JOT
JOURNAL OF
OBJECT TECHNOLOGY

Journal of Object Technology | **RESEARCH ARTICLE**

Flexible Modelling: a Systematic Literature Review

Robbert Jongeling[*] **and Federico Ciccozzi**[*]
[*]Mälardalen University, Västerås, Sweden

ABSTRACT Canonical software and systems modelling regards only models that conform to modelling languages and that are created in modelling tools. In practice, these models are often supplemented with free-form activities such as sketching and informal diagramming. Flexible modelling has been proposed to benefit from the combination of free-form activities, from relaxed to no conformance to modelling languages, and canonical modelling with more or less strict conformance of diagrammatic elements to modelling languages. Various tools and approaches have proposed mechanisms to support flexible modelling. In this paper, we report on a systematic literature review of these tools and approaches. We present an analysis of the existing body of knowledge in this area and discuss open research challenges that can help the modelling community identify promising next steps in this area.

KEYWORDS Flexible Modelling, Model-Driven Engineering, Systematic Literature Review

1. Introduction

Free-form sketches and diagrams are commonly used means by which software engineers understand, design, and communicate about software systems (Baltes & Diehl 2014; Störrle 2017; Zarwin et al. 2014). These free-form artefacts may describe the intended structure or behaviour of the system and are, to some degree, models. Indeed, they are abstractions since they contain only the relevant aspects of the system for the descriptive or prescriptive purpose of the model (Brambilla et al. 2017). However, to benefit from the powerful manipulations supported by the model-driven engineering paradigm, additional underlying semantics is required so that these free-form artefacts can be considered and processed as canonical *models*. By "canonical", we refer to models that conform to a modelling language and that are created and viewed in a tool that may support their navigation and automated manipulation.

The paradigm of flexible modelling has arisen to bring human-friendly, easy-to-edit, and free-form artefacts closer to the more stringent models conforming to modelling languages. Flexible modelling considers all modelling practices in software engineering that range from "napkin sketches" on one extreme and models strictly conforming to modelling languages on the other. The vision for flexible models is thus to combine the best of both worlds, allowing editing freedom while at some point during development being able to enforce strict enough conformance to support automated actions on models. Our specific interest is in how flexible modelling approaches can help software engineers benefit from (i) sketching for free-form design and understanding, (ii) informal diagramming for communication and brainstorming, and (iii) stricter modelling for analysis, simulation, and transformation purposes.

In the last 15 years (from e.g. (Ossher et al. 2009) to (Jongeling et al. 2022)), work on multiple aspects of flexible modelling has been presented from the perspectives of various subdisciplines of software engineering, such as model-driven engineering, agile software engineering, and software architecture. While there are not many studies in absolute numbers, with this study, we aim at summarizing the knowledge gained from these different perspectives and identify what gaps remain to be filled, especially towards the realization of flexible modelling in practice. To do so, we perform a systematic literature review (SLR) following the well-established guidelines in software engineering (Kitchenham 2007).

The remainder of this paper is organized as follows. Section 2 discusses the history of work on flexible modelling and elaborates on the need for this SLR. Section 3 includes a detailed overview of our research method. Section 4 presents quantitative data resulting from our search. Section 5 presents

---

JOT reference format:
Robbert Jongeling and Federico Ciccozzi. *Flexible Modelling: a Systematic Literature Review*. Journal of Object Technology. Vol. 23, No. 3, 2024.

http://dx.doi.org/10.5381/jot.2024.23.3.a3

qualitative analysis on the presented motivations, needs, evaluations, and terminology used by the primary studies. Section 6 presents the technical characteristics of flexible modelling approaches presented by the primary studies. Section 7 extracts open research challenges. Section 8 concludes.

2. Background and Related work

In this section, we establish the need for an SLR on flexible modelling and discuss related initiatives in this area.

2.1. Background: Workshops on flexible modelling

An orthogonal means of getting insights into the history of this topic was to consider the venues that have explicitly solicited contributions on it. We highlight the following three venues.

FlexiTools At CASCON 2009, ICSE 2010, SPLASH 2010, ICSE 2011, and SPLASH 2013, workshops on flexible modelling tools (from 2010 named "FlexiTools") were held. The workshops have focused on: "bridging the gap between formal modelling and free-form authoring" (Kimelman et al. 2010). This formulation is in line with our definition of flexible modelling and shows a first interest in specific tooling to support it. The work from Correia and Aguiar (Correia & Aguiar 2013) analyzes works presented at these workshops.

