Model-Driven Engineering: A State of Affairs and Research Agenda

Charlotte Verbruggen(✉) and Monique Snoeck

Research Center for Information Systems Engineering, KU Leuven,
Naamsestraat 69, 3000 Leuven, Belgium
{charlotte.verbruggen,monique.snoeck}@kuleuven.be

Abstract. The Model-Driven Architecture has been launched in 2001 by the OMG. Since then, model-driven engineering has been embraced by the research community but less than hoped for by practitioners. To ensure the relevance of a research agenda, we need a good understanding of practitioners' problems, in particular with modelling. We therefore performed a literature review on the state of practice in the use of modelling languages for software engineering in the last 5 years according to Kitchenham's guidelines. This paper serves as orientation within the research field and as a basis for further research. It contributes to literature by focusing on papers discussing practical use of modelling languages and the benefits and problems perceived by practitioners. The main finding presented in this paper is that while practitioners experience benefits of modelling for analysis and design, requirements engineering, quality management, implementation and deployment, they still struggle with external tool integration/model transformation and export, cognitive fit, visual expressiveness, high effort required in acquiring skills, automated analysis and high effort required in using tools. Other findings are that modelling is mostly used for documentation and requirements elicitation, the most used modelling language is UML.

Keywords: Model-driven engineering · Modelling in practice · Systematic literature review · UML · BPMN

1 Introduction

The Model-Driven Architecture has been launched in 2001 by the OMG. In Model-Driven Engineering, software systems are modelled platform-independent [1]. Platform dependent models can then be obtained through transformations. Since then, modelling has been embraced by the research community but less than hoped for by practitioners. In [2], the authors discuss the need to 'pull' from the needs of practitioners instead of 'pushing' solutions from research to the industry. This is an important motivation for researchers to conduct surveys on practitioners in order to identify research paths that will solve problems the industry faces. Several surveys of practitioners' problems have been performed in the past and they all report different results. The differences may (for example) be due to the papers focusing on different aspects, having different

A. Augusto et al. (Eds.): BPMDS 2021/EMMSAD 2021, LNBIP 421, pp. 335–349, 2021.
https://doi.org/10.1007/978-3-030-79186-5_22

demographics, having been performed at a different time or because of analyzing the results through a different lens. For example, [3] is a survey that discusses the evolution of modelling in software design over a decade, whereas [4] focusses on the use of UML in open source projects. These differences will be discussed throughout this literature review. While this conveniently allows a researcher to cherry-pick a survey that best demonstrates a specific gap, the main drawback is that knowledge about the state of practice remains scattered. In order to address practitioners' problems in an adequate way, a good understanding of their problems is needed, and in order to set priorities, it is interesting to know what problems surface consistently across surveys.

The goal of this research is performing a meta-review in order to obtain a general overview of the needs of practitioners. Therefore, the research questions addressed will be of a general nature in order to capture all relevant information from the last five years. The timespan is set to five years since the industry is rapidly expanding and therefore the use of modelling languages can be drastically different now compared to five or more years ago.

In the next section, we first discuss the related research. Then, the methodology will be discussed. In Sect. 4, the results of the literature review are reported. In Sect. 5, the findings from the results are discussed.

2 Related Research

In order to find existing meta-reviews related to modelling in practice, we executed a general query on Web of Science and Scopus. The query returned 15 papers closely related to this topic. From these 15 papers, two papers have the same focus as our meta-review and are discussed below. The other 13 papers have a too narrow focus e.g.: aspects of a language or a specific domain such as cyber-physical systems, or focus on academic research instead of practitioners.

In [5], the authors performed a literature review investigating "the mismatch between the research field of modelling *language quality* evaluation and the actual MDE practice in industry". They identified seven challenges that the industry faced and that were not addressed by the research. While language quality and the identified challenges are pertinent, we aim for a broader scope.

In [6], the authors performed a systematic literature review on the "applications of ontologies in requirements engineering" selecting papers from 2007 to 2013, thus addressing a different time-frame than we aim for. Some of their conclusions are that OWL was used by the most studies as an ontology-related language to support requirements engineering and that the three main benefits of ontologies in the RE process are that they "reduce ambiguity, inconsistency or/and incompleteness", they "aid requirements management" and they improve "domain knowledge representation for guiding requirements elicitation".

3 Methodology

This paper follows the guidelines for a systematic literature review by Kitchenham [7]. First, the research questions are defined. Then, we discuss the search strategy for finding

relevant papers. In Sects. 3.3 and 3.4 we explain the inclusion and exclusion criteria and the selection procedure respectively. In the final section, the quality criteria are explained and discussed.

