentioned several times by interviewees in both case companies. As both companies build products that are maintained for a long period of time and that often build on already existing products, evolution is an important aspect in their daily work.

One interviewee in each company suggested that requirements models could be useful for regression testing. This could be in the form of executable models similar to Behaviour Driven Development [42]. That is, requirements would be expressed in terms of executable models that contain acceptance criteria or similar information that could be used for testing later on.

Similar to understanding the impact of new feature requests, as discussed previously, several interviewees stated that it could be useful to have a structural model of the system's interfaces for change impact analysis. In fact, one interviewee in AppSE stated that they used to have a model of all the use cases in the past. While this model had been discarded in the new agile process, the interviewee expressed the need for a similar solution.

> [..] there is the work in progress and there is the product as it is. [..] we created a functional model of natural language use cases. This means the product is structured into its main functions [..], a tree with use cases in the leaves. And this represents the current state of the product. Which use cases are there? And I have to maintain that, because the work in progress [..] extends the tree, it changes the tree. But if there is no tree [..], then I don't know how you do that.—B3, AppSE

Finally, one interviewee mentioned that requirements models could be used in areas where computer tools are still missing and repeated work is necessary. In these areas, it could be useful to build a library of reusable model elements that can be used by engineers to reduce the effort and errors.

> [..] we plan to use MBE not as a individual method in small islands, but using synergies and reuse. On the long term we would like to build up a library of standard elements that can be reused.—B2, SysE

#### 5.2.4 Provide context (T2.4)

In our interviews, lack of context information in requirements specifications was mentioned by multiple interviewees as problematic (see also [32]). That is, engineers lack information of how a requirement fits into the system context, which other components it affects or it is affected by, and who is working on them. In particular for subcontractors, this information is often not available, which makes it difficult to interpret and understand requirements.

In line with this challenge, two interviewees in Company A suggested to have structural models that show how functions or subsystems are connected in the overall product. As examples, the interviewees named classical architectural models such as component models with in- and outgoing signals, but also higher-level structural models that show the relation between requirements. Such models could then be used for actually understanding the requirements better, but also to identify circular references or for change impact analyses as mentioned before.

#### 5.2.5 Standardisation (T2.5)

As a last theme, interviewees in both companies mentioned the importance of models to standardise and streamline the process in the organisation.

Two interviewees in Company B suggested a formal model syntax, e.g. as a profile for SysML. This would force engineers to think deeply about what they want to express, instead of just writing it down in natural language.

> Often, the problem is not the existence of a requirements documents, but the method to create this document. [..] That is, every specification looks slightly different [..]. And model-based means in this case a strong use of SysML as a language, which brings the advantage that people are more aware of the fact that they express something in a more formal way that before.—B1, SysE.

In a similar direction, one more interviewee from Company B suggested to use defined terms or model elements as a means of standardisation. This would allow to check for consistency on syntax level, have a uniform (graphical) representation, and create different views of the model for different stakeholders.

Finally, four interviewees brought up the need for interface models to structure the work. These would allow a clearer separation of subsystems, but also of the system to its environment. As discussed in Sect. 5.1, these models already exist to some extent in both companies and, while their main purpose is architecture and design, they do connect to RE, e.g. for structuring of requirements.

### 5.3 RQ3: Why are models not used during automotive RE?

While models are clearly already used to some extent during RE in the automotive domain, we identified several obstacles that restrict a wider use of models during this important phase of systems engineering. In the following, we discuss these in detail. An overview is given in Table 6.

Mirroring the large focus on tooling in related work on MBE, we identified several challenges with respect to modelling tooling in our data. These relate to the customisation of tools, the use of multiple tools in an integrated way, and the extraction of information from tools.

Additionally to tooling challenges, we identified several themes that can be seen as general challenges in the area of modelling and model use.

**Table 6** Challenges for model use during RE: support for different themes by area

| ID | Challenge | EmbSys | SysE | AppSE |
|---|---|---|---|---|
| T3.1 | Interoperability or single tool | 3/8 | 0/3 | 1/3 |
| T3.2 | Need for customisation | 3/8 | 2/3 | 0/3 |
| T3.3 | Information extraction from tools | 2/8 | 2/3 | 2/3 |
| T3.4 | High effort | 2/8 | 2/3 | 2/3 |
| T3.5 | High complexity | 3/8 | 0/3 | 0/3 |
| T3.6 | Accidental design/detail | 6/8 | 2/3 | 0/3 |
| T3.7 | Insufficient maturity | 3/8 | 3/3 | 1/3 |
| T3.8 | Organisation resistance | 2/8 | 2/3 | 1/3 |

#### 5.3.1 Shortcomings of modelling tools

In both companies, interviewees stated that a variety of different tools is needed, meaning that these tools **need to be connect and interoperate (T3.1)**. Five interviewees stated that it would be unrealistic to assume that a company can use a single tool in the same way across different organisation units and especially disciplines. For example, one interviewee in Company B stated:

> One size fits all won't work for Company B. Definitely not. [..] as I already said in systems engineering we are in the mechatronic world...we wouldn't be interested to re-build existing UML models for software for SysML. It also wouldn't help us.—B2, SysE

Similarly, interviewees in Company A refused the idea to use Simulink as a general purpose tool for requirements modelling, as other interviewees in the same company proposed.

