Software and Systems Modeling (2024) 23:249–265
https://doi.org/10.1007/s10270-023-01117-1

REGULAR PAPER

Modelling guidance in software engineering: a systematic literature review

Shalini Chakraborty[1] · Grischa Liebel[1]

Received: 15 June 2022 / Revised: 12 June 2023 / Accepted: 27 June 2023 / Published online: 17 July 2023

Abstract

Despite potential benefits in Software Engineering, adoption of software modelling in industry is low. Technical issues such as tool support have gained significant research before, but individual guidance and training have received little attention. As a first step towards providing the necessary guidance in modelling, we conduct a systematic literature review to explore the current state of the art. We searched academic literature for guidance on model creation and selected 35 papers for full-text screening through three rounds of selection. We find research on model creation guidance to be fragmented, with inconsistent usage of terminology, and a lack of empirical validation or supporting evidence. We outline the different dimensions commonly used to provide guidance on software and system model creation. Additionally, we provide definitions of the three terms modelling method, style, and guideline as current literature lacks a well-defined distinction between them. These definitions can help distinguishing between important concepts and provide precise modelling guidance.

Keywords Modelling styles · Modelling training · Modelling guidance · Modelling method · Systematic literature review

1 Introduction

Despite the potential benefits of using models in Software Engineering (SE), adoption has been low, typically pointing to issues such as tool support, organisational resistance, and a lack of guidance/training [39–41, 53, 54, 67, 68, 97]. Technical issues have historically received substantial attention in the modelling community, seen for instance by the large amount of tools for modelling and Model-Based Engineering (MBE). However, work on guiding individuals in creating models receives typically only marginal attention. For instance, Schätz et al. state that "methodical guidelines are missing how to use suitable abstractions of (parts of) a cyber-physical systems at varying level of detail to enable the engineering of those systems with a sufficient level of confidence concerning the quality of the implemented systems" [84].

To better understand shortcomings in the literature on guidance and training in modelling, this paper reports on a Systematic Literature Review (SLR) [48] surveying work on guidelines, styles, or training approaches on creating software and system models in SE. We address the following research question **RQ: What kind of guidance exists in SE literature on model creation?** To answer the question, we collected a total of 8604 papers starting from 1998 when the first UML conference was held. In addition to this database search, we conducted a second search using more general terminology and random sampling among the 71450 resulting papers, and snowballing [99] in the area of Business Process Modelling to complement our findings.

After several rounds of exclusion, we analysed 35 papers, finding that systematic guidance to create software and system models is limited in SE literature. Existing work proposes guidance for specific domains or problems, but generally lacks validation and/or empirical evidence. In BPM, which partially overlaps with SE research, there exists initial empirical work investigating modelling styles and how cognitive processes affect model creation. We further find that terminology is used inconsistently in the literature, with terms such as method, guidelines, or style being used inconsistently and seemingly arbitrary. To address this, we define *modelling method*, *modelling guideline*, and *modelling style*.

Communicated by Juergen Dingel.

✉ Shalini Chakraborty
shalini19@ru.is

Grischa Liebel
grischal@ru.is

[1] Reykjavik University, Menntavegur 1, 102 Reykjavík, Iceland

Our SLR shows that further work is necessary in several directions in the modelling community. First, we see that work on modelling needs to be conducted more systematically. Our definitions presented in this paper can help supporting reaching a common terminology. Secondly, empirical evidence and validation is needed. Currently, we find mainly solution proposals that are at best substantiated through simplified examples. The data set accompanying this paper is published on Zenodo.[1]

The rest of the paper is structured as follows. In Sect. 2, we discuss the lack of guidance in software modelling through related work. Section 3 explains the methodology of our conducted SLR and validity threats. We present the results in Sect. 4, followed by an overall discussion of the findings in Sect. 5. Finally, we conclude the paper in Sect. 6 with future plans to use the findings suitably in software modelling.

2 Related work

Text books on UML and object-oriented design commonly refer to heuristics for creating UML diagrams, e.g. [9, 79]. For instance, Abbott's [1] strategy for identifying system objects by scanning informal descriptions for nouns is typically referred to. These and similar heuristics are common for some diagram types, e.g. class diagrams, but often lacking for behaviour. Additionally, there is, to our knowledge, no empirical evidence that these kind of heuristics support system analysis or other modelling tasks.

Previous work has investigated how to design modelling languages in a *good* way, by studying both the notations themselves, e.g. [69, 82, 83, 92], and how models or diagrams are read and understood by human subjects, e.g. [55, 60, 89–91]. Moody [69] defines in his Physics of Notation a number of principles for cognitively effective notations. However, the value of the Physics of Notation is unclear [82, 83, 92]. For diagram understanding, Störrle et al. [60, 89–91] find that the size and layout quality of UML diagrams directly impacts understanding. Outside of SE, Lohmeyer and Meboldt [55] find that there are different patterns individuals follow when reading engineering drawings. Understanding how models are read and understood can help to derive heuristics for the quality of a model. However, this knowledge might not help guiding individuals to create *good* models. That is, while the desired outcome might be known, the process to arrive at this outcome remains unclear.

A way to describe how to create models is to provide reusable parts of a solution, or patterns, for different kinds of models. In the area of patterns, there has been substantial research during the late 1990 s, e.g. [7, 20–22, 29]. The well-known Gang-of-Four design patterns [29] provide reusable patterns that should help to improve the design of object-oriented systems. Douglass describes patterns for real-time behaviour diagrams [20], and Bordeleau [7] proposes patterns that should aid a designer to move from a scenario-based specification to a state-based specification. Dwyer et al. [21, 22] propose property specification patterns, reusable logic constructs that can be re-used to specify properties in specifications, specifically the order and the occurrence of system events. The work on patterns has evolved substantially since then, especially in the area of real-time systems, e.g. [3, 33, 61]. However, patterns describe only a part of model creation, namely *what* should be part of the model. There are several other aspects that could be of interest during model creation, e.g. in which order to create parts of the model, what to include and what to exclude, how to decide on a reasonable level of abstraction, and individual differences that affect this process. A promising area of interest is AI-assisted modelling guidance. Recent studies have shown the application of AI in model creation, leading to enhanced functional objectives. Cámara et al. [12] describe the role of generative AI models such as large language models (LLM) in software modelling regulation. Chaaben et al. [13] propose using LLM to improve domain modelling activities. A recent special issue on AI-enhanced model-driven engineering [10] also summarises several contributions [81, 93, 96] to support modellers with better search facilities, automated integration and better learning experience from independent metamodels. While clearly relevant for future modelling support, we excluded AI-assisted work in our literature review, as these approaches as of now do not provide explicit and explainable guidance, but rather simply improve the resulting models.

In terms of guiding individual modellers and understanding the differences between individuals when creating models, Pinggera et al. [15, 74, 75] investigate BPM model creation. The authors present an exploratory study with 115 students exploring how students create BPM. They find distinct styles of modelling by performing cluster analysis on phases of model comprehension (where a modeller builds a mental model out of the domain behaviour), modelling (where a modeller maps the mental model to actual modelling constructs), and finally reconciliation (where a modeller acknowledges the process model).

3 Research methodology

In this paper, we aim to answer the following research question:

RQ: What kind of guidance exists in SE literature on model creation?

To do so, we conduct an SLR, a secondary form of study used to identify and evaluate available research relevant to a certain

[1] https://doi.org/10.5281/zenodo.7685694.

research question or topic of interest [48]. SLRs are useful to obtain a detailed picture of a research area, and to integrate existing evidence [49]. They are increasingly common in SE [47], and specifically in the modelling community in the last decade, e.g. [30, 57, 70, 71, 87]. We chose an SLR over a mapping study, as we were interested in the detailed picture, not the broad coverage of a research area a mapping study provides [72].

We followed the steps proposed by Kitchenham and Charters [48] to perform our SLR. An overview of this process is depicted in Fig. 1.

In the following, we discuss the review steps in detail, starting with the search and extraction process.

3.1 Search and extraction

We started the review with preliminary research steps, by manually reading through the last five years (2015 to 2019 at the time of starting the review) of papers appearing in *MODELS conference* (http://modelsconference.org/) and *SoSyM journal* (https://link.springer.com/journal/volumesAndIssues/10270), the two prime venues for SE modelling research. Note that the automated keyword search was later updated until January 2023. We observed that papers use the terms *framework*, *guidelines*, *method*, *approach*, and *pattern*. Hence, we used these terms in our initial search string. We searched within modelling research in SE, thus resulting in the following initial search string:

(“modelling” OR “modelling” OR “model-driven” OR “model-based”)
AND (“framework” OR “guidelines” OR “method” OR “approach” OR “pattern”)
AND (“software engineering”)

Applying the initial search string on Scopus, IEEE Xplore, ACM Digital Library, and Web of Science resulted in 17713 papers, a result we deemed infeasible to analyse. Therefore, we decided to make a number of adaptations to the search string (Step 1 in Fig. 1). First, we added three terms we had missed initially: *creation*, *training*, and *styles*. Adding creation and training is directly motivated by our research question and goal, while adding styles is motivated by existing work in BPM, i.e. by Pingerra et al. [74]. Secondly, to reduce the amount of found papers, we decided to remove *method*, *approach*, and *pattern*. This resulted in the following final search string **S1**:

(“modelling” OR “modelling” OR “model-driven” OR “model-based”)
AND (“guidelines” OR “training” OR “styles” OR “creation”)
AND (“software engineering”).