3.1 Defining Research Questions

The first step in a systematic literature review is to identify the research questions. Architecting systems requires dealing with a variety of concerns (also named viewpoints) and many tasks related to phases of a development lifecycle and/or goals of the requirements engineering process. This raises the following question:

RQ1 - For which aspects (type of activities) of software development do practitioners use modelling languages?

Two often named modelling languages are UML and BPMN. As a general-purpose language, UML can be used for a wide range of goals. The use of BPMN is constrained to the domain of Business Process Management. As the popularity of these languages is growing, an interesting question is to what extent these two languages are dominating the field. This leads to the second research question:

RQ2 - Which other modelling languages are in use next to BPMN and UML and how often are they used?

The first two research questions discuss how modelling languages are currently used. The final two research questions identify the negative and positive experiences that practitioners have of modelling languages.

RQ 3 - Which problems/difficulties/requirements do practitioners experience with model-driven engineering?

RQ 4 - What are the benefits of model-driven engineering according to practitioners?

3.2 Search Strategy for Finding Relevant Papers

Prior to identifying the search query, we selected a set of 5 papers as a golden standard. These are surveys that would ideally be included in the literature review. The goal of this golden standard is to check whether the query finds all these surveys: [2, 3, 8–10]. The query was executed on Web of Science & Scopus. We choose these databases because they cover a wide selection of publishers including Springer, Elsevier, IEEE and ACM. The initial query executed on both databases was:

topic = [(conceptual modelling OR UML OR BPMN) AND (practice* OR use OR practitioner* OR professional*) AND (review OR survey OR summary OR summarize*)] AND LANGUAGE = English.

This resulted in 766 papers on Web of Science and 320 papers on Scopus. In order to filter more thoroughly, we restricted the document type to “Review”. This resulted in 66 papers on Web of Science and 92 papers on Scopus. There were still many results that were related to different fields. Therefore, the results were additionally filtered on the categories that seemed irrelevant. Checking the filtering confirmed that only irrelevant papers were left out through this additional filtering.

The initial query was executed again with exclusion of papers with publication date before 2015 or in the irrelevant categories. This resulted in 221 papers on Web of Science and 102 papers on Scopus. This selection included all 4 of the 5 golden standard papers published after 2014. Removing the duplicate papers that appear in both databases led to a final query result of 277 papers.

3.3 Inclusion and Exclusion Criteria

The inclusion and exclusion criteria determine which papers will be discussed in the review. A paper is included if (1) it is an empirical study with practitioners or a literature review of empirical studies with practitioners, (2) it addresses at least one of the research questions and (3) its topic is information systems modelling, not statistical or mathematical modelling or simulation. A paper is excluded if (1) it is an empirical study with only students or academics as participants or (2) it is a theoretical paper.

3.4 Study Selection Procedure

The study selection procedure consists of three iterations. In the first iteration, two researchers assessed each paper against the inclusion and exclusion criteria using the title and abstract. When there was doubt or disagreement, the papers were included in the next iteration. 42 papers were selected in this iteration. In the second iteration, one researcher read the full papers and reassessed the inclusion and exclusion criteria. Final decision was taken jointly by the two researchers. This second iteration resulted in a set of 20 papers. In the final iteration, we analyzed the authors of the papers and the data used in the papers, and discarded two more papers to eliminate duplicate reporting on the same data.

The final selection of 18 papers is [2–4, 10–24]. Since the focus of this literature review is the experience of practitioners, the majority of the selected papers are surveys. Table 1 provides an overview of the demographic information of all the surveys included in this literature review. Papers [11–13] and [18] are not surveys but they are included because they focus on the perspective of practitioners. Paper [11] is an analysis of modelling tools and how they comply with a set of requirements. Paper [12] is a literature review that focusses on primary studies. Paper [13] is an analysis of architectural languages and how they comply with a set of requirements and paper [18] is a literature review of empirical studies on BPMN.

3.5 Quality Criteria

The quality criteria were not used for the selection of the papers, but are useful to indicate the overall quality of the papers. For each type of paper, we defined a set of criteria. For systematic literature reviews, the quality criteria are (1) the paper has a clear search strategy based on the guidelines by Kitchenham [7] and (2) the paper has a clear selection strategy based on the guidelines by Kitchenham [7]. For systematic mapping studies, the quality criterion is (3) the paper reports the protocol used. Furthermore, surveys should comply with (4) the survey design is reported in the paper, (5) detailed survey

results are included in the paper and (6) the demographics of the survey participants are discussed. Finally, all types of papers should comply with (7) a discussion of the threats to validity is included. Most papers fulfilled the quality criteria. One survey [16] did not fulfill criterion 6. The two literature reviews [12] and [18] and one survey [21] did not fulfill criteria 7.