> I think Simulink is really good for some parts. It's absolutely, really wrong for HMI parts. And it's not really quite super good for state machines. So the thing is that this modelling has to be very flexible —A6, EmbSE

Similarly, tools **need to be customisable (T3.2)** to fit existing processes. However, according to the interviewees, the possibility to customise existing modelling tools is currently limited.

> [..] they are all in their core 10, 15 years old, or older. [..] that means you have big problems customising these tools in an easy way. That is, directly highlighting certain menus, hiding certain features.—B2, SysE

Finally, six interviewees in both companies stressed the need to be able to **make information stored in models more accessible (T3.3)**. In this context, one interviewee stated:

> Most users don't care about diagrams. They are concretely interested in 'I'm responsible for this component, and this requirements, or this function. I'd simply like to have a list, whatever, a matrix view, which tells me what the function is, what attributes it has, [..], which interfaces [..].' [..] I'm not interested in the whole model. That is, the gap between the expert who creates the model and the group [..] which interacts or works with the model later.—B1, SysE

Concretely, interviewees expressed the need for tables, layouting, easy model navigation, and tracing capabilities in modelling tools.

Despite these challenges, it is important to point out that several interviewees stated that tooling is not the main problem in the introduction of modelling in RE.

> [..] most of the functionality that we have today in (Requirements tool) is good enough...I see few tooling problems, very few tooling problems. There is a lack of information thinking. But that's very much linked to the key concept of how the product is documented.—A6, EmbSE

#### 5.3.2 Complexity and effort of modelling

In both companies, the **high effort (T3.4)** to create and maintain requirements models was brought up. One interviewee stated that it would be too much effort to reach complete formalisation, but suggested that it would be feasible to reach a large amount of formalisation with little effort. Therefore, formalising parts of the overall requirements specifications should be aimed for. In particular, one interviewee in Company A added that it is difficult to express non-functional requirements as models.

A similar aspect raised by the interviewees was that natural language is needed in addition to models, as these quickly **become very complex (T3.5)**.

> I think you still need to write what the purpose of the function would be. But probably not like the exact logical behaviour.—A2, EmbSE

In particular, interviewees in Company A saw the risk that there is a large amount of domain and modelling knowledge needed to understand existing models, meaning that recipients of the requirements models would need additional explanations in natural language.

#### 5.3.3 Organisation and process aspects of modelling

One of the large risks brought up by eight interviewees was the risk of **accidental complexity and design (T3.6)** when creating requirements models. One interviewee stated that it is easy to go into too much detail when creating models.

> It should be on higher level. [..] usually if you do a model and you want simulate it and so on, then you go into too much detail. [..] So to avoid these kind of problems, for the high-level requirements I think it is enough with textual requirements.—A3, EmbSE

Another interviewee added that this level of detail would require requirements engineers to have detailed development knowledge to be able to create requirements models.

While many interviewees felt that they should distinguish between requirements and design, a position also common in RE literature, they found this difficult to achieve in practice. In Company A, they even removed one abstraction layer in their requirements in the past, as engineer had difficulties to distinguish them in practice.

> What we have now is sort of a merge of the analysis and the design. It's more towards the design, but it's sort of a mix. Because it was too complex for users to grasp how to distinguish between analysis and design.—A5, EmbSE

However, one interviewee in Company B stated that, overall, the right level of abstraction would simply be an iterative learning process.

> In the first step, it will always end up wrong. Either too little or too much or something else. [..] You will approximate this iteratively. What is needed? What is meaningful?—B1, SysE

In addition to more technical concerns, several interviewees stated that their **organisations** are **not mature enough (T3.7)** to introduce modelling during RE. This was stated both with respect to the necessary skills to actually create and understand models, and with respect to the process currently used in the companies. Three interviewees stated that engineers would currently lack the necessary skills for modelling.

> I mean because the persons that write requirements for (vehicles) know about (vehicles). [..] So how many (domain experts) know Simulink? I don't know.—A8, EmbSE

On an organisation level, several interviewees stated that there would either be a lack of support from management, or a lack of understanding of the actual problem that requirements models would aim to solve.

> Technically, everything works. It's only that companies like Company B [..] have to figure out 'What do we want to fix?'. [..] Sometimes, they don't even have a real understanding of their problem, leave alone a solution.—B2, SysE

Specifically, interviewees named the integration of requirements modelling into an interdisciplinary work environment, a suitable process for using requirements models, and the right education and training strategy.

Finally, interviewees stated that there might be **resistance from employees (T3.8)** when introducing models during RE. While several interviewees in Company B stated that employees with different education levels and backgrounds had been successfully trained in using models, in particular SysML, they expressed concerns that modelling would be a paradigm shift in the company. Additionally, one interviewee added that there could be resistance specifically when employees see their roles changing or even fear that their expertise is no longer needed.

> I think a lot is simply a human factor. There are for example also those who have the overview. They sometimes don't want it (the change), as they realise that the privilege they're having, what defines their role, is in danger.—B2, SysE

## 6 Discussion

In the following, we discuss our results and their implications with respect to related work, our hypotheses, and our three research questions. The support of our data to our hypotheses is summarised in Table 7.

### 6.1 RQ1: use of models in RE

Regarding the current use of models in RE (RQ1), we observe that indeed requirements are still mainly specified in natural language (**H1**). However, there are several attempts to introduce model-based specifications, and different kinds of models already exist in both case companies, ranging from informal sketches to formal UML models.