XM Co-located with the MODELS conference, the XM (eXtreme Modelling) workshop was held between 2012 and 2014. The goal of the workshop was to explore the range between informal diagramming and strict conformance of models as in model-driven engineering (Di Ruscio et al. 2012). Hence this workshop series can be seen as a successor to FlexiTools. As explained in the remainder of the paper, several primary studies were gathered from this workshop and its successor.

FlexMDE In 2015, the XM workshop was renamed to FlexMDE (Flexible Model-Driven Engineering) and its first edition was held in conjunction with the MODELS conference. The call for papers of the last edition in 2019 described the limited flexibility of modelling tools as a major impediment to their adoption and thus a strong motivation for the workshop. From 2020 until the time of writing, there has not been any dedicated workshop on this topic at MODELS. Nevertheless, there remains some interest in this topic, with publications in other venues.

2.2. Related work: The need for a review

Research has found that many informal diagrams include some UML elements (Baltes & Diehl 2014). An explanation for this is that users want to keep some precision even when using free-form notations to communicate designs. In a survey with practitioners, UML was found to be the most commonly used modelling notation (Störrle 2017), but most commonly in a rather informal way, indicating the need to supplement strict models with more free-form notations. In that survey, Störrle found three distinct ways in which models are used: informally, for understanding and communication, semi-formal, for planning and documentation, and formal, for automated analysis and transformation. We notice that strict conformance is only needed for the latter activities, which make up a small minority of all tasks reported by the surveyed practitioners. Another aspect of this desired informality is indicated by UML models being persisted as images, rather than model files. In a repository mining study aiming to build a corpus of UML diagrams on GitHub, a majority of the collected UML diagrams were identified in images (Robles et al. 2017).

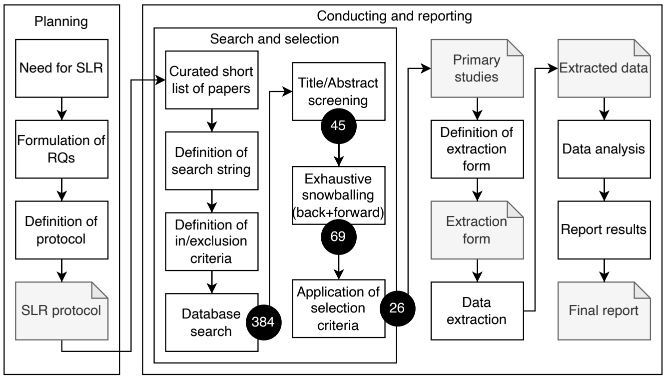

Figure 1 Overview of our SLR process, values indicate number of total papers after each step.

The common usage of informal notations in combination with modelling indicates a desire for the adoption of flexible modelling techniques. Therefore, we are interested to know what advances have been made in the field, which approaches and tools have been used successfully, and what the open research gaps towards realizing the flexible modelling vision are. To the best of our knowledge, there is currently no secondary study on this topic available in the literature. In a reflection from 2013 (Correia & Aguiar 2013), the authors derive patterns by which flexibility is provided in a set of papers that present flexible modelling tools, discussing for each pattern which problem it addresses, how that problem is solved, and in which tools the pattern is used. Furthermore, the paper (Guerra & de Lara 2018) provides an excellent overview of several flexible modelling approaches. In this paper, we extend these overviews by means of a systematic review of the literature and analysis of flexible modelling approaches in terms of the motivations and technical characteristics, as provided by the authors.

3. Research method

Figure 1 shows a schematic overview of our research method. In establishing this method, we followed established guidelines for systematic literature reviews in software engineering (Kitchenham 2007).

3.1. Planning

We started the planning phase by identifying the need for an SLR, as described in Section 2. After this initial exploration, we defined the following research questions and corresponding goals.

RQ1 *What are the publication trends of research studies in the context of flexible modelling?*

Goal: classify primary studies in terms of relevant venues and contribution types.
Answer: This RQ is answered in Section 4.

RQ2 *What are the characteristics of the existing approaches and tools for flexible modelling?*
Goal: identify what technical characteristics are provided by the proposed approaches and study which types of flexibility are desired.
Answer: This RQ is answered in Section 5 and Section 6.

RQ3 *What are open research challenges towards supporting flexible modelling in practice?*
Goal: assess the extent to which approaches have been evaluated and identify open research challenges.
Answer: This RQ is answered in Section 7.

Upon definition of these research questions, we formulated the remainder of the research protocol that is described in this section.