Table 1. Summary of demographic information of the surveys.

Paper	Year of survey	Nr of re-spondents	Europe	Americas	Asia/ Pacific	Middle East	Africa	Largest group re-ported
[10]	March '18-June '18	109	34 countries; USA is the top-popular country, followed by India, France, UK, and Turkey.					USA
[2]	Oct '16-March '17	108	33%	20%	8%	5%	2%	
[14]	June-Dec 2016	115	28 countries					
[15]	Oct 2013	113	unknown					
[3]	unknown	228	10,3 %	70,5 %	19,2 %			USA/Canada 70%
[4]	July 2016	485	91 countries.					
[16]	unknown	50	unknown					
[17]	unknown	52	5,8%	15,40 %	55,80 %	0%	23,1 %	Asia/Pacific 55.80%
[19]	Jan 2016 to June 2016.	66	Worldwide					USA 37% Unknown 39%
[20]	Apr.-May 2015	627	66%	14%	19%		1%	Europe 66%
[21]	unknown	222		Brazil				
[22]	Sept 2014 - April 2016	96	67%	8%	4%			Germany 40%
[23]	Feb - April 2013	178	86,5 %	8,90 %	3,40 %	0,60 %	0,60 %	Europe 86,5% (Italy 61,8%)
[24]	unknown	17	unknown					

4 Results

The selected papers use a variety of methods of data collection. Therefore, summarizing this data is not straight forward. In order to make meaningful summaries of the data for each research question, we often discuss the papers in separate groups depending on how the results are reported. For each research question, the summary approach is clearly indicated.

4.1 RQ1 - For Which Aspects (Type of Activities) of Software Development Do Practitioners Use Modelling Languages?

Two surveys [3, 22] asked their participants to answer using a 5-point Likert scale. In order to be able to combine the results of these papers with the results of the other surveys that do not use a Likert scale, the 5-points Likert scale data needed to be transformed to binary data. The general approach used in this paper is to identify the questions of the survey that correspond to the research question and sum the percentages for the points of the Likert Scale that indicate an agreement. For example, in the survey of paper [22] a Likert scale is used to investigate to what extent participants use models for a set of suggested activities in software development (never & rarely – sometimes – often & always). In this case, only one column clearly indicates agreement with the suggested activities (often & always). Regarding the activity 'discuss with colleagues', 79% of participants indicated 'often & always' and therefore consider this as an activity for which they use models. These percentages can now be compared to the results of the surveys not using Likert scale data [2, 4, 10, 13, 15, 18, 20].

The surveys addressing the first research question use a variety of frameworks to base their survey questions on. A first framework are the software architecture viewpoints by Rozanski and Woods [25]: the Context, Functional, Information, Concurrency, Development, Deployment and Operational viewpoints. Paper [10] reports that the Information and Functional viewpoints are modelled by the highest number of participants (99% and 96% respectively). Other viewpoints that were modelled by many participants are the Deployment, Concurrency and Development viewpoints (75%, 66% and 64% respectively). The Operational viewpoint was only modelled by 29% of the participants. Paper [13] presents an evaluation of 113 architectural languages. This paper confirms that the Logical and Information viewpoints are supported by the majority of the architectural languages (91% and 78% respectively). The concurrency viewpoint is supported by 45% of the languages, but the Development, Deployment and Operational viewpoint are supported by a relatively few architectural languages (26%, 15% and 10% respectively). Two additional viewpoints discussed in this paper are the behavioral viewpoint which is supported by 46% of the architectural languages and the Physical viewpoint which is supported by 15% of the architectural languages. The viewpoints that are supported by the most languages according to paper [13] are also the viewpoints that are the most modeled by practitioners according to paper [10].

Two papers designed their survey according to phases in the development lifecycle. While not referring explicitly to the Rational Unified Process (RUP) framework, we used it to unify the results of two papers. The RUP framework [26] contains 6 engineering disciplines (Business modelling, Requirements Engineering, Analysis and Design, Implementation, Testing, Deployment) and three supporting disciplines (Configuration and change management, Project management, Environment). Paper [4] is a survey of open source software developers. In 68% of the projects that use UML for design, the UML models were implemented completely or with minor changes. Paper [20] is a survey in the embedded systems industry. The disciplines where modelling was used the most are Analysis and Design (89.5%), Implementation (74.4%) and Requirements Engineering (64.1%).