The use of existing models covers a variety of purposes, namely communication, change impact analysis, or guidance. While this supports our hypothesis **H2**, it is important to note that the use of formal models is typically restricted to modelling experts.

The range of models that interviewees use as a part of RE supports the statements by Eliasson et al. [16], that architecture and requirements are indeed intertwined in the automotive industry. Specifically, structural models are used for the architecture as well as to structure and guide RE.

### 6.2 RQ2: potential of models in RE

The potential of models in RE (RQ2) is perceived differently by our interviewees.

**Table 7** Hypotheses support

| Nr. | Description | Support |
|---|---|---|
| **H1** | Requirements are mainly specified in natural language | Yes |
| **H2** | Models are already used in automotive RE | Yes, but restricted to experts |
| **H3** | Using models during Requirements Engineering could significantly improve requirements validation and verification | Yes |
| **H4** | There is insufficient tool support for modelling requirements | Yes |
| **H5** | Using UML for modelling requirements is deficient and frustrating | Not enough evidence |
| **H6** | Requirements Engineers lack guidance on how to use models within Requirements Engineering | Yes, on organisation level |
| **H7** | The main problem in automotive RE is finding the right level of abstraction | Yes, when modelling requirements |

While several interviewees point out that they see the value of models in RE mainly as a tool for communication without the need for formality, other interviewees express the potential benefit of simulations or automated analyses using formal models. Specifically, we see that only interviewees in Company B mentioned models as a communication tool, whereas simulation was mentioned only in Company A. These different perceptions could be related to multiple factors, including the abstraction level of requirements, the amount of disciplines involved in the development of a subsystem, and experience or ongoing discussions within the company in a certain direction.

The potential that interviewees in Company A see for early simulation of requirements aligns well with related work by Brandstetter et al. [9] and Vogelsang et al. [53], both of which highlight the feasibility of such an approach. In contrast, existing frameworks that propose formalisation on all abstraction levels (e.g. SPES 2020 [46] or REMSeS [10]) conflict with our interviewees' opinion that a large amount of formalisation is not feasible. In particular, it is highly interesting to observe that even in the area of SysE, none of the interviewees expressed a need for formal analyses. It is however important to point out that our interviewees did not question the value of a completely formal approach, but rather the feasibility of it.

We believe that certain topics were not raised due to the abstraction level in RE. For example, while we know that it is not uncommon at Company A to use Simulink models for code generation, our interviewees might not have seen the feasibility to do this at a high level of abstraction. The concern regarding accidental design that several interviewees expressed supports this hypothesis.

Several interviewees brought up aspects related to system evolution, e.g. using models for regression testing. This aligns well with arguments that models during RE could improve validation and verification activities, thus corroborating our hypothesis **H3**, that the use of models during RE could significantly improve requirements validation and verification (V&V).

While our interviewees see the potential to improve V&V using models, this does not imply that textual requirements are indeed "insufficient" [25] or "not satisfactory" [50]. In contrast, several of our interviewees point out that on a higher level of abstraction, textual requirements should remain the state of the art, e.g. to lower the effort of specification and management of requirements, and to allow expressing uncertainty. This shows that we need a more differentiated picture of using models in RE in future work, not assuming that every abstraction level in RE or architecture can be treated equally.

Finally, several of our findings connect well with the discussion on combining RE and architecture in an iterative fashion, e.g. the TwinPeaks model [43]. In addition to the already used structural models, several of the proposed use cases for RE models connect to architectural concerns, e.g. automated change impact analyses, defined interface models for subsystems, or standardisation using models. None of the interviewees stated a desire to separate RE and architecture more.

### 6.3 RQ3: showstoppers of models in RE

Based on our data, there are several explanations to why models are not used in RE (RQ3). We see a clear overlap with themes mentioned in empirical studies that focus on the use of models in general.

One of the central showstoppers of models in RE is insufficient tool support (corroborating hypothesis **H4**), in particular the (lack of) customisability of modelling tools, the (lack of) interoperability between multiple tools, and the capabilities to access information stored in models. While the former two are typically mentioned in related work, e.g. in [4,30,40,41], the latter aspect is not commonly named. This could be attributed to our focus on RE, as RE is concerned with capturing knowledge from different domains and making this knowledge available to engineers later on. As such, the knowledge potentially captured in models needs to be accessible in an easy and quick fashion.

While UML was mentioned by some of our interviewees, we do not have enough evidence supporting hypothesis **H5**, that using UML for modelling requirements is deficient and frustrating. It is, however, worth mentioning that multiple interviewees at Company B reported promising first results using SysML, which is after all based on UML.

Also similar to other empirical studies, e.g. [28,56], organisation aspects were raised by several of our interviewees. These relate to training and cultural issues in an organisation, which is in line with [56].

In addition to organisation aspects mentioned in related work, our interviewees brought up that it is often unclear to organisations what problem in particular modelling should solve. This relates to hypothesis **H6**, that requirements engineers lack guidance on how to use models within RE, but on an organisation level not restricted to a role or an organisation unit. Hence, the general proposition that modelling or MBE offers and increase productivity, effectiveness or similar aspects might be too vague for organisations to grasp the actual relevance to their own challenges.