To not entirely discard relevant search terms, we constructed a second search term **S2** as follows:

(“modelling” OR “modelling” OR “model-driven” OR “model-based”)
AND (“approach” OR “process” OR “method” OR “template”)
AND (“software engineering”).

S2 contains terms that are heavily conflated in SE research. That is, we expect less relevant results from **S2**. Therefore, combining **S1** and **S2** would have required to do random sampling on the whole data set, and resulted in a higher chance to miss relevant papers. We decided to extract all papers found through **S1**, while randomly sampling from **S2** up to a feasible amount of papers.

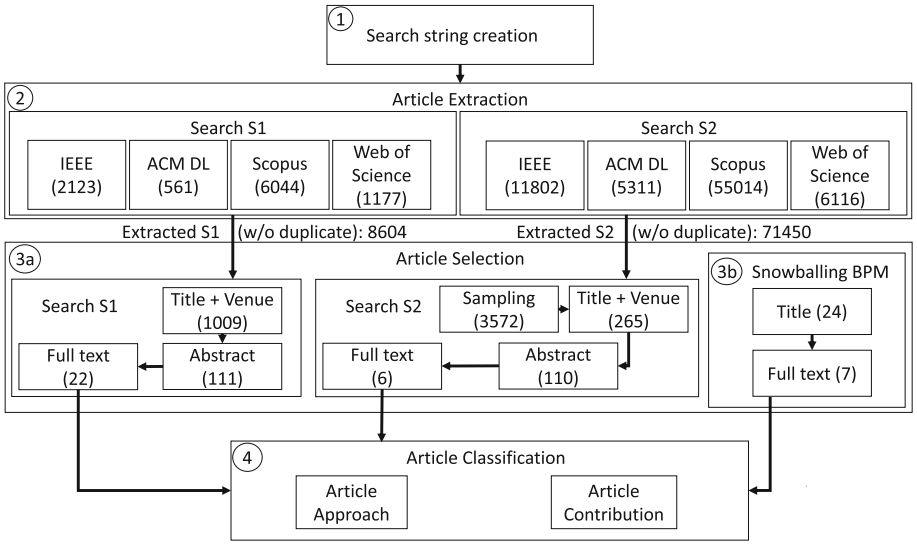
We then adapted the two search strings to the four search engines we used, further limiting the search to English papers published from 1998, as the first UML conference was held in that year [6]. We followed the UK spelling of modelling throughout our review, but we used both UK and US spelling in our search string. Additionally, we added *model-driven* and *model-based* to target papers specifically aimed towards model-driven and model-based engineering (MDE & MBE) as well.

We searched in title, abstract, and keywords for Scopus, IEEE Xplore, and ACM Digital Library, and in all fields offered by Web of Science (Step 2 in Fig. 1). For ACM Digital Library, we had to further exclude the keyword *creation*, since one of the standardised ACM keywords (“CCS concepts”) is coined “Software creation and its management.” As it was not possible to exclude the CCS concepts from the search, we would get all papers that used this keyword prior to removing *creation* from the search. The search for **S1** yielded 9905 papers, 8604 after removing duplicates, covering the years from 1998 until January 2023. The search for **S2** yielded a staggering 78243 papers, 71450 after duplicate removal. We decided that sampling 5 per cent of these papers would be reasonable, as it both is a substantial amount of papers, and one we could realistically process without experiencing fatigue bias. That is, we randomly selected 3572 papers from the results of **S2**.

3.2 Paper selection

We performed paper selection in three rounds with different inclusion and exclusion criteria for each round (Step 3a in Fig. 1). In the beginning of each round, both authors went through a random 5 per cent of the remaining papers in **S1** to determine inter-rater agreement on the respective inclusion/exclusion criteria. We used Fleiss kappa to measure our agreement, a statistical measure used to evaluate the reliability of an agreement between a fixed number of raters [46]. We decided on a threshold value of $\kappa > 0.7$ as a minimum agreement to continue the selection process. This is a compromise between Fleiss [27] suggestion to understand kappa values of > 0.75 as excellent, and suggestions of Landis and Koch

Fig. 1 Review Process



[50] to label agreements between $0.61 - 0.80$ as substantial, both of which have been criticised for being arbitrary [36]. In case of lower values, we would discuss our disagreements, potentially refine the criteria, and then re-run the process with another random selection of papers.

In the first round, we excluded papers based on title and the paper venue. We needed three rounds to reach a Fleiss Kappa of $\kappa \approx 0.73$. Ultimately, we excluded papers in the first round if they matched any of the following criteria. Note that we applied the criteria extremely conservative at this stage, to not exclude relevant papers.

1. It is clear from the title that modelling/diagrams are not a contribution.
2. The conference/journal is from a different field of science.
3. The paper is not peer reviewed.
4. The paper discusses properties of modelling languages or compares languages.
5. The paper has a prose title, or clearly points into a different direction than investigated in our review.
6. The paper is about descriptive models of a scientific or real-life concept, as opposed to a system. For instance, process improvement models such as CMMI, or models of course curricula.
7. Modelling in a different domain of computer science, e.g. database models, machine learning models, or neural network models.
8. Extended abstracts for tool demos or posters.
9. Meta studies such as literature reviews, or comparative studies such as controlled experiments.

After applying these criteria, we were left with 1009 papers of S1 and 265 papers of S2 for the second round of selection.

In the second round, we considered the paper abstracts. After two rounds of discussions and adapting the exclusion criteria, we reached a kappa of $\kappa \approx 0.79$.

The exclusion criteria for excluding based on abstracts are as follows.

1. Creation of models is not clearly mentioned in the abstract.
2. It is clear from the abstract that the paper focuses on any of the following.
   (a) Language design/meta modelling
   (b) Model transformations
   (c) Secondary studies such as SLRs
   (d) Controlled experiments or other comparative studies
   (e) Descriptive models of a scientific or real-life concept, as opposed to a software system
   (f) Architectural styles or design patterns
   (g) AI-based modelling guidance

We excluded language design/meta modelling and model transformations, as we consider them special cases of modelling aimed at the meta modelling level. That is, they aim at creating models that describe languages and/or domains, and not software and/or systems. As such, it can be expected that the resulting guidance would be substantially different from models that describe systems. For similar reasons, we excluded architectural styles and design patterns, as they present tried solutions, but do not provide guidance in any other way. Finally, we excluded papers that use AI to improve or create models, as they lack explicitly stated guidance [23, 51]. As such, while valuable, their output is often not explainable. For example, we found papers like [45, 77] where modelling frameworks are mentioned, but no particular steps for modelling are introduced. The papers we found through

our search that include AI-related guidance mostly detailed with the model's impact on systems rather than creation. Applying these criteria, 111 papers from S1 and 110 papers from S2 remained for full-text reading.

After the second exclusion round, we found three papers from the area of BPM that fit into the scope of our review. However, they were not included through the search, as they did not fit the software engineering keyword. Nevertheless, to be able to compare SE-specific research with promising work in BPM (but outside SE), we took those three papers as a starting set for backwards and forwards snowballing [99] (Step 3b in Fig. 1). That is, we searched both the references and citations of the three papers, including papers that fit our scope. We used the same exclusion criteria as for the second round resulting in a set of 24 papers from BPM.

In the third and last round, we extracted and read the full-text of the 111 papers from S1 and 110 papers from S2, as well as the 24 papers from the snowball search in BPM. We used the same exclusion criteria as for the second round. We applied the criteria to each paper's full text. Furthermore, we excluded one paper, as an extended version of that paper was already included in the review.

This resulted in a final set of 28 papers from S1 and S2, and 7 papers from the snowball search left for analysis.

3.3 Data extraction

The final 35 papers we analysed are listed in Table 7 for S1, Table 9 for S2, and Table 8 for the snowballing search in Appendix A. We provide the citations of all 35 papers in Appendix A. Both authors analysed all 35 papers, then discussed their disagreements. We did not calculate inter-rater agreement at this step. We deemed this unnecessary, as re-running the analysis several times would likely result in the authors memorising the discussions, not an actual assessment of inter-rater agreement.

For analysis, we classified each paper based on their **research approaches** and **research contributions** (Step 4 in Fig. 1). We use the classification of research approaches by Wieringa et al. [98] and later updated by Petersen et al. [73] to describe the contribution of the papers and thus the maturity of the research area. Table 1 summarises the different paper categories.