Table 2. Software development activities grouped by the dimension of the RE framework.

RE dimension	paper	What are models used for?	% using models frequently for this activity
HIGH frequency of use			
negotiation	[22]	discuss with colleagues	0,790
documentation	[20]	Documentation generation	0,768
	[22]	visualize an idea or concept	0,750
	[4]	Documentation (e.g.: reverse engineered)	0,712
elicitation	[4]	Design/architecture for (existing/new) systems parts	0,705
	[22]	help me think, sketch a thought	0,700
	[20]	Understanding a problem	0,670
-	[20]	Code generation	0,762
	[15]	Simulation	0,681
	[15]	Code generation	0,664
MEDIUM frequency of use			
negotiation	[22]	communicate with clients	0,440
	[20]	Communication	0,405
documentation	[20]	Documenting designs	0,578
	[15]	Information/documentation	0,531
	[3]	transcribing a design into digital format	0,517
	[22]	document a system or code	0,450
elicitation	[22]	capture domain knowledge	0,570
	[3]	developing a design	0,551
	[22]	design systems or code	0,550
	[22]	capture technical requirements	0,430
	[22]	capture client requirements	0,470
	[3]	brainstorming possible designs	0,448
LOW frequency of use			
negotiation	[22]	negotiate consensus	0,260
	[22]	define contract (model is part of contract)	0,180
documentation	[22]	reconstruct knowledge from source code etc.	0,200
-	[15]	Test case generation	0,398
	[20]	Test-case generation	0,384
	[15]	Structural consistency checks	0,381
	[20]	Model-to-Model (M2M) transformation	0,373
	[22]	generate prototype code	0,350
	[3]	generating code (code editable)	0,344
	[15]	Traceability	0,336
	[15]	Behavioral consistency checks	0,336
	[3]	prototyping a design	0,322
	[3]	generating all code	0,310
	[15]	Timing analysis	0,283
	[22]	generate production code	0,280
	[22]	create a DSL	0,260

(*continued*)

Table 2. (*continued*)

	[15]	Safety compliance checks	0,230
	[15]	Formal verification	0,221
	[22]	look up product details	0,200
	[4]	Verification	0,178
	[20]	Model simulation	0,151
	[15]	Reliability analysis	0,142
	[4]	Refactoring	0,141
	[4]	Code generation	0,129
	[4]	Models are test data	0,060

Finally, some surveys asked questions that were not based on a framework. To be able to compare the answers from this group, we grouped the topics from surveys [3, 4, 15, 20, 22] according to the "Dimensions of RE" framework [27] as shown in Table 2. Table 2 lists the topics with high ($\geq$70%), medium ($<$70% and $\geq$40%) and low ($<$40%) frequency of use separately. Papers [15] and [20] focus on software engineering for embedded systems and report that the main purposes are code generation and documentation. The activities for which more than 70% of participants indicate that models are frequently used, lead to the conclusion that the main purposes are documentation and elicitation of requirements and being used for negotiation to a lesser extent. The survey in [2] is discussed separately since it is the only survey where participants chose on average just one purpose for their models. Therefore, we cannot compare the percentages reported in paper [2] to the other surveys. The activities in paper [2] were coded into six purposes. We mapped the first five onto the RE dimensions. The sixth coded purpose was Requirements Engineering. Similar to the other surveys, the majority of participants chose an elicitation activity: 42.6% participants selected an elicitation activity, 28.7% of participants selected a documentation activity, 26.9% selected a negotiation activity and 11.1% of participants selected an activity that the authors classified under the general category "Requirements Engineering".

Paper [18] is a systematic literature review on BPMN that concludes that documentation is the main activity that BPMN is used for. A final observation is that the purposes that fall outside the three "Dimensions of RE" score low (Code generation, creating a DSL, prototyping design, reconstruct knowledge from source code…).

4.2 RQ2 - Which Other Modelling Languages Are in Use Next to BPMN and UML and How Often Are They Used?

For this research question, we only considered general papers that do not focus on a specific language: surveys [2, 3, 14–16, 20, 22]. In 6 of the 7 papers, UML is reported as the most used modelling language, with at least 41.6% of the participants in each of the surveys using it. Paper [3] shows slightly different results. Here, ERD is the most popular modelling language with 40% of participants using it very often, followed by structured design models which were used very often by 38% of the participants. The use of UML is split into three categories in this survey. 34% of the participants use UML

2.* very often, 33% of the participants use any version of UML very often and 27% of the participants use UML 1.* very often. Paper [3] is the only paper with American authors and it reports that 70% of the participants are from the USA or Canada as shown in Table 1.