The third main aspect of our findings related to RQ3 is the effort and complexity to create and maintain requirements models. Several interviewees saw this as a showstopper for modelling, as a challenge, or at least as a lengthy learning process within the organisation. In both case companies, interviewees stated that existing requirements would have to be re-created as models, as they are rarely written from scratch.

Given the effort, multiple interviewees highlighted that a complete formalisation is not feasible. However, they reasoned that lots of benefits could be reached from formalising smaller proportions of the requirements, e.g. the vocabulary or the meaning of certain model elements. Similar to the discussion regarding models for communication purposes (see RQ2), this finding conflicts with existing approaches containing large amounts of formalised requirements. In fact, our findings are in line with experiences made with SPES, namely that there are "problems resulting from an increased number of dependencies" and that there is a "the need to cope with redundancies" [11]. That is, problems arising due to the size of a large system, and its requirements or specification.

Finally, several interviewees also voiced concern that when creating requirements models, it would be easy to end up with accidental design instead of requirements. This supports to some extent hypothesis **H7**, that the main problem in automotive RE is finding the right level of abstraction, albeit only with respect to requirements models and not that it is in fact the main problem.

### 6.4 Implications to research and practice

Reaching beyond the individual research questions, our findings have several implications for research and for industrial practice.

We note that **an exclusive focus on formalisation and analyses based on formal models does not align well with current industry needs**. While none of the interviewees outright questioned formalisation of requirements, they questioned the feasibility of creating and maintaining large amounts of formal models. Therefore, there should be an increased focus in future work on investigating the use of semi-formal or informal notations in RE, basically a *realistic, operable* way to create, manage and use models in RE.

We observe that **several of the challenges preventing the use of models in automotive RE are topics typically neglected in research**. In many cases, these topics do not pose a clear academic contribution, e.g. improvements in usability or customisability. Hence, while the theoretical foundations needed for using models during RE in an effective fashion, the technology transfer is as of now insufficient. It is however worth noting that the academic community is starting to embrace this shortcoming, as shown by several keynotes and panels at recent editions of the MODELS conference series focusing on precisely the mentioned topics, e.g. user experience, tool interoperability, and technology transfer.

A further important aspect arising from our findings is that **creators, but especially users, of models are diverse, and often not experts in formal methods or modelling**. That is, the presence of multiple disciplines with a focus not exclusively on software engineering, and the existence of a wide variety of tools and processes within an organisation needs to be taken into account. As such, the need to easily obtain information stored in existing models is a key aspect. This goes beyond the already mentioned issues of tool interoperability and customisation, towards information extraction and management from models.

Similar to the diverse nature of model creators/users in practice, we see that the **issues preventing the use of models in RE reach beyond software engineering research**. Starting from classical software engineering and computer science research topics, such as technical tool interoperability solutions, the topic includes questions of how to industrialise existing tools or questions of organisational nature, such as introduction of models or managing organisational change. Similar to benefiting from models for bridging disciplines, as mentioned by our interviewees, there are therefore substantial issues related to modelling that call for interdisciplinary research.

Our findings contain **several positive experiences of using models during RE**. Examples include the model-based tool support in Company A that structures all informa-

tion within systems engineering, or the use of both informal and formal models for communication purposes Company B. Additionally, our interviewees mention multiple potential use cases for models in RE. We believe that these can serve as a source of inspiration for companies that would like to introduce models in RE, or for individuals trying to convince management of the benefits of modelling. In particular, some of the mentioned approaches require only low introduction overhead compared, e.g. using models for interdisciplinary communication.

Finally, the views of our interviewees in modelling and pilot projects currently ongoing in both companies show that **there is still a strong belief in the benefits of modelling within RE**. As researchers, this should serve as a motivation to continue our endeavours, while practitioners can be reassured that modelling is likely not a dead end.

## 7 Conclusions and future work

In this paper, we presented a multiple-case study conducted at two automotive companies with data collected from 14 interviews with practitioners in different roles. Our goal was to investigate to what extent models are currently used in automotive RE, what the perceived potential of models is, and what reason exists for not using them. We complement existing work on the use of models in industry with a detailed view on RE, providing in-depth qualitative data.

Our findings are that models are widely used in automotive RE. This usage is however often restricted to specific purposes, e.g. to align RE with verification activities. Additionally, we see widespread use of semi-formal and informal notations, e.g. model sketches to improve understanding, and keywords or sentence patterns in textual requirements.

Regarding the potential of models in automotive RE, our interviewees disagree to the extent models can be beneficial. Few interviewees consider it feasible to create and maintain formal models, even though their benefits would be significant. Instead, multiple interviewees see the main benefit in semi-formal notations, e.g. for communication purposes.

Finally, we report reasons for this limited adoption of models. The main showstoppers, according to our data are tool restrictions with respect to customisability, information extraction and interoperability, high complexity of tools, and organisational reasons, such as resistance to change.

For future work, we encourage interdisciplinary studies on the introduction and use of models, both in RE and in general. While focusing on aspects exclusively related to software engineering, there is a clear need to study MBE in an organisation and management context.