To provide a detailed view on what parts of modelling the papers contribute to, we break down the papers into different *modelling concerns*. These concerns are based on a classification we developed in previous work [52]. We do not claim that these concerns are exhaustive, but they serve as a useful tool to further break down a paper's contribution in terms of modelling. The modelling concerns are depicted in a basic feature model notation in Fig. 2. Solid lines from the *modelling concerns* box with an empty circle at the end depict optional features. That is, all of the features on the right-hand side of the figure are potential concerns of modelling that can be discussed in a paper. However, none is mandatory, as all features can be omitted in a paper. The different concerns are as follows:

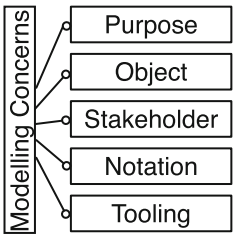


Fig. 2 Classification of Modelling Concerns. Adapted from [52]

1. Purpose: For what purpose is/are the model(s) created? For instance, models could serve as an input for code generation, or as an architecture description.
2. Object: What is being described in the model? For instance, the model could describe an entire system, a subsystem or component, or a process.
3. Stakeholder: Who is/are the primary stakeholders of the model? For example, a model could primarily serve as end-user documentation, or as a blueprint for developers.
4. Notation: What modelling notation(s) is/are used? For instance, UML might be used.
5. Tooling: Which tools are prescribed? For instance, Eclipse Papyrus might be prescribed due to a dependence on a custom plugin.

To understand the modelling concerns, we used open coding in the 35 papers.

3.4 Threats to validity

In this section, we describe the potential validity threats of our study and the steps we have taken to mitigate them. We follow the categorisation of threats into internal validity, construct validity, external validity, and reliability according to Runeson et al. [80].

3.4.1 Internal validity

Internal validity reflects to what extent causal relationships are closely examined and other, unknown factors might impact the findings.

As a main means to reduce threats to internal validity, we tried to increase reliability of our exclusion steps by calculating inter-rater agreements and repeating the steps until a satisfactory level of agreement was reached between the two authors. To avoid memory effects, we chose a new random sample to calculate inter-rater agreements after each round.

A threat of our review protocol is the strictness of the exclusion criteria. As we had to reduce a large number of initial papers to a manageable amount for detailed analysis,

Table 1 Classification of paper types based on Wieringa et al. [98] and Petersen et al. [73]

Research approach	Definition
Evaluation research	Investigation of a problem employing case study, controlled experiment, survey etc in the industrial context, implementing a technique in practice, and evaluating the implementation, i.e. showing the benefits and drawbacks of the implementation
Solution proposals	Proposed solution (either novel or extension of an existing technique) with a relevant argument or a good example to show the solution's benefits. No empirical evaluation
Validation research	Novel research techniques used for experiments which are not established in practice (for example, studies conducted with students); investigation of properties of a solution proposal that has not been implemented yet
Philosophical papers	A conceptual framework that sketches a new way of looking at existing things
Opinion papers	An author's personal opinion whether a certain technique is good or bad, or how things should be done
Experience papers	An author's personal experience of how something has been done

and furthermore had to deal with ambiguous terminology, we decided to exclude strongly in the first two exclusion rounds (based on title and venue, and based on abstract). This could lead to papers being excluded that do contain valuable guidance to model creation, but do not clearly state so in their abstracts. Given the trade-off between search coverage and detailed analysis, we decided to accept this threat. In particular, we observe that venues such as Requirements Engineering (RE) conference and the European Conference on Modelling Foundations and Applications (ECMFA) are not represented in the data set, despite strong focus on modelling. This might be due to a different focus, e.g. on theoretical aspects in ECMFA, but we do not know whether this could represent a bias.

3.4.2 Construct validity

Construct validity reflects to what extent the measures represent the construct investigated.

In this paper, we investigate model creation, guidance, and related topics. A threat to the validity of our study is that these topics are not described using established, standardised terminology in the modelling community. Therefore, it might in some cases be difficult to decide when a paper in fact provides guidance for model creation, and when not. To reduce this threat, we used open coding for data analysis, without relying on fixed keywords being used in the respective papers.

3.4.3 External validity

External validity is reflecting to what extent findings can be generalised beyond the concrete sample.

In case of our literature review, we use the major digital libraries for data collection, which should lead to a representative sample of publications in software engineering. A potential threat to validity in the review was our decision to treat the search string S2 separate from S1, due to the conflated keywords "approach," "method," "process" and "template." The relative amount of included papers is somewhat similar in S1 and S2, indicating that a lot of relevant papers might have been excluded by performing random sampling on S2. This is a threat we have to accept given the large amount of search results. However, we also note that the papers from S2 did not yield any results substantially different from the papers in S1.

3.4.4 Reliability

Reliability describes the degree to which similar results would be obtained if the same study would be repeated, by the same or by other researchers.

The different steps of the data collection and analysis are clearly described in the previous subsections, so that we are confident that other researchers could repeat the study. Nevertheless, there is subjectivity to several parts of our study, in particular the exclusion and analysis steps of the review. We tried to be conservative in our exclusion, and calculated inter-rater agreement at all exclusion steps. Furthermore, the analysis was performed by multiple researchers, and disagreements were discussed. To increase reliability, the data set with detailed tables is published[2]).

4 Results

In this section, we present the results of the SLR. First, we classify the final 35 papers in Sect. 4.1. Based on the little empirical evidence, we find in the modelling literature, we then introduce definitions for modelling guidelines, methods and styles in Sect. 4.2. Finally, we answer the RQ in Sect. 4.3.

[2] https://doi.org/10.5281/zenodo.7685694.

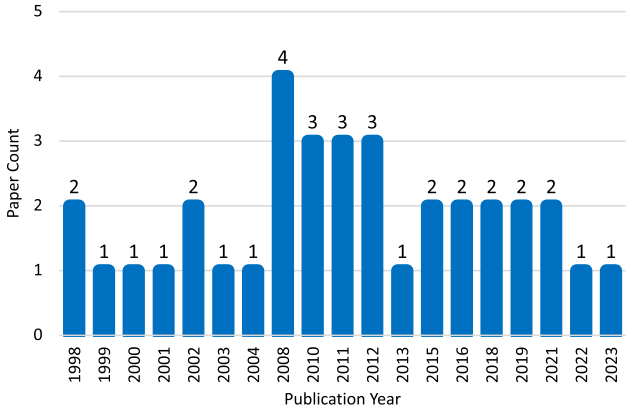


Fig. 3 Paper count per year. Years without papers are omitted

Table 2 Paper types in our analysis (based on Wieringa et al. [98])

Paper type	# papers
Evaluation research (EV)	8
Solution proposals (SP)	22
Validation research (VR)	1
Philosophical papers (PH)	1
Opinion papers (OP)	0
Experience papers (EX)	3

4.1 Paper classification

We ultimately analysed 22 papers through **S1**, with an additional 7 papers added through snowball search and 6 papers added through **S2**. The publication years of our final 35 papers range from 1998 to 2023, with the majority of the papers from 2008 to 2015. Figure 3 depicts the actual counts, where years without paper are omitted.

There is no clear pattern with respect to publication venues in the data. Venues targeted towards modelling, such as the ER and UML/Models conference series or the Software and Systems Modelling journal are present, as well as general software engineering venues, such as ICSE or Transactions on Software Engineering.

Table 2 summarises the types of the 35 papers.

In terms of modelling concerns, we find that the model purpose, the modelling notation, and the object that is modelled are typically described in the papers, or can be derived from the descriptions. In contrast, stakeholders and tooling are rarely reported.

The included papers state several *purposes* for the models created. Most commonly (15 out of 35 papers), there is no exact purpose stated for the constructed models, i.e. the purpose is to model parts of or a complete system. The guidance provided in the papers thus aims broadly at improving the quality of the resulting models, or make the process more systematic. The remaining papers state specific purposes for their constructed models. These purposes vary widely, e.g. models for simulation, formal verification, evaluation of any system property (e.g. security), requirements and system analysis, avoiding inconsistencies between models, improving communication between teams and generate (parts of) an application. Given our snowball search in the area of BPM, business processes are the dominating *object* modelled in the included papers. This is followed by six papers where modelling of embedded control systems is reported. One paper describes modelling to address many challenges during the design of Space Interferometry Mission (SIM). The remainder of the papers focuses on a mix of different models, such as use cases, architecture, and requirements.

Only two papers report a specific *stakeholder*. Additionally, several papers refer to general terms such as *engineers, programmers, users, domain experts, or software developers*. That is, it is often unclear who creates, modifies, reads, or otherwise comes in contact with the proposed models.

15 of our 35 papers discuss models in UML *notation*, or extensions thereof, such as UML profiles or the UML-RT extensions. This is followed by 6 papers using BPMN. Finally, several papers do not focus on a specific notation, or propose their own notations.

Tools are rarely reported in our data set. Only 7 out of 35 papers report a specific tool, 6 of which are specialised tools developed for the presented approaches and one paper uses FAME toolchain. Of the remaining 28 papers, most are tool agnostic, e.g. as they target general quality improvement to a specific type of model or as they do not rely on specific tool features.