In papers [10, 14, 20, 23], participants were also asked about the UML diagram types they use the most. Across all papers alike, class diagrams are among the top 2 most used diagram types.

According to papers [3, 14, 15, 20, 22] a secondary group of languages are Domain Specific Languages. The percentage of use varies from 8% [15] to 36% [14].

BPMN is mentioned by 2 papers that report contradicting results. Paper [2] reports 34,6% of the 108 participants using BPMN and paper [21] reports only 12 of the 96 participants (12,5%) using BPMN. A notable difference is that in paper [22] 40% of the respondents are from Germany while the population surveyed in paper [2] is more diverse. Another difference is that paper [22] specifically targets Industrial Software Development. In five surveys [2, 14, 15, 20, 22], 5% to 21.2% of the participants use SysML and in two surveys [14, 20], 16.9% to 19% of the participants use UML profiles.

4.3 RQ3 - Which Problems/Difficulties/Requirements Do Practitioners Have with Model-Driven Engineering?

Out of the 18 selected papers, 14 discuss this topic. Ten of these are surveys and asked about the requirements that practitioners have for modelling languages, shortcomings in model-driven engineering, disadvantages of modelling languages and tools and inhibitors for successful adoption of model-driven engineering. Six surveys [2, 3, 15, 19–21] asked their participants to answer using a 5-point Likert scale. The approach described in RQ1 is applied. For example, in the survey of paper [20], regarding the problem 'difficulties with model-level debugging', 6% of participants indicated 'strongly agree' and 68% of the participants indicated 'agree'. Therefore, we summarize this as 74% of participants considering 'difficulties with model-level debugging' a problem of MDE environments or tools.

These percentages can now be compared to the results of the surveys not using Likert scale data [14, 16, 17, 24]. For these 10 surveys, the problems that were identified by more than 60% of the participants were consolidated. The next step is to code the different responses in order to summarize them. This was done by mapping the responses onto one of two complementing frameworks: the Physics of Notations framework [28] and Lago's framework [29]. The Physics of Notations framework refers to characteristics of a Notation in terms of its ease of understanding, while Lago's framework provides an overview of practical requirements for architectural languages based on a survey with practitioners. The remaining problems that could not be mapped to one of these two frameworks were sorted in categories under 'Other' in Table 3.

The problems can be summarized in Table 3. The 5 problems that were identified by three or four separate surveys are highlighted in the table.

Table 3. Practitioners' issues with model-driven engineering.

Issues	[2]	[14]	[15]	[3]	[16]	[17]	[19]	[20]	[21]	[24]	Total
Lago			**3**	**5**	**1**			**2**	**3**		**14**
(Lago) automated analysis					1			2			3
(Lago) extensibility and customization				3							3
(Lago) large-view management				1					1		2
(Lago) programming framework			1								1
(Lago) support for collaboration			1						1		2
(Lago) support for software architecture-centric design									1		1
(Lago) support for versioning			1								1
(Lago) support to specify nonfunctional properties/(Lago) automated analysis				1							1
Other		**1**	**9**	**7**			**8**	**2**	**1**		**28**
-				1			1				2
cost			1								1
cultural/social/management inhibitors							4				4
customer support by tool manufacturer								1			1
external tool integration/model transformation & export			5	1					1		7
high effort in acquiring MBSE skills		1	1	1			3				6
high effort in using tool			2	2				1			5
synchronization between model and code				2							2
PoN	**9**			**1**		**9**		**1**		**3**	**23**
(PoN) cognitive fit	1					1		1		2	5
(PoN) cognitive fit/(PoN) cognitive integration										1	1
(PoN) cognitive integration	1					1					2
(PoN) complexity management	1					1					2
(PoN) dual coding	1					1					2
(PoN) graphical economy	1					1					2
(PoN) perceptual discriminability	1					1					2
(PoN) semantic transparency	1					1					2
(PoN) semiotic clarity	1					1					2
(PoN) visual expressiveness	1			1		1					3

Finally, there are four papers addressing this research question that don't include a survey [11–13, 18]. In paper [11] the authors provide a list of requirements for tools that practitioners find important. Paper [12] is a systematic literature review that list quality criteria used in primary studies and their frequency of appearance. Paper [13] lists the requirements for modelling languages that are part of Lago's framework. Paper [18] is a systematic literature review on BPMN that reports two common disadvantages of BPMN: "Less than 20% of the BPMN vocabulary is common used" and "Utilization in

specific domain can be difficult". The problems mentioned by these four papers can be mapped to the above mentioned problems, whereby each problem is mentioned at most once or twice.