Due to the large amount of issues not directly related to research topics, we see the need for strategic partnerships between academia, companies that want to use/introduce MBE, and companies that offer commercial support and customisation for MBE. The academic side has the task of understanding the state of practice (in a similar fashion as this study aimed to do) and pushing the boundaries of MBE research. However, industry can only benefit from this line of research to the fullest extent if combined with customisation and tailoring of MBE tools and approaches to the organisation context, something that research can and should not focus on.

**Acknowledgements** The research leading to these results has received partial funding from the European Union's Seventh Framework Program (FP7/2007-2013) for CRYSTAL-Critical System Engineering Acceleration Joint Undertaking under Grant Agreement No 332830 and from Vinnova under DIARIENR 2012-04304.



## A Interview guide

### A.1 Part A, introduction

- The interviewer shortly presents the topic and the research questions (2–3 min).
- The interviewer points out that the interview is anonymous, covered by an NDA agreement with the respective company.
- The interviewer asks for permission to record the interview, in order to facilitate data analysis later on.
- The interviewer also points out that the interviewee will receive the transcribed interview once it is available and may review it and/or object to some of the stated points. In this case, the interviewers may not use these parts of the data.
- Finally, the interviewer points out that the final data analysis for each company will be discussed with the contact person at the respective company and possibly with the interviewees as well, if desired.

### A.2 Part B, demographic questions

1. In which role do you work at [Company]?
2. Could you shortly introduce your work at [Company]?
3. How long have you worked with Requirements Engineering/Software Engineering/Verification?
4. In which other areas at [Company] have you worked before?
5. How long have you worked overall in the automotive industry?

### A.3 Part C, topic questions

1. Could you shortly describe the Requirements Engineering process at [Company] based on your role?
   *Comment: e.g. from a Requirements Engineer's point of view, this will be the elicitation process, from a developer's view more the exchange/usage of requirements, etc.*
2. When do you elicit new requirements within [Company]?
   *For non requirements engineers: Their point of view, their role*
3. How do you elicit new requirements within [Company]?
   *For non requirements engineers: Their point of view, their role*
   *Quality requirements?*
4. How are requirements documented?
   *Req. Eng.: How do you do it?; SE: How are they provided to you?*
5. How do you specify variability in your requirements?
   *All roles: How is it done right now? Req. Eng.: Where does the variability come from? Who decides on the variants?*
6. (How) Do you reuse requirements?
   *SE/Verification: Do you often get similar requirements? Do they seem to be copied or follow a similar schema?*
7. Is there a glossary of terms which have to be used during specification or guidelines to follow for writing the specification?
   *Probably implicit?*
8. At which level of abstraction are requirements specified?
9. How is the right level of abstraction determined?
   *Are there any policies/rules? Is it up to the Req. Eng.?*
10. How much does the level of abstraction change throughout the process? When are the requirements refined?
    *Goes back to the second question, different phases/ iterations.*
11. Is this the right abstraction, in your opinion?
12. How is the specification used?
    *During requirements engineering, during negotiation, during development, testing, etc.?*
13. How are requirements traced to/from later artefacts?
    *Unidirectional traces? Bidirectional?*
14. How do you handle ambiguous or unclear requirements?
    *Req. Eng.: How do you avoid them?, Developer/Tester: What do you do with them?*
15. Do you know of any problems you had in the past with ambiguous/unclear requirements?
16. Do you have ways to measure the quality of your specification? Which ones?
17. Are you using any kind of models for specifying or clarifying requirements within [Company]?
    **Yes:**
    (a) What is the purpose of these models?
    *Non-functional properties as well?*
    (b) Are the models used exclusively or in combination with NL requirements?
    (c) What are the (dis-)advantages of this method?
    (d) Is this use of models widespread?
    (e) Which aspects of modelling/models do you like/ dislike in particular?
    (f) How do you provide tracing to elements in these models?
    (g) For which additional purposes could models be beneficial?

    **No:**
    (a) Why do you not use any models?
    (b) What would have to change in order for you to introduce models?
    (c) Do you think it could be beneficial to use some kind of model?
    (d) For which purposes?
    *Only functionality, or also non-functional properties?*
18. Which tools, languages, formalisms would you like to use during requirements engineering? Why?
    *What might possible risks of doing so be?*
19. Which ones wouldn't you like to use? Why not?

### A.4 Part D, finish

- The interviewer thanks for the participation
- Additionally, the interviewer encourages the interviewee to come back with any comments or questions after the interview, if they may arise.

## B Chain of evidence

The interview questions are connected to our seven hypotheses as depicted in Table 8. Please note that only part C of the questions is connected to hypotheses, as parts A and D are only the introduction and conclusion of the interviews, and part B is to collect additional background.

## C Codes

1. Current state of practice
2. FutureImprovement
3. Problem
4. Process

**Table 8** Hypothesis-question linking

| Question | H1 | H2 | H3 | H4 | H5 | H6 | H7 |
|---|---|---|---|---|---|---|---|
| B1 | X | X | | X | | | X |
| B2 | | | | | | | |
| B3 | | | | | | | X |
| B4 | X | | | | | | X |
| B5 | X | | | | | | |
| B6 | | | | | | | |
| B7 | X | | | | | | |
| B8 | | | | | | | X |
| B9 | | | | | | | X |
| B10 | | | | | | | X |
| B11 | | | | | | X | X |
| B12 | | | | | | | |
| B13 | X | | | | | | |
| B14 | | | X | | | | |
| B15 | | | X | | | X | |
| B16 | | | | | | | X |
| B17 | X | X | X | X | X | X | |
| B18 | | | X | X | X | X | |
| B19 | | | X | X | X | X | X |