4.2 Definitions of key terms in model creation

Starting from the initial keyword search and continuing throughout the entire analysis, we noticed that terms such as *method*, *guidelines*, and *style* are used seemingly arbitrary in the community. Therefore, we decided to provide definitions for a number of key terms, in order to allow us to better structure the literature, and to support future work in the area.

Modelling is a creative task, which requires thinking and problem solving, both of which can affect the quality of the model. In the general literature on problem solving, Isaksen et al. [42] discuss that, "...when individuals, in both school and corporate settings, understand their own style of problem solving, they are able to learn and apply process tools more effectively, and when teams appreciate the styles of their individual members, their problem solving efforts are enhanced". That is, styles in problem solving differ between individuals, which makes it necessary to distinguish from guidelines or methods, where specific steps or activities are prescribed on good practice.

We define the term *modelling style* in reference to cognitive styles, which are "an individual's preferred way of

gathering, processing and evaluating data" [2]. Messik [64] defines cognitive styles as "consistent individual differences in preferred ways of organising and processing information and experience." Accordingly, we define modelling styles as follows.

> **Modelling styles** are "consistent individual differences in preferred ways of creating and processing software models."

Here, creation can include aspects such as the choice of modelling language, preferred elements of those languages, of tools, and organising the created model (i.e. layouting, or using a specific subset of modelling elements from a given language). Processing includes reading, understanding and validating models. For example, styles could be "I like to draw all classes first, then connect them" or "I like to have the most general classes in the top."

In contrast, we define modelling methods as follows.

> **Modelling methods** are "sets of steps or constraints on how one or multiple models are created."

In contrast to a modelling style, a modelling method is identical across individuals, and does not depend on preferences. However, modelling methods and modelling styles overlap and may conflict with each other, e.g. if an individual's preferred ways of creating a model are in conflict with the prescribed method. For instance, the method proposed by Machado et al. [59] proposes a step-wise formalisation of behaviour, starting from textual rules. This could be in conflict with an engineer who works by incrementally adding formal behaviour models, entirely omitting (informal) textual input. Another example of a modelling method would be the steps of creating the diagram such as "first underline all nouns, then underline verbs that relate two or more nouns. Then use the resulting nouns as classes, and the verbs as relations." (adapted from the classic recommendations by Abbott [1]).

Finally, for guidelines we use an existing dictionary definition.

> **Guidelines** are "information intended to advise people on how something should be done or what something should be[3]".

Following this definition, modelling methods are always also guidelines, as they advise people how to create a model. The opposite is not true, as guidelines do not necessarily have to be in the form of steps or constraints. An example set of guidelines is provided by Reggio et al. [76], who recommend which parts of UML activity diagrams should be used depending on the purpose of the resulting model. A simplified example for guidelines could be instructions such as "do not include the user in a class diagram" or "only use classes that do not refer to implementation details."

Note that the definitions for methods and guidelines do not include the purpose they serve in detail. For instance, guidelines could be used to provide recommendations on how to use a notation given a chosen task, such as in [76]. Similarly, providing recommendations on naming, as in [103], could help creating models that are easily communicated and maintained in an organisation, i.e. a form of standardisation. While we do not include it in our definitions, we believe providing clear purpose statements increases the usefulness of modelling guidance.

4.3 Modelling creation guidance in literature

After defining the terms modelling *style*, *method*, and *guidelines* in Sect. 4.2, we now re-visit the literature in terms of our RQ, i.e. *What kind of guidance exists in SE literature on model creation?*.The included papers and their corresponding information are described in Tables 3, 4, 5, and 6.

Based on the definitions, we found 11 papers containing guidelines (i.e. [5, 16–18, 25, 26, 35, 63, 76, 85, 102]), 20 papers describing modelling methods (i.e. [11, 19, 24, 31, 32, 34, 37, 38, 43, 44, 58, 59, 62, 65, 66, 78, 88, 95, 100, 103]), and 1 paper investigating modelling styles (i.e. [75]). Out of the remaining 3 papers, 1 (i.e.[28]) includes both method and a guideline. The rest (i.e. [14, 86]) discuss selected modelling concerns without providing any of the above. In the following, we describe different concerns addressed by the included papers, and relate these to whether guidelines, methods, or styles are described.

4.3.1 Inconsistent terminology

As discussed in Sect. 4.2, terminology is used inconsistently across the literature. For instance, [75] describes modelling style in a similar way as we defined it above, while the "styles" described in [76] are different levels of formality used in business process models. That is, they do not refer to individual differences between modellers. Similarly, the described "method" in [76] is not a set of steps or constraints and therefore rather falls under our definition of guidelines. We also find "guidelines" described in [31] that fit our description of a method. Finally, we see a mixed approach consisting of a step-wise "prescribed process" with a list of "guidelines" to apply those steps in [28].

[3] https://dictionary.cambridge.org/dictionary/english/guideline?q=Guidelines.

Table 3 Paper overview, Part 1

Paper	[43]	[59]	[38]	[24]	[19]	[25]	[76]	[35]	[31]
Guidance type	M	M	M	M	M	G	G	G	M
Paper type	SP	SP	SP	SP	SP	EX	SP	SP	SP
What versus how	Both	What	Both	What	What	What	What	What	Both
Views/perspectives	No	No	Yes	Yes	Yes	No	No	No	No
Decomposition/refinement	Yes	Yes	No	No	Yes	Yes	No	Yes	No
Practices/anti-patterns	No	No	Yes	No	No	No	Yes	No	No
Cognition	No	No	No	No	No	No	No	No	No

Guidance type consists of Methods (M), Guidelines (G), and Styles (S). Paper types are listed according to Table 2

Table 4 Paper overview, Part 2

Paper	[85]	[62]	[18]	[102]	[5]	[95]	[100]	[26]	[44]
Guidance type	G	M	G	G	G	M	M	G	M
Paper type	SP	SP	EV	EV	SP	SP	EV	EX	SP
What versus how	Both	What	How	How	What	What	Both	Both	What
Views/perspectives	No	No	No	No	No	Yes	No	No	No
Decomposition/refinement	No	Yes	No	No	No	Yes	No	No	No
Practices/anti-patterns	Yes	No	Yes	Yes	Yes	No	No	Yes	No
Cognition	No	No	No	No	No	No	No	No	No

Guidance type consists of Methods (M), Guidelines (G), and Styles (S). Paper types are listed according to Table 2

Table 5 Paper overview, Part 3

Paper	[65]	[16]	[75]	[14]	[86]		[63]	[17]	[32]
Guidance type	M	G	S	N/A	N/A	M	G	G	M
Paper type	EV	SP	EV	PH	VR	EV	EV	SP	SP
What versus how	How	How	N/A	N/A	N/A	How	How	Both	What
Views/perspectives	No	Yes	N/A	N/A	N/A	No	No	No	No
Decomposition/refinement	Yes	Yes	N/A	N/A	N/A	Yes	No	No	No
Practices/anti-patterns	No	No	N/A	N/A	N/A	No	Yes	Yes	No
Cognition	No	No	Yes	Yes	Yes	No	No	No	No

Guidance type consists of Methods (M), Guidelines (G), and Styles (S). Paper types are listed according to Table 2

Table 6 Paper overview, Part 4

Paper	[37]	[28]	[103]	[58]	[88]	[11]	[34]	[66]
Guidance type	M	M	M	M	M	M	M	M
Paper type	SP	SP	SP	SP	EV	SP	SP	EX
What versus how	Both	How	Both	What	What	Both	What	Both
Views/perspectives	No	No	No	No	Yes	No	No	No
Decomposition/refinement	Yes	Yes	Yes	No	Yes	Yes	No	Yes
Practices/anti-patterns	No	Yes	Yes	No	No	No	No	Yes
Cognition	No	No	No	No	No	No	No	No

Guidance type consists of Methods (M), Guidelines (G), and Styles (S). Paper types are listed according to Table 2

4.3.2 What instead of how

There is one major distinction that can be made between our included papers. While some papers give concrete advice on how a single model (or models of the same type) should be created, 25 papers provide guidance on using different models/diagrams towards an aim, such as modelling an entire system or specific property of a system. That is, the latter type of papers typically focuses more on *what* to use, e.g. in terms of diagram types. As timing and order are important elements in these descriptions, several of these papers are method papers (18 out of 25).

For example, in [43], the authors describe a method to systematically model service-oriented systems in a series of steps. The method aims at avoiding inconsistencies between models and has a natural order of steps. That is, constraints are imposed at language level before the actual system models are created. In [24], a six-step method for modelling software architecture is presented, composed of selecting an architectural style/pattern, defining architectural structural elements and rules, specifying data structures, specifying structural units, specifying mechanisms to support state models and timers, and eventually build the architecture model. Again, different models are used at different steps of the method, and the order of the steps is important. [35] describe guidelines for the use of OntoUML. These guidelines are essentially restrictions on what parts of the OntoUML notation to use. Neither of these three papers provides detailed guidance on *how* the individual models are created.