Table 4. Areas of benefits of model-driven engineering according to practitioners

Area of benefits	[14]	[15]	[3]	[4]	[19]	[20]	[22]	Total
RUP discipline - Analysis and Design	1	2	4	5	5	1	1	19
Quality management		7			2	3	1	13
RUP discipline - Implementation		4	4	1		3		12
RUP discipline - Requirements Engineering		2	1		4	1	1	9
RUP discipline - Deployment		2	1			2	1	6
General	1		2				2	5
RUP supporting discipline - Project management						1	2	3
Cost savings		1				1		2
RUP discipline - Testing						2		2
RUP discipline - Business modelling							1	1
Accessibility/support	1							1
RUP discipline - Implementation & Testing							1	1

4.4 RQ4 - What Are the Benefits of Model-Driven Engineering According to Practitioners?

Nine of the 18 selected papers discuss this topic, out of which 8 are surveys: [3, 4, 14–16, 19, 20, 22]. Again, the data collection approaches are quite varied. From each survey, we selected the benefits that were confirmed by at least 60% of the participants. For surveys using a Likert scale, the same approach was used as in RQ3. We collected all these benefits and mapped them onto a discipline of the Rational Unified Process framework presented. Some of the benefits could not be placed in a discipline. They were put in one of the following categories: Quality management, General benefits, Accessibility/support and Cost savings. Most of the benefits fell under these five categories:

- RUP discipline – Analysis and Design (e.g.: making complex design decisions, brainstorming & collaboration, understanding & explaining a system's behavior etc.)
- RUP discipline – Requirements Engineering (requirements analysis, traceability of requirements, etc.)
- Quality management (integrity, reliability, etc.)
- RUP discipline – Implementation (code generation, reusability, shorter development time, etc.)
- RUP discipline – Deployment (maintainability, modifications, etc.)

Paper [18] is the only paper without a survey. Three of the six benefits of BPMN that were identified in this paper can be categorized under 'accessibility/support'. The others fall under 'RUP discipline – Analysis and Design', 'RUP discipline – Implementation'

and 'RUP discipline – Business modelling'. Table 4 presents the results in descending order of number of mentioned benefits.

5 Discussion

In general, there is a lack of unified terminology used in the papers to describe the issues that practitioners face when they use modelling languages. Many papers also do not provide enough details about the demographics of the survey's sample, or do not provide the details about the questionnaire used. All these elements make it hard to compare and summarize results across different surveys.

For RQ1, we noticed that the modelling purpose 'Documentation (e.g.: reverse engineered)' scores high in paper [4], but the similar purpose 'reconstruct knowledge from source code etc.' in paper [22] scores low. This could be due to differences in the populations of the surveys. Paper [4] is a survey with 485 respondents across 91 different countries, and paper [22] is a survey with 96 participants of which 40% are from Germany. Similarly, For RQ2, the reported results in the papers varied. Also these differences in results could be due to differences in the population or a different focus/domain of expertise of the respondents.

When contrasting the results of RQ1 and RQ4, we see that in RQ1 we concluded that modelling is mostly used for the 'Analysis and Design', 'Implementation' and 'Requirements Engineering' discipline of the RUP framework [26]. These disciplines were also considered among the five main benefits of modelling in RQ4.

When contrasting the results of RQ1 and RQ3, the results for RQ1 reveal that besides the information and functional viewpoint, modelling is frequently used for the other viewpoints as well, and -considering the lifecycle phases- not only for the early phases of analysis and design and requirements engineering, but also for implementation. Given the high frequency of goals in the range of communication, the fact that of all 65 reported problems in Table 4, 23 (35%) are PoN -related problems, is a significant issue.

When contrasting the results of RQ3 and RQ4, we see that while some of the main benefits of modelling are collaboration and understanding and explaining the system's behavior according to the results of RQ4, RQ3 indicates that there are still many practitioners that experience understandability issues related to the PoN framework. This contradiction could be due to the varying levels of experience and fields of expertise of the participants of the surveys. It is therefore important to keep improving the understandability of modelling languages to account for the diversity in users.