(a) Clarification
(b) Communication
(c) Elicitation
(d) Guidance
(e) Inter-organisational
(f) Intra-organisational
(g) Measurement
(h) Modelling
(i) Refinement
(j) Requirements usage
(k) Verification and validation

5. Requirements

(a) Ambiguity/understandability
(b) Notation
(c) Level of abstraction
(d) Quality
(e) Reuse
(f) Tooling
(g) Tracing
(h) Variability

## References


1. Agner, L.T.W., Soares, I.W., Stadzisz, P.C., Simão, J.M.: A brazilian survey on UML and model-driven practices for embedded software development. J. Syst. Softw. **86**(4), 997–1005 (2013). SI : Software Engineering in Brazil: Retrospective and Prospective Views
2. Autili, M., Grunske, L., Lumpe, M., Pelliccione, P., Tang, A.: Aligning qualitative, real-time, and probabilistic property specification patterns using a structured english grammar. IEEE Trans. Softw. Eng. **41**(7), 620–638 (2015)
3. Avgeriou, P., Grundy, J., Hall, J.G., Lago, P., Mistrík, I.: Relating Software Requirements and Architectures. Springer, Berlin (2011)
4. Baker, P., Loh, S., Weil, F.: Model-driven engineering in a large industrial context—motorola case study. In: Briand L.C., Williams C. (eds.) Model Driven Engineering Languages and Systems, Lecture Notes in Computer Science, vol. 3713, pp. 476–491 (2005)
5. Barmi, Z.A., Ebrahimi, A.H., Feldt, R.: Alignment of requirements specification and testing: a systematic mapping study. In: 2011 IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops, pp. 476–485 (2011)
6. Berenbach, B., Schneider, F., Naughton, H.: The use of a requirements modeling language for industrial applications. In: 20th IEEE International Requirements Engineering Conference (RE), pp. 285–290 (2012)
7. Böhm, W., Henkler, S., Houdek, F., Vogelsang, A., Weyer, T.: Bridging the gap between systems and software engineering by using the spes modeling framework as a general systems engineering philosophy. Procedia Comput. Sci. **28**, 187–194 (2014)
8. Böhm, W., Junker, M., Vogelsang, A., Teufl, S., Pinger, R., Rahn, K.: A formal systems engineering approach in practice: an experience report. In: Proceedings of the 1st International Workshop on Software Engineering Research and Industrial Practices, pp. 34–41. ACM (2014)
9. Brandstetter, V., Froese, A., Tenbergen, B., Vogelsang, A., Wehrstedt, J.C., Weyer, T.: Early validation of automation plant control software using simulation based on assumption modeling and validation use cases. Complex Syst. Inform. Model. Q. **4**, 50–65 (2015)
10. Braun, P., Broy, M., Houdek, F., Kirchmayr, M., Müller, M., Penzenstadler, B., Pohl, K., Weyer, T.: Guiding requirements engineering for software-intensive embedded systems in the automotive industry. Comput. Sci. Res. Dev. **29**(1), 21–43 (2014)
11. Brings, J., Bellendorf, J., Keller, K., Kempe, M., Kurt, N., Palm, A., Daun, M.: Applying the spes modeling framework. In: Joint Proceedings of REFSQ-2017 Workshops, Doctoral Symposium, Research Method Track, and Poster Track co-located with the 22nd International Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2017) (2017)
12. Broy, M.: Challenges in automotive software engineering. In: Proceedings of 28th International Conference on Software Engineering (ICSE '06), pp. 33–42 (2006)
13. Creswell, J.W.: Research Design: Qualitative, Quantitative, and Mixed Methods Approaches, Chap. 10, 2nd edn. SAGE Publications, Beverley Hills (2003)
14. de Almeida Ferreira, D., da Silva, A.R.: A controlled natural language approach for integrating requirements and model-driven engineering. In: 2009 Fourth International Conference on Software Engineering Advances, pp. 518–523 (2009)
15. Dwyer, M.B., Avrunin, G.S., Corbett, J.C.: Patterns in property specifications for finite-state verification. In: Proceedings of the 1999 International Conference on Software Engineering (IEEE Cat. No.99CB37002), pp. 411–420 (1999)
16. Eliasson, U., Heldal, R., Knauss, E., Pelliccione, P.: The need of complementing plan-driven requirements engineering with emerging communication: Experiences from volvo car group. In: IEEE 23rd International Requirements Engineering Conference (RE '15), pp. 372–381 (2015). https://doi.org/10.1109/RE.2015.7320454
17. Fockel, M., Holtmann, J.: A requirements engineering methodology combining models and controlled natural language. In: IEEE