Finally, twelve papers mix guidance on what notations, diagrams, or formalisms to use with detailed instructions on how to use these. For example, [85] aims to improve the quality of information models by providing guidelines in the form of best practices. These best practices include instructions on *what* diagrams to create, and *how* to ensure several of them are consistent. [26] also provides guidelines, including *what* to use (e.g. in terms of tooling), and *how* to approach modelling of use cases.

4.3.3 View-based or perspective-based guidance

Another common way to provide modelling guidance, found in 6 out of 35 papers, is to propose modelling according to multiple different views or system concerns, similar to architectural documentation that is typically presented according to multiple dimensions or in multiple views [4].

For example, guidelines for increasing the quality of process models are presented in [5], using both perspectives and views. The six perspectives of correctness, relevance, economic efficiency, clarity, comparability, and systematic design are considered, as well as several modelling views, such as a data view, organisational view, and control view. Similarly, [38] proposes a method to semi-automatically create hypermedia applications by modelling the application using different views, such as the navigation and presentation views.

4.3.4 Decomposition-driven or refinement-driven guidance

A common approach to deal with complexity in computer science is to decompose a problem into sub-problems, or to incrementally refine a solution. These two approaches are also reflected in guidance on modelling. That is, several papers propose methods and guidelines in which an abstract, incomplete or informal model is incrementally refined.

For example, the method proposed in [59] suggests to incrementally formalise behaviour, starting from textual rules, which are then refined into graphical models, and finally formalised using colored Petri nets. Similarly, Marincic et al. [62] suggest a step-wise process in which requirements for embedded control systems are refined. A method to model cyber-physical systems in terms of abstraction levels is presented in [19], where abstract concepts such as initial requirements are modelled first, and then increasingly described in more detail. Ultimately, the method allows for simulation of the CPS' behaviour. A method that prescribes step-wise decomposition of business processes is described in [75]. Between each decomposition step, it is decided which parts should be expressed together in a process model. [65] describes decomposition steps, each of which is accompanied by guiding questions. Finally, in [11], a method to model agent-based systems is described, which relies on breaking down the overall process into modelling the agent space, the agent components, and the agents.

4.3.5 Best practices and anti-patterns

Best practices and anti-patterns are common ways to describe the state of practice and guide practitioners, e.g. in core computer science areas such as programming [56, 94] and software design [8]. This is also reflected in our data set, with 12 papers suggesting such best practices or anti-patterns.

For example, in [76], five modelling styles are proposed ranging from the so-called ultra light style, where UML activity diagrams are supported with free text, to the so-called precise operational style, where UML and OCL are used according to their semantics. While the authors call these *styles*, they are essentially best practices on which parts of the official UML notation should be used depending on the functional context (who, when, where, how and why will the model be used). The styles are based on the authors' understanding, without any empirical data supporting them. In another direction, Frank [28] provides general design guidelines for multi-level models. These include principles and rationales, e.g. specifying knowledge on the highest possible level to avoid redundancy.

4.3.6 Considering cognition

Finally, three papers in our dataset explicitly consider the modeller's cognitive processes, or argue that it should be taken into consideration.

In [86], the authors suggest that a mental model is created when a problem is conceptualised. The mental model is then mapped to the actual solution model. Assuming this process, the authors argue that improving the mental model could lead to an improved domain understanding. [14] argues that cognition needs to be taken into account to explain shortcomings in BPM. Finally, cognition is taken into account explicitly as a part of the theoretical framework in [18], where three styles for BPM are extracted based on an observational study. The results describe a "high-efficiency" style, a "good layout but less efficiency" style, and a "neither good layout nor very efficient" style. The authors describe task-specific and modeller-specific factors behind each style, especially how the above-mentioned two factors influence specific aspects of each style.

5 Discussion

The objective of our SLR is to investigate how much guidance exists in the literature for model creation in SE. We discuss these findings with respect to a number of observed themes below.

5.1 Little empirical evaluation

Among 35 included papers, 22 are solution proposals (see Table 2), and only 1 contains validation research. These statistics demonstrate a lack of empirical evaluation present in the literature. In contrast, the papers obtained through snowball sampling in BPM show a higher amount of evaluation and validation, with 4 out of 7 papers being evaluation research, and one paper being validation research. That is, the area of BPM seems to be more mature when it comes to supporting model creation through empirical studies. Similarly, the only paper explicitly considering cognition and the construction of mental models were in the area of BPM.

This lack of empirical studies is comparably common in the area of software modelling. Zhang et al. [101] find that empirical research methods within SE are mostly applied to software maintenance, quality and testing. While software models and methods gained empirical attention[3] but none of them investigating modelling style nor model creation.

5.2 Lack of categorisation and organisation

We used two search strings, one with direct keywords (**S1**) like "guidelines" and "creation"; other with generic terms (**S2**), such as "approach" and "method." From **S1**, we find a variety of papers that, according to our definitions, propose guidelines and methods. However, they use terms such as **guidelines**, **methods**, **frameworks** and others in a seemingly arbitrary fashion. We address this issue by proposing definitions that make use of existing ones as much as possible. Additionally, we outlined dimensions and properties of existing modelling guidelines and methods. Interestingly, the 6 papers collected through **S2** all propose methods according to our definition. This indicates that the terms used are not entirely arbitrary, as guidelines were only found in S1. Potentially, "methods" might subjectively be considered as something larger among researchers, and therefore commonly include stepwise instructions. Finally, we did not find any single paper in **S1** or **S2** that proposes a modelling style according to our definition.

Common ways to provide guidance in modelling are to describe what diagrams should be created, to prescribe views or perspectives that should be considered, to encourage a stepwise decomposition or refinement, and to suggest general best practices or anti-patterns. With the exception of stepwise decomposition/refinement, these forms of guidance can either follow a dedicated set of steps or not. For instance, modelling guidelines could simply outline a few UML diagrams that need to be created in arbitrary order, while a modelling method could prescribe the same diagrams, but in a stepwise fashion.

5.3 Limited consideration of cognition

In BPM, substantial work exists that explicitly discusses the role of cognitive processes and individual preferences in creating business process models. In the SE modelling community, we do not find such work outside of the BPM community. On the one hand, we believe this relates to the relatively specialised nature of BPM as opposed to system modelling on a general level. That is, notations, level of abstraction, and purpose vary considerably in software and systems modelling and are likely to affect the cognitive processes and modelling styles of individuals. Nevertheless, we consider it relevant to explore this direction more in the future. Existing work from the BPM community can be used as valuable guidance to design similar studies in software modelling as a whole. This work necessarily needs to include expertise from other fields, such as Psychology.

[3] 66 papers published from 2013–2017 at EMSE and ESEM.

6 Conclusion

We conducted an SLR on guidance in software modelling. Using two search strings (S1 and S2) and snowballing, we extracted 35 papers out of 80051 for full-text reading.

We find that terminologies are used arbitrary and inconsistently, and that little empirically validated modelling guidance exists. Existing papers use different dimensions to provide guidance for modellers, often in the form of prescribing which diagrams/notations to use, which views/perspectives to model, how to decompose or refine a model of a system, or by providing best practices or anti-patterns. In the BPM community, cognition has been explicitly considered as an important aspect of the modelling process. However, our results show that we lack similar work outside the BPM community.

Our results help researchers by highlighting gaps in research, e.g. on cognition in software modelling. The link to research in BPM clearly highlights differences in this direction, that could be picked up by modelling researchers in future work. Thus, we believe future work should explicitly consider cognition when creating models, and design guidance and tooling accordingly. This work should be multi-disciplinary, including researchers from, e.g. Psychology.Furthermore, to address the lack of consistent terminology, we provide definitions for modelling *guidelines*, *methods*, and *styles*. These definitions help to clearly articulate modelling guidance in future work. We believe this is an important step in order to ensure a greater clarity of research results in the community. One clear result from our study is that empirical validation is lacking. Hence, it is important to address this in future work and indeed validate proposed modelling guidance.

For modelling practitioners, our results show the possible forms modelling guidance can take. Thus, it serves as an inspiration on how engineers can be supported when creating and maintaining models for different purposes, e.g. for formal analyses, increased understandability of models, or communication among stakeholders.