Due to the diversity in scope, perspective and reporting in the papers selected for this review, it is not possible to draw conclusions about the evolution of Model-Driven Engineering in the last five years. When contrasting the results of this literature review with the related research, we find that in paper [5], which was published in 2015, the authors identify the complexity of modelling tools as one of the seven main issues of the model-driven engineering field. In particular, they discuss the lack of usability of the tools and the problems with tool interoperability. Based on the results of RQ3 these are still big issues that practitioners face. A second problem identified in paper [5] that practitioners still have is the lack of automated analysis of the quality of the models. This problem also surfaced in the results of RQ3. One of the results reported in paper [6] is

that OWL is the most used ontology related language, with very few papers discussing UML. In our results for RQ2, we see that UML is the most used modelling language in practice. The main benefits of ontologies in paper [6] are related to improvements in quality and requirements management. This corresponds to the benefits found in RQ4.

The results presented in this paper are also subject to some validity threats. The first threat to the validity of this paper is the limitation of the search. While we followed Kitchenham's guidelines and used two databases containing millions of papers, it is nevertheless possible that some papers were missed by the query. We nevertheless consider the current set sufficiently large to allow for an acceptable level of confidence in the results. Assuming that all surveys have been issued to disjoint populations, in total a population of around 2500 practitioners is reported on in this meta-survey.

A second threat to validity is the publication lag. The survey that was conducted most recently is from 2018, but often the surveys are older or the period wherein the survey was conducted is not reported. While we were aiming for reviewing the period of 2015–2020, in practice, the data itself relates to the period of 2013–2018.

A Final threat to validity is that the surveys were based on different frameworks and often incomplete in the reporting of the demographic information and in the explanation of their questions and answer options. While we attempted to be as careful as possible when matching results, we nevertheless were sometimes required to make certain interpretations in order to summarize the results.

6 Conclusion

This meta-review of survey and review papers published in the last 5 years provides an overview of how practitioners use modelling languages and what they perceive to be the benefits and issues of Model-Driven Engineering. We found that modelling is very highly used in the functional and information viewpoint, during the design phase of the software development lifecycle and for documentation and elicitation of requirements. The most frequently used modelling language is UML and the most frequently used UML diagram types are class diagrams. While practitioners experience benefits of modelling for analysis & design, requirements engineering, quality management, implementation and deployment, they still struggle with a number of significant issues. In particular, the most common reported problems are external tool integration/model transformation & export, cognitive fit, visual expressiveness, high effort required in acquiring skills, automated analysis and high effort required in using tools.

The issues identified in this review may serve for the identification of a research agenda. In particular, a first point would be to identify possible improvements for usability of the tools. A second closely related issue to address is the interaction, integration or interoperability of different tools. Finally, a third issue we want to investigate further is how the understandability of models can be enhanced, either by improving current languages or by providing additional help for non-proficient users.