4th International Model-Driven Requirements Engineering Workshop (MoDRE '14), pp. 67–76 (2014)
18. Forward, A., Lethbridge, T.C.: Problems and opportunities for model-centric versus code-centric software development: a survey of software professionals. In: International Workshop on Models in Software Engineering (MiSE '08), pp. 27–32 (2008)
19. Gervasi, V., Zowghi, D.: Reasoning about inconsistencies in natural language requirements. ACM Trans. Softw. Eng. Methodol. **14**(3), 277–330 (2005)
20. Glinz, M.: Problems and deficiencies of uml as a requirements specification language. In: 10th International Workshop on Software Specification and Design, p. 11 (2000)
21. Gorschek, T., Tempero, E., Angelis, L.: On the use of software design models in software development practice: an empirical investigation. J. Syst. Softw. **95**, 176–193 (2014). https://doi.org/10.1016/j.jss.2014.03.082
22. Haghighatkhah, A., Oivo, M., Banijamali, A., Kuvaja, P.: Improving the state of automotive software engineering. IEEE Softw. **34**(5), 82–86 (2017). https://doi.org/10.1109/MS.2017.3571571
23. Hebisch, E., Book, M., Gruhn, V.: Scenario-based architecting with architecture trace diagrams. In: 2015 IEEE/ACM 5th International Workshop on the Twin Peaks of Requirements and Architecture, pp. 16–19 (2015)
24. Heumesser, N., Houdek, F.: Experiences in managing an automotive requirements engineering process. In: Proceedings of 12th IEEE International Requirements Engineering Conference (RE '04), pp. 322–327 (2004)
25. Houdek, F., Pohl, K.: Analyzing requirements engineering processes: a case study. In: Proceedings of 11th International Workshop on Database and Expert Systems Applications, pp. 983–987 (2000)
26. Hutchinson, J., Rouncefield, M., Whittle, J.: Model-driven engineering practices in industry. In: 33rd International Conference on Software Engineering (ICSE '11), pp. 633–642 (2011)
27. Hutchinson, J., Whittle, J., Rouncefield, M.: Model-driven engineering practices in industry: social, organizational and managerial factors that lead to success or failure. Sci. Comput. Program. **89, Part B**, 144–161 (2014). SI: Success Stories in Model Driven Engineering
28. Hutchinson, J., Whittle, J., Rouncefield, M., Kristoffersen, S.: Empirical assessment of mde in industry. In: 33rd International Conference on Software Engineering (ICSE '11), pp. 471–480 (2011)
29. Kirstan, S., Zimmermann, J.: Evaluating costs and benefits of model-based development of embedded software systems in the car industry–results of a qualitative case study. In: Workshop C2M: EEMDD "From Code Centric to Model Centric: Evaluating the Effectiveness of MDD" (2010)
30. Liebel, G., Marko, N., Tichy, M., Leitner, A., Hansson, J.: Assessing the state-of-practice of model-based engineering in the embedded systems domain. In: Dingel J., Schulte W., Ramos I., Abraho S., Insfran E. (eds.) Model-Driven Engineering Languages and Systems, Lecture Notes in Computer Science, vol. 8767, pp. 166–182 (2014). https://doi.org/10.1007/978-3-319-11653-211
31. Liebel, G., Marko, N., Tichy, M., Leitner, A., Hansson, J.: Model-based engineering in the embedded systems domain: an industrial survey on the state-of-practice. Softw. Syst. Model. (2016). https://doi.org/10.1007/s10270-016-0523-3
32. Liebel, G., Tichy, M., Knauss, E., Ljungkrantz, O., Stieglbauer, G.: Organisation and communication problems in automotive requirements engineering. Requir. Eng. **23**(1), 145–167 (2016)
33. Loniewski, G., Insfran, E., Abraho, S.: A systematic review of the use of requirements engineering techniques in model-driven development. In: Petriu D., Rouquette N., Haugen O. (eds.) Model Driven Engineering Languages and Systems, Lecture Notes in Computer Science, vol. 6395, pp. 213–227 (2010). https://doi.org/10.1007/978-3-642-16129-216
34. Marko, N., Leitner, A., Herbst, B., Wallner, A.: Combining xtext and oslc for integrated model-based requirements engineering. In: 2015 41st Euromicro Conference on Software Engineering and Advanced Applications, pp. 143–150 (2015)
35. Mayring, P.: Qualitative content analysis-research instrument or mode of interpretation. Role Res Qual. Psychol. **2**, 139–148 (2002)
36. Méndez Fernández, D., Lochmann, K., Penzenstadler, B., Wagner, S.: A case study on the application of an artefact-based requirements engineering approach. In: Evaluation and Assessment in Software Engineering (EASE 2011), 15th Annual Conference on, pp. 104–113. IET (2011)
37. Méndez Fernández, D., Penzenstadler, B., Kuhrmann, M., Broy, M.: A meta model for artefact-orientation: fundamentals and lessons learned in requirements engineering. In: International Conference on Model Driven Engineering Languages and Systems, pp. 183–197. Springer Berlin Heidelberg (2010)
38. Méndez Fernández, D., Wagner, S., Lochmann, K., Baumann, A., de Carne, H.: Field study on requirements engineering: Investigation of artefacts, project parameters, and execution strategies. Inf. Softw. Technol. **54**, 162–178 (2011)
39. Méndez Fernández, D., Wieringa, R.: Improving requirements engineering by artefact orientation. In: International Conference on Product Focused Software Process Improvement, pp. 108–122. Springer Berlin Heidelberg (2013)
40. Mohagheghi, P., Dehlen, V.: Where is the proof?—A review of experiences from applying mde in industry. In: Schieferdecker I., Hartman A. (eds.) Model Driven Architecture—Foundations and Applications, Lecture Notes in Computer Science, vol. 5095, pp. 432–443 (2008)
41. Mohagheghi, P., Gilani, W., Stefanescu, A., Fernandez, M.A., Nordmoen, B., Fritzsche, M.: Where does model-driven engineering help? Experiences from three industrial cases. Softw. Syst. Model. **12**(3), 619–639 (2013)
42. North, D.: Introducing behaviour driven development. Better Software Magazine (2006). https://dannorth.neintroducinbdd/
43. Nuseibeh, B.: Weaving together requirements and architectures. Computer **34**(3), 115–119 (2001)
44. Paech, B., Dutoit, A.H., Kerkow, D., Von Knethen, A.: Functional requirements, non-functional requirements, and architecture should not be separated œa position paper (2002)
45. Penzenstadler, B., Sikora, E., Pohl, K.: A requirements reference model for model-based requirements engineering in the automotive domain. In: International Working Conference on Requirements Engineering: Foundation for Software Quality, pp. 212–217 (2009)
46. Pohl, K., Hönninger, H., Achatz, R., Broy, M.: Model-Based Engineering of Embedded Systems: The SPES 2020 Methodology. Springer, Berlin (2012)
47. Runeson, P., Höst, M., Rainer, A., Regnell, B.: Case Study Research in Software Engineering, 1st edn. Wiley, Hoboken (2012)
48. Rungta, N., Tkachuk, O., Person, S., Biatek, J., Whalen, M.W., Castle, J., Gundy-Burlet, K.: Helping system engineers bridge the peaks. In: Proceedings of the 4th International Workshop on Twin Peaks of Requirements and Architecture, TwinPeaks 2014, pp. 9–13 (2014)
49. Scaled Agile Inc.: Scaled agile framework. http://www.scaledagileframework.com/. Last Accessed Feb 2015
50. Sikora, E., Tenbergen, B., Pohl, K.: Requirements engineering for embedded systems: an investigation of industry needs. In: Berry D., Franch X. (eds.) Requirements Engineering: Foundation for Software Quality, Lecture Notes in Computer Science, vol. 6606, pp. 151–165 (2011)
51. Torchiano, M., Tomassetti, F., Ricca, F., Tiso, A., Reggio, G.: Preliminary findings from a survey on the md* state of the practice. In:

5th International Symposium on Empirical Software Engineering and Measurement (ESEM '11), pp. 372–375 (2011)
52. van Akkeren, E., Baumann, L., Cannegieter, J.J., Hood, C., Hruschka, P., Lampe, M., Leutbecher, E., van Loenhoud, H., de Roo, P., Staal, S., et al.: Handbook of requirements modeling according to the ireb standard. IREB International Requirements Engineering Board e.V. (2016)
53. Vogelsang, A., Eder, S., Hackenberg, G., Junker, M., Teufl, S.: Supporting concurrent development of requirements and architecture: A model-based approach. In: Model-Driven Engineering and Software Development (MODELSWARD), 2014 2nd International Conference on, pp. 587–595. IEEE (2014)
54. Weber, M., Weisbrod, J.: Requirements engineering in automotive development-experiences and challenges. In: Proceedings of IEEE Joint International Conference on Requirements Engineering (RE '02), pp. 331–340 (2002)
55. Weinstein, M.: TAMS Analyzer. http://tamsys.sourceforge.net/. Last Accessed Feb 2015
56. Whittle, J., Hutchinson, J., Rouncefield, M., Burden, H., Heldal, R.: Industrial adoption of model-driven engineering: Are the tools really the problem? In: Moreira A., Schätz B., Gray J., Vallecillo A., Clarke P. (eds.) Model-Driven Engineering Languages and Systems, Lecture Notes in Computer Science, vol. 8107, pp. 1–17 (2013)
57. Yin, R.K.: Case study: Design and Methods. Applied Social Research Methods Series; 5, 4th edn. Sage, Los Angeles (2009)

**Grischa Liebel** received the Diploma degree (Dipl.-Inform.) from Karlsruhe Institute of Technology, Germany, in computer science. He is currently with the Software Engineering division, Department of Computer Science and Engineering at Chalmers | University of Gothenburg, Sweden, pursuing a PhD degree in the area of model-driven requirements engineering. His research interests lie in model-driven engineering, software modelling, requirements engineering, empirical software engineering as well as education in these areas.

**Matthias Tichy** is a professor and director of the Institute of Software Engineering and Compiler Construction at Ulm University, Germany. Previously, he was an Assistant and Associate Professor at Chalmers University of Technology and the University of Gothenburg in Sweden. His research focuses on software engineering for embedded and technical systems. He works on model-driven system development, requirements engineering, software evolution and empirical software engineering. Matthias Tichy is author or co-author of over 100 internationally reviewed publications.

**Eric Knauss** is Associate Professor at the Department of Computer Science and Engineering, Chalmers | University of Gothenburg. His research focuses on managing requirements and related knowledge in large-scale, distributed software and systems development: Requirements Engineering, Software Ecosystems, Global Software Development, and Agile Methods. He has co-authored more than 50 publications in journals and international conferences, has been on organization and program committees of leading conferences in the field, and is a reviewer for top ranked journals. In national and international research projects, he collaborates with companies such as Bosch, Ericsson, Grundfos, GM, IBM, Siemens, TetraPak, and Volvo. More information at https://oerich.wordpress.com.