A Paper categorisation details

In the following, we list the papers we used during our analysis. Table 7 shows the papers extracted during the original

Table 7 Selected papers from search_S1

Paper	Titles	Authors
[43]	Context-based modelling: Introducing a novel modelling approach	M. Juhrisch and G. Dietz
[59]	Scenario-based modelling in industrial information systems	R. J. Machado, J.o M. Fernandes, J. P. Barros and L. Gomes
[38]	A UML-Based Methodology for Hypermedia Design	R. Hennicker and N. Koch
[24]	Architecture modelling for translative model-driven development	A. Fatwanto and C. Boughton
	Modelling and simulation of CPS based on SysML and modellica(KG)	F. Deng, Y. Yan, F. Gao and Linbo Wu
	Modelling Industrial Embedded Systems with UML	J. M. Fernande, R. J. Machado and H. D. Santos
[76]	Business Process Modelling: Five Styles and a Method to Choose the Most Suitable One	G. Reggio, M. Leotta, F. Ricca and E. Astesiano
[35]	Design patterns and inductive modelling rules to support the construction of ontologically well-founded conceptual models in OntoUML	G. Guizzardi, A. P. das Graças, and R. S. S. Guizzardi
[31]	Guidelines for modelling reactive systems with coloured Petri nets	M. P. Gonçalves and J. M. Fernandes
[85]	The guidelines of modelling - An approach to enhance the quality in information models	R. Schuette and T. Rotthowe
[62]	Non-Monotonic Modelling from Initial Requirements: A Proposal and Comparison with Monotonic Modelling Methods	J. Marincic, A. Mader, H. Wupper and R. Wieringa
[18]	Model development guidelines for UML-RT: conventions, patterns and antipatterns	T. K. Das and J. Dingel
[102]	Design guidelines for feature model construction: Exploring the relationship between feature model structure and structural complexity	X. Zhao and J. Gray

Table 7 continued

Paper	Titles	Authors
[5]	Guidelines of business process modelling	J. Becker, M. Rosemann and C. van Uthmann
[95]	A modelling approach for use-cases model in UML	Z. Wang
[100]	A modelling methodology to facilitate safety-oriented architecture design of industrial avionics software	J. Wu, T. Yue, S. Ali and H. Zhang
[26]	Use case modelling guidelines	D. G. Firesmith
[44]	WWM: a practical methodology for Web application modelling	C. Kaewkasi and W. Rivepiboon
[17]	A BPMN-driven framework for Multi-Robot System development	F. Corradini, S. Pettinari, B. Re, L. Rossi, and F. Tiezzi
[32]	MDD4CPD: Model-Driven Development Approach Proposal for Cyber-Physical Devices	R. F. Goncalves, A. Menolli and G. M. Dionisio
[37]	Communication Oriented Modelling of Evolving Systems of Systems	S. K. R. Harbo, M. K. Kristensen, E. P. Voldby, S. V. Andersen, F. C. Petersen and M. Albano
[28]	Prolegomena of a Multi-Level Modelling Method Illustrated with the FMML x	U. Frank

Table 8 Selected papers from snowballing set

Paper	Titles	Authors
[86]	Towards Understanding the Process of Process Modelling: Theoretical and Empirical Considerations	P. Soffer, M. Kaner and Y. Wand
	Guiding goal modelling with scenarios	C. Rolland, C. Souveyet and C. Ben Achour
[65]	Modelling Families of Business Process Variants: A Decomposition Driven Method	F. Milani, M. D., N. Ahmed and R. Matulevičius
[16]	Guidelines of a Unified Approach for Product and Business Process Modelling in Complex Enterprise	A. Corallo, P. De Paolis, M. Ippoliti, M. Lazoi, M. Scalvenzi and G. Secundo
[75]	Modelling Styles in Business Process Modelling	J. Pinggera, P. Soffer, S. Zugal, B. Weber, M. Weidlich, D. Fahland, H. A. Reijers and J. Mendling
[14]	The Structured Process Modelling Theory (SPMT) a cognitive view on why and how modellers benefit from structuring the process of process modelling	J. Claes, I. Vanderfeesten, F. Gailly, P. Grefen and G. Poels
[63]	Seven Process Modelling Guidelines (7PMG)	J. Mendling, H. A. Reijers and W. M. P. van der Aalst

Table 9 Selected papers from search_S2

Paper	Titles	Authors
[103]	Multidisciplinary interface model for design of mechatronic systems	C. Zheng, J. Le Duigou, M. Bricogne and B. Eynard
[58]	Model-based security engineering for secure systems development	A. Lunkeit and H. Pohl
[88]	User interface derivation from business processes: a model-driven approach for organisational engineering	K. Sousa, H. Mendonça, J. Vanderdonckt, E. Rogier and J. Vandermeulen
[11]	Supporting agent-based distributed software development through modelling and simulation	L. Cai, CK Chang and J. Cleland-Huang
[34]	Towards a Human-Centered UML for Risk Analysis: Application to a medical robot	J. Guiochet, G. Motet, C. Baron and G. Boy
[66]	Integrating System and Software Engineering Through Modelling	J. Mindock and G. Watney

search, while Table 8 shows the papers from the snowball search.

References

1. Abbott, R.J.: Program design by informal English descriptions. Commun. ACM **26**(11), 882–894 (1983). https://doi.org/10.1145/182.358441
2. Allinson, C., Hayes, J.: The cognitive style index: a measure of intuition analysis for organizational research. J. Manag. Stud. **33**, 119–135 (1996). https://doi.org/10.1111/j.1467-6486.1996.tb00801.x
3. 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). https://doi.org/10.1109/TSE.2015.2398877
4. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice. Addison-Wesley Professional (2003)
5. Becker, J., Rosemann, M., Von Uthmann, C.: Guidelines of business process modeling. In: Business Process Management, pp. 30–49. Springer (2000)
6. Bézivin, J., Muller, P. (eds.): The Unified Modeling Language, UML'98: Beyond the Notation, First International Workshop, Mulhouse, France, June 3-4, 1998, Selected Papers, Lecture Notes in Computer Science, vol. 1618. Springer (1999). https://doi.org/10.1007/b72309
7. Bordeleau, F.: A Systematic and Traceable Progression from Scenario Models to Communicating Hierarchical State Machines. Ph.D. thesis. Carleton University (2000)
8. Brown, W.H., Malveau, R.C., McCormick, H.W.S., Mowbray, T.J.: AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis. John Wiley & Sons, Inc. (1998)
9. Bruegge, B., Dutoit, A.H.: Object-Oriented Software Engineering Using UML, Patterns, and Java, 3rd edn. Prentice Hall Press (2009)
10. Burgueño, L., Cabot, J., Wimmer, M., Zschaler, S.: Guest editorial to the theme section on ai-enhanced model-driven engineering. Softw. Syst. Model. **21**(3), 963–965 (2022)
11. Cai, L., Chang, C.K., Cleland-Huang, J.: Supporting agent-based distributed software development through modeling and simulation. In: The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems, 2003. FTDCS 2003. Proceedings, pp. 56–62. IEEE (2003)
12. Cámara, J., Troya, J., Burgueño, L., Vallecillo, A.: On the assessment of generative ai in modeling tasks: an experience report with chatgpt and uml. Softw. Syst. Model. 1–13 (2023)
13. Chaaben, M.B., Burgueño, L., Sahraoui, H.: Towards using few-shot prompt learning for automating model completion. arXiv preprint arXiv:2212.03404 (2022)
14. Claes, J., Vanderfeesten, I., Gailly, F., Grefen, P., Poels, G.: The structured process modeling theory (spmt) a cognitive view on why and how modelers benefit from structuring the process of process modeling. Inf. Syst. Front. **17**, 1401–1425 (2015). https://doi.org/10.1007/s10796-015-9585-y
15. Claes, J., Vanderfeesten, I., Pinggera, J., Reijers, H.A., Weber, B., Poels, G.: A visual analysis of the process of process modeling. Inf. Syst. e-Bus. Manag. **13**(1), 147–190 (2015)
16. Corallo, A., Paolis, P., Ippoliti, M., Lazoi, M., Scalvenzi, M., Secundo, G.: Guidelines of a unified approach for product and business process modeling in complex enterprise. Knowl. Process Manag. (2011). https://doi.org/10.1002/kpm.381
17. Corradini, F., Pettinari, S., Re, B., Rossi, L., Tiezzi, F.: A bpmn-driven framework for multi-robot system development. Robot. Auton. Syst. **160**, 104,322 (2023)
18. Das, T., Dingel, J.: Model development guidelines for uml-rt: conventions, patterns and antipatterns. Softw. Syst. Model. (2018). https://doi.org/10.1007/s10270-016-0549-6
19. Deng, F., Yan, Y., Gao, F., Wu, L.: Modeling and simulation of cps based on sysml and modelica (kg). In: Proceedings of the 31st International Conference on Software Engineering & Knowledge Engineering SEKE 2019 (2019)
20. Douglass, B.P.: Doing hard time: developing real-time systems with UML, objects, frameworks, and patterns, vol. 1. Addison-Wesley Professional (1999)
21. Dwyer, M.B., Avrunin, G.S., Corbett, J.C.: Property specification patterns for finite-state verification. In: Proceedings of the Second Workshop on Formal Methods in Software Practice, pp. 7–15. ACM (1998)
22. 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. IEEE (1999)
23. Fang, J., Zhu, Z., Li, S., Su, H., Yu, Y., Zhou, J., You, Y.: Parallel training of pre-trained models via chunk-based dynamic memory management. IEEE Trans. Parallel Distrib. Syst. **34**(1), 304–315 (2022)
24. Fatwanto, A., Boughton, C.: Architecture modeling for translative model-driven development. In: 2008 International Symposium on Information Technology, vol. 1, pp. 1–9 (2008). https://doi.org/10.1109/ITSIM.2008.4631619
25. Fernandes, J., Machado, R., Santos, H.: Modeling industrial embedded systems with uml. In: Proceedings of the Eighth International Workshop on Hardware/Software Codesign. CODES 2000, pp. 18–22 (2000). https://doi.org/10.1109/HSC.2000.843700
26. Firesmith, D.: Use case modeling guidelines. In: Proceedings of Technology of Object-Oriented Languages and Systems - TOOLS 30 (Cat. No.PR00278), pp. 184–193 (1999). https://doi.org/10.1109/TOOLS.1999.787548
27. Fleiss, J.L., Levin, B., Paik, M.C.: Statistical Methods for Rates and Proportions. Wiley (2013)
28. Frank, U.: Prolegomena of a multi-level modeling method illustrated with the fmml x. In: 2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), pp. 521–530. IEEE (2021)
29. Gamma, E.: Design patterns: elements of reusable object-oriented software. Pearson Education India (1995)
30. Giraldo, F.D., España, S., Pastor, O.: Analysing the concept of quality in model-driven engineering literature: a systematic review. In: 2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS), pp. 1–12. IEEE (2014)
31. Gonçalves, M., Fernandes, J.M.: Guidelines for modelling reactive systems with coloured petri nets. In: Machado, R.J., Maciel, R.S.P., Rubin, J., Botterweck, G. (eds.) Model-Based Methodologies for Pervasive and Embedded Software, pp. 126–137. Springer, Berlin Heidelberg, Berlin, Heidelberg (2013)
32. Goncalves, R.F., Menolli, A., Dionisio, G.M.: Mdd4cpd: model driven development approach proposal for cyber-physical devices. In: Anais do XVIII Simpósio Brasileiro de Sistemas de Informação. SBC (2022)
33. Grunske, L.: Specification patterns for probabilistic quality properties. In: 2008 ACM/IEEE 30th International Conference on Software Engineering, pp. 31–40. IEEE (2008)
34. Guiochet, J., Motet, G., Baron, C., Boy, G.: Toward a human-centered uml for risk analysis: Application to a medical robot. In: Human Error, Safety and Systems Development: IFIP 18th World