References

1. Soley, R.: Model Driven Architecture Model Driven Architecture Preface: OMG's Accomplishments, p. 308 (2000)

2. van der Linden, D., Hadar, I., Zamansky, A.: What practitioners really want: requirements for visual notations in conceptual modeling. Softw. Syst. Model. **18**(3), 1813–1831 (2018). https://doi.org/10.1007/s10270-018-0667-4
3. Badreddin, O., Khandoker, R., Forward, A., Masmali, O., Lethbridge, T.C.: A decade of software design and modeling: a survey to uncover trends of the practice. In: Proceedings - 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS 2018, pp. 245–256 (2018). https://doi.org/10.1145/3239372.3239389
4. Ho-Quang, T., Hebig, R., Robles, G., Chaudron, M.R.V., Fernandez, M.A.: Practices and perceptions of UML use in open source projects. In: Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track, ICSE-SEIP 2017, pp. 203–212 (2017). https://doi.org/10.1109/ICSE-SEIP.2017.28
5. Giraldo, F.D., España, S., Giraldo, W.J., Pastor, O.: Modelling language quality evaluation in model-driven information systems engineering: a roadmap. In: International Conference on Research Challenges in Information Science, pp. 64–69 (2015)
6. Dermeval, D., et al.: Applications of ontologies in requirements engineering: a systematic review of the literature. Requirements Eng. **21**(4), 405–437 (2015). https://doi.org/10.1007/s00766-015-0222-6
7. Kitchenham, S., Charters, B.: Guidelines for performing systematic literature reviews in software engineering. Technical Report Ver. 2.3 EBSE, vol. EBSE-2007-, no. School of Computer Science and Mathematics, p. 65 (2007). https://www.elsevier.com/__data/promis_misc/525444systematicreviewsguide.pdf
8. Petre, M.: "No shit" or "Oh, shit!": responses to observations on the use of UML in professional practice. Softw. Syst. Model. **13**(4), 1225–1235 (2014). https://doi.org/10.1007/s10270-014-0430-4
9. Baltes, S., Diehl, S.: Sketches and diagrams in practice. In: Proceedings of the ACM SIGSOFT Symposium on the Foundations of Software Engineering, pp. 530–541 (2014). https://doi.org/10.1145/2635868.2635891
10. Ozkaya, M., Erata, F.: A survey on the practical use of UML for different software architecture viewpoints. Inf. Softw. Technol. **121**, 106275 (2020). https://doi.org/10.1016/j.infsof.2020.106275
11. Ozkaya, M.: Are the UML modelling tools powerful enough for practitioners? A literature review. IET Softw. 13(5), 338–354. Institution of Engineering and Technology (2019). https://doi.org/10.1049/iet-sen.2018.5409
12. Awadid, A., Nurcan, S., Ayachi Ghannouchi, S.: On leveraging the fruits of research efforts in the arena of business process modeling formalisms: a map-driven approach for decision making. Softw. Syst. Model. **18**(3), 1905–1930 (2018). https://doi.org/10.1007/s10270-018-0689-y
13. Ozkaya, M.: The analysis of architectural languages for the needs of practitioners. Softw. Pract. Experience **48**(5), 985–1018 (2018). https://doi.org/10.1002/spe.2561
14. Ozkaya, M.: Do the informal & formal software modeling notations satisfy practitioners for software architecture modeling? Inf. Softw. Technol. **95**, 15–33 (2018). https://doi.org/10.1016/j.infsof.2017.10.008
15. 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. **17**(1), 91–113 (2016). https://doi.org/10.1007/s10270-016-0523-3
16. Ozkaya, M.: What is software architecture to practitioners: a survey. In: MODELSWARD 2016 - Proceedings of the 4th International Conference on Model-Driven Engineering and Software Development, pp. 677–686 (2016). https://doi.org/10.5220/0005826006770686
17. Saleh, F., El-Attar, M.: A scientific evaluation of the misuse case diagrams visual syntax. Inf. Softw. Technol. **66**, 73–96 (2015). https://doi.org/10.1016/j.infsof.2015.05.002

18. Kocbek, M., Jošt, G., Heričko, M., Polančič, G.: Business process model and notation: the current state of affairs. Comput. Sci. Inf. Syst. **12**(2), 509–539 (2015). https://doi.org/10.2298/CSIS140610006K
19. Huldt, T., Stenius, I.: State-of-practice survey of model-based systems engineering. Syst. Eng. **22**(2), 134–145 (2019). https://doi.org/10.1002/sys.21466
20. Akdur, D., Garousi, V., Demirörs, O.: A survey on modeling and model-driven engineering practices in the embedded software industry. J. Syst. Architect. **91**, 62–82 (2018). https://doi.org/10.1016/j.sysarc.2018.09.007
21. Farias, K., Gonçales, L., Bischoff, V., da Silval, B., Guimarães, E., Nogle, J.: On the UML use in the brazilian industry: a state of the practice survey. In: Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE, pp. 372–375 (2018). https://doi.org/10.18293/SEKE2018-183
22. Störrle, H.: How are conceptual models used in industrial software development? A descriptive survey. In: ACM International Conference Proceeding Series, vol. Part F128635, pp. 160–169 (2017). https://doi.org/10.1145/3084226.3084256
23. Fernández-Sáez, A.M., Caivano, D., Genero, M., Chaudron, M.R.V.: On the use of UML documentation in software maintenance: results from a survey in industry. In: MODELS, pp. 292–301 (2015)
24. Monsalve, C., April, A., Abran, A.: Business Process Modeling with Levels of Abstraction (2015)
25. Rozanski, N., Woods, E.: Software Systems Architecture. https://www.viewpoints-and-perspectives.info/home/viewpoints/. Accessed 16 Mar 2021
26. Kruchten, P.: The Rational Unified Process: An Introduction, 3rd edn. Addison-Wesley, Boston (2000)
27. Pohl, K.: The Requirements Engineering Framework. Springer, Berlin Heidelberg (2010)
28. Moody, D.: The physics of notations toward a scientific basis for constructing visual notations in software engineering. IEEE Trans. Softw. Eng. **35**(6), 756 (2009). https://doi.org/10.1109/TSE.2009.67
29. Lago, P., Malavolta, I., Muccini, H., Pelliccione, P., Tang, A.: The road ahead for architectural languages. IEEE Softw. **32**(1), 98–105 (2015)