Computer Congress TC13/WC13. 5 7th Working Conference on Human Error, Safety and Systems Development 22–27 August 2004 Toulouse, France, pp. 177–191. Springer (2004)
35. Guizzardi, G., das Graças, A.P., Guizzardi, R.S.S.: Design patterns and inductive modeling rules to support the construction of ontologically well-founded conceptual models in ontouml. In: C. Salinesi, O. Pastor (eds.) Advanced Information Systems Engineering Workshops, pp. 402–413. Springer, Berlin, Heidelberg (2011)
36. Gwet, K.L.: Handbook of Inter-rater Reliability Advanced Analytics. LLC, Gaithersburg, MD (2010)
37. Harbo, S.K.R., Kristensen, M.K., Voldby, E.P., Andersen, S.V., Petersen, F.C., Albano, M.: Communication oriented modeling of evolving systems of systems. In: 2021 16th International Conference of System of Systems Engineering (SoSE), pp. 88–94. IEEE (2021)
38. Hennicker, R., Koch, N.: A uml-based methodology for hypermedia design. In: Evans, A., Kent, S., Selic, B. (eds.) ≪ UML ≫ 2000 – The Unified Modeling Language, pp. 410–424. Springer, Berlin Heidelberg, Berlin, Heidelberg (2000)
39. 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)
40. 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)
41. 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)
42. Isaksen, S., Kaufmann, A., Bakken, B.T.: An examination of the personality constructs underlying dimensions of creative problem-solving style. J. Creat. Behavi. **50**, 268–281 (2016)
43. Juhrisch, M., Dietz, G.: Context-based modeling: introducing a novel modeling approach. In: Esswein, W., Turowski, K., Juhrisch, M. (eds.) Modellierung betrieblicher Informationssysteme (MobIS 2010). Modellgestütztes Management, pp. 111–130. Gesellschaft für Informatik e.V., Bonn (2010)
44. Kaewkasi, C., Rivepiboon, W.: Wwm: a practical methodology for web application modeling. In: Proceedings 26th Annual International Computer Software and Applications, pp. 603–608 (2002). https://doi.org/10.1109/CMPSAC.2002.1045070
45. Kharchenko, V., Fesenko, H., Illiashenko, O.: Quality models for artificial intelligence systems: characteristic-based approach, development and application. Sensors **22**(13), 4865 (2022)
46. Kılıç, S.: Kappa testi. J. Mood Disord. **5**(3) (2015)
47. Kitchenham, B., Brereton, P., Budgen, D., Turner, M., Bailey, J., Linkman, S.: Systematic literature reviews in software engineering-a systematic literature review. Inf. Softw. Technol. **51**, 7–15 (2009). https://doi.org/10.1016/j.infsof.2008.09.009
48. Kitchenham, B.A., Charters, S.: Guidelines for performing systematic literature reviews in software engineering. Tech. Rep. EBSE 2007-001, Keele University and Durham University Joint Report (2007)
49. Kuhrmann, M., Méndez Fernández, D., Daneva, M.: On the pragmatic design of literature studies in software engineering: an experience-based guideline. Empir. Softw. Eng. **22**, 2852–2891 (2017). https://doi.org/10.1007/s10664-016-9492-y
50. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. Biometrics 159–174 (1977)
51. Langford, M.A., Chan, K.H., Fleck, J.E., McKinley, P.K., Cheng, B.H.: Modalas: model-driven assurance for learning-enabled autonomous systems. In: 2021 ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems (MODELS), pp. 182–193. IEEE (2021)
52. Liebel, G.: Model-Based Requirements Engineering in the Automotive Industry: Challenges and Opportunities. Chalmers Tekniska Högskola (Sweden) (2016)
53. 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 (2018). https://doi.org/10.1007/s10270-016-0523-3
54. Liebel, G., Tichy, M., Knauss, E.: Use, potential, and showstoppers of models in automotive requirements engineering. Softw. Syst. Model. (2018). https://doi.org/10.1007/s10270-018-0683-4
55. Lohmeyer, Q., Meboldt, M., et al.: How we understand engineering drawings: an eye tracking study investigating skimming and scrutinizing sequences. In: International conference on engineering design ICED, vol. 15 (2015)
56. Long, F., Mohindra, D., Seacord, R.C., Sutherland, D.F., Svoboda, D.: Java Coding Guidelines: 75 Recommendations for Reliable and Secure Programs. Addison-Wesley (2013)
57. Loniewski, G., Insfran, E., Abrahão, S.: A systematic review of the use of requirements engineering techniques in model-driven development. In: International Conference on Model Driven Engineering Languages and Systems, pp. 213–227. Springer (2010)
58. Lunkeit, A., Pohl, H.: Model-based security engineering for secure systems development. In: ARCS Workshop 2018; 31th International Conference on Architecture of Computing Systems, pp. 1–10. VDE (2018)
59. Machado, R.J., Fernandes, J.M., Barros, J.P., Gomes, L.: Scenario-based modeling in industrial information systems. In: Hinchey, M., Kleinjohann, B., Kleinjohann, L., Lindsay, P.A., Rammig, F.J., Timmis, J., Wolf, M. (eds.) Distributed, Parallel and Biologically Inspired Systems, pp. 19–30. Springer, Berlin Heidelberg, Berlin, Heidelberg (2010)
60. Maier, A., Baltsen, N., Christoffersen, H., Störrle, H.: Towards diagram understanding: a pilot study measuring cognitive workload through eye-tracking. In: Proceedings of International Conference on Human Behaviour in Design 2014 (2014)
61. Maoz, S., Ringert, J.O.: Gr(1) synthesis for ltl specification patterns. In: Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2015, pp. 96–106. Association for Computing Machinery (2015). https://doi.org/10.1145/2786805.2786824
62. Marincic, J., Mader, A., Wupper, H., Wupper, H., Wieringa, R.: Non-monotonic modelling from initial requirements: a proposal and comparison with monotonic modelling methods. In: Proceedings of the 3rd International Workshop on Applications and Advances of Problem Frames, pp. 67–73 (2008). https://doi.org/10.1145/1370811.1370825
63. Mendling, J., Reijers, H., Aalst, W.: Seven process modeling guidelines (7pmg). Inf. Softw. Technol. **52**, 127–136 (2010). https://doi.org/10.1016/j.infsof.2009.08.004
64. Messick, S.: The nature of cognitive styles: problems and promise in educational practice. Educ. Psychol. **19**, 59–74 (1984)
65. Milani, F., Dumas, M., Ahmed, N., Matulevičius, R.: Modelling families of business process variants: a decomposition driven method. Inf. Syst. (2013). https://doi.org/10.1016/j.is.2015.09.003
66. Mindock, J., Watney, G.: Integrating system and software engineering through modeling. In: 2008 IEEE Aerospace Conference, pp. 1–12. IEEE (2008)
67. 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. Springer, Berlin Heidelberg (2008)
68. Mohagheghi, P., Gilani, W., Stefanescu, A., Fernandez, M.A., Nordmoen, B., Fritzsche, M.: Where does model-driven engineer-

ing help? experiences from three industrial cases. Softw. Syst. Model. **12**(3), 619–639 (2013)
69. Moody, D.: The "physics" of notations: Toward a scientific basis for constructing visual notations in software engineering. IEEE Trans. Softw. Eng. **35**(6), 756–779 (2009). https://doi.org/10.1109/TSE.2009.67
70. Nguyen, P.H., Klein, J., Le Traon, Y., Kramer, M.E.: A systematic review of model-driven security. In: 2013 20th Asia-Pacific Software Engineering Conference (APSEC), vol. 1, pp. 432–441. IEEE (2013)
71. Nguyen, P.H., Kramer, M., Klein, J., Le Traon, Y.: An extensive systematic review on the model-driven development of secure systems. Inf. Softw. Technol. **68**, 62–81 (2015)
72. Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M.: Systematic mapping studies in software engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering, p. 17 (2008)
73. Petersen, K., Vakkalanka, S., Kuzniarz, L.: Guidelines for conducting systematic mapping studies in software engineering: an update. Inf. Softw. Technol. **64**, 1–18 (2015). https://doi.org/10.1016/j.infsof.2015.03.007
74. Pinggera, J., Soffer, P., Fahland, D., Weidlich, M., Zugal, S., Weber, B., Reijers, H., Mendling, J.: Styles in business process modeling: an exploration and a model. Softw. Syst. Model. (2013). https://doi.org/10.1007/s10270-013-0349-1
75. Pinggera, J., Soffer, P., Zugal, S., Weber, B., Weidlich, M., Fahland, D., Reijers, H.A., Mendling, J.: Modeling styles in business process modeling. In: Bider, I., Halpin, T., Krogstie, J., Nurcan, S., Proper, E., Schmidt, R., Soffer, P., Wrycza, S. (eds.) Enterprise, Business-Process and Information Systems Modeling, pp. 151–166. Springer, Berlin Heidelberg, Berlin, Heidelberg (2012)
76. Reggio, G., Leotta, M., Ricca, F., Astesiano, E.: Business process modelling: five styles and a method to choose the most suitable one. In: Proceedings of the Second Edition of the International Workshop on Experiences and Empirical Studies in Software Modelling, EESSMod '12. Association for Computing Machinery, New York, NY, USA (2012). https://doi.org/10.1145/2424563.2424574
77. Rivera, L.F., Müller, H.A., Villegas, N.M., Tamura, G., Jiménez, M.: On the engineering of iot-intensive digital twin software systems. In: Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops, pp. 631–638 (2020)
78. Rolland, C., Souveyet, C., Achour, C.: Guiding goal modeling using scenarios. IEEE Trans. Softw. Eng. TSE (1999). https://doi.org/10.1109/32.738339
79. Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W.E., et al.: Object-Oriented Modeling and Design, vol. 199. Prentice-hall Englewood Cliffs, NJ (1991)
80. Runeson, P., Höst, M., Rainer, A., Regnell, B.: Case study research in software engineering—guidelines and examples (2012)
81. Saini, R., Mussbacher, G., Guo, J.L., Kienzle, J.: Automated, interactive, and traceable domain modelling empowered by artificial intelligence. Softw. Syst. Model. 1–31 (2022)
82. Santos, M., Gralha, C., Goulão, M., Araújo, J.: Increasing the semantic transparency of the kaos goal model concrete syntax. In: Trujillo, J.C., Davis, K.C., Du, X., Li, Z., Ling, T.W., Li, G., Lee, M.L. (eds.) Conceptual Modeling, pp. 424–439 (2018)
83. Santos, M., Gralha, C., Goulão, M., Araújo, J., Moreira, A.: On the impact of semantic transparency on understanding and reviewing social goal models. In: 2018 IEEE 26th International Requirements Engineering Conference (RE), pp. 228–239 (2018). https://doi.org/10.1109/RE.2018.00031
84. Schätz, B., Törngreen, M., Bensalem, S., Cengarle, M.V., Pfeifer, H., McDermid, J., Passerone, R., Sangiovanni-Vincentelli, A.L.: Cyber-physical european roadmap and strategy: research agenda and recommendations for action. CyPhERS . Tech. Rep (2015)
85. Schuette, R., Rotthowe, T.: The guidelines of modeling – an approach to enhance the quality in information models. In: Ling, T.W., Ram, S., Li Lee, M. (eds.) Conceptual Modeling – ER '98, pp. 240–254. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)
86. Soffer, P., Kaner, M., Wand, Y.: Towards understanding the process of process modeling: theoretical and empirical considerations. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) Business Process Management Workshops, pp. 357–369. Springer, Berlin, Heidelberg (2012)
87. Somogyi, F.A., Asztalos, M.: Systematic review of matching techniques used in model-driven methodologies. Softw. Syst. Model. **19**(3), 693–720 (2020)
88. Sousa, K., Mendonça, H., Vanderdonckt, J., Rogier, E., Vandermeulen, J.: User interface derivation from business processes: a model-driven approach for organizational engineering. In: Proceedings of the 2008 ACM symposium on Applied computing, pp. 553–560 (2008)
89. Störrle, H.: On the impact of layout quality to understanding uml diagrams: Size matters. In: Dingel, J., Schulte, W., Ramos, I., Abrahão, S., Insfran, E. (eds.) Model-Driven Engineering Languages and Systems, pp. 518–534 (2014)
90. Störrle, H.: Diagram size vs. layout flaws: understanding quality factors of uml diagrams. In: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM '16, pp. 31:1–31:10 (2016)
91. Störrle, H.: On the impact of size to the understanding of uml diagrams. Softw. Syst. Model. **17**(1), 115–134 (2018). https://doi.org/10.1007/s10270-016-0529-x
92. Störrle, H., Fish, A.: Towards an operationalization of the "physics of notations" for the analysis of visual languages. In: Moreira, A., Schätz, B., Gray, J., Vallecillo, A., Clarke, P. (eds.) Model-Driven Engineering Languages and Systems, pp. 104–120. Springer, Berlin, Heidelberg (2013)
93. Sunkle, S., Saxena, K., Patil, A., Kulkarni, V.: Ai-driven streamlined modeling: experiences and lessons learned from multiple domains. Softw. Syst. Model. **21**(3), 1–23 (2022)
94. Sutter, H., Alexandrescu, A.: C++ coding standards: 101 rules, guidelines, and best practices. Pearson Education (2004)
95. Wang, Z.: A modeling approach for use-cases model in uml. In: 2012 IEEE Fifth International Conference on Advanced Computational Intelligence (ICACI), pp. 176–179 (2012). https://doi.org/10.1109/ICACI.2012.6463145
96. Weyssow, M., Sahraoui, H., Syriani, E.: Recommending metamodel concepts during modeling activities with pre-trained language models. Softw. Syst. Model. **21**(3), 1071–1089 (2022)
97. 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. Springer, Berlin Heidelberg (2013)
98. Wieringa, R., Maiden, N., Mead, N., Rolland, C.: Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. Requir. Eng. **11**, 102–107 (2006). https://doi.org/10.1007/s00766-005-0021-6
99. Wohlin, C.: Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, EASE '14. Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2601248.2601268
100. Wu, J., Yue, T., Ali, S., Zhang, H.: A modeling methodology to facilitate safety-oriented architecture design of industrial avionics

software. Softw. Pract. Exp. (2015). https://doi.org/10.1002/spe.2281
101. Zhang, L., Tian, J.H., Jiang, J., Liu, Y.J., Pu, M.Y., Yue, T.: Empirical research in software engineering–a literature survey. J. Comput. Sci. Technol. (2018). https://doi.org/10.1007/s11390-018-1864-x
102. Zhao, X., Gray., J.: Design guidelines for feature model construction: Exploring the relationship between feature model structure and structural complexity. In: Proceedings of the 7th International Conference on Model-Driven Engineering and Software Development - MODELSWARD, pp. 325–333. INSTICC, SciTePress (2019). https://doi.org/10.5220/0007388703250333
103. Zheng, C., Le Duigou, J., Bricogne, M., Eynard, B.: Multidisciplinary interface model for design of mechatronic systems. Comput. Ind. **76**, 24–37 (2016)

Shalini Chakraborty completed her bachelor's (2017) and master's (2019) in Computer Science from Calcutta University, India. She is pursuing her Ph.D., working in Software Engineering (SE), primarily focusing on Model-Based Engineering (MBE) and empirical research. She is interested in human factors in SE and social software engineering. Currently, she is employed at Reykjavik University as a Ph.D. candidate and teaching assistant.

Grischa Liebel is an Assistant Professor in Software Engineering and the director of the CRESS research centre at Reykjavik University, Iceland. He holds a PhD degree from Chalmers University, Sweden. His research focuses on human factors in software engineering, typically in areas such Modelling and Model-Based Engineering, Requirements Engineering, Processes, and Education. Much of Grischa's research is done in collaboration with industry (and with humans).