3.2. Search and selection strategy

Search and selection is the first major part of conducting the study and we outlined our step-wise strategy in Figure 1. Before starting a database search, we defined a search string. The search string was validated by comparing search results to a curated short list of papers that we believed should be included in the search results. That shortlist included primary studies P8, P12, P18, P20, P25, and P26 (see Table 1) and was compiled as a result of an informal exploration of the literature by experimenting with several tentative search strings. This resulted in the following search string that returned in the result most of the studies in the shortlist: `ALL ( ( "flexible modeling" OR "flexible modelling" ) AND "software engineering" ) AND ( LIMIT-TO ( LANGUAGE , "English" ) )`. The search string searches within “all fields” in Scopus, for flexible model(l)ing and software engineering, furthermore the results are limited to papers written in English.

Not all search results shall be included as primary studies in our SLR. If a study meets any of the following exclusion criteria, it is no longer considered a candidate for inclusion in the study.

EC1 *The study is an opinion or vision paper.* Motivation: Since we seek to collect and compare the characteristics of proposed approaches, we exclude those papers that are only providing an opinion or vision.

EC2 *In the study, “model” does not refer to a software engineering artefact.* Motivation: The term “model” is overloaded and used in many contexts, we limit our interest to the software engineering domain.

Furthermore, for the remaining candidates, we defined the following inclusion criteria that a study must meet as one to be included:

IC1 *Studies that present a technique, approach, method, or tool that supports flexible modelling.* Motivation: We are interested in comparing concrete approaches.

IC2 *Studies that are written in English.* Motivation: English is the de-facto standard language of communicating research results in our community.

IC3 *Studies for which the full text is available.* Motivation: The full text is required for a complete analysis.

We used Scopus to perform the initial search, which we later supplemented with exhaustive snowballing. The initial search resulted in 388 papers, of which 4 were automatically marked as duplicates when we imported the results in the tool Covidence, which we used for the initial screening. Both authors read all titles and abstracts and applied inclusion and exclusion criteria. There were 27 papers (7% out of the 384 total unique results) for which the authors did not mark the same decision on inclusion or exclusion. After a discussion between the authors, all conflicts were resolved and 45 papers in total were included in the initial set before snowballing. Then, the first author performed exhaustive backward and forward snowballing (Wohlin 2014) on the set of included papers. In the first round, 23 new papers were added, in the second round 1 more paper was added and in the third round, no more papers were added and so the snowballing was completed. After full-text reading, out of the in total 69 papers (45 initial + 24 snowballed) we have selected 26 primary studies that met all the selection criteria. As part of the final selection, we have also selected single publications as representative when there were multiple papers on a single approach. For example, our initial selection contained 7 papers on FlexiSketch (Wüest et al. 2019), so we only selected the most recent and most complete one to be part of this study. In total, we have excluded 21 papers based on choosing a single representative study, the remaining 22 excluded papers have been excluded due to EC1 (2) and IC1 (20).

3.3. Data extraction

The result of the search and selection phase is the list of included primary studies in Table 1. We continued by defining a data extraction form, subsequently performing the data extraction, and analysing the extracted data.

To support the data extraction process, we created an initial data extraction form based on our research questions and the corresponding goals. The complete form is included in our replication package (Jongeling 2023), which consists of a workbook with multiple spreadsheets showing the selection and data extraction steps. We initially extracted keywords and short descriptions for each data item from the primary studies. When we encountered data of interest in the primary studies, we extended the data extraction form and revisited old papers to supplement the missing information. In our analysis in Section 5 and Section 6, we present and discuss this data.

3.4. Threats to validity

We now describe the four aspects of threats to validity as distinguished by Wohlin et al. (Wohlin et al. 2012) and how we mitigated them to maximise the validity of this study.

Construct validity It deals with the degree to which the selected primary studies are relevant to answering our research

Table 1 List of primary studies

ID	Title	Authors-Year
P1	SUMLOW: early design-stage sketching of UML diagrams on an E-whiteboard	(Chen et al. 2008)
P2	Flexible modelling tools for pre-requirements analysis: Conceptual architecture and research challenges	(Ossher et al. 2010)
P3	From a freeform graphics tool to a repository based modelling tool	(Peltonen et al. 2010)
P4	Cross-layer modeler - A tool for flexible multilevel modelling with consistency checking	(Demuth et al. 2011)
P5	A Spectrum of Flexibility – Lowering Barriers to modelling Tool Adoption	(Kimelman & Hirschman 2011)
P6	EuGENia live: A flexible graphical modelling tool	(Rose et al. 2012)
P7	Collaborative Creativity: From Hand Drawn Sketches to Formal Domain Specific Models and Back Again	(Bartelt et al. 2013)
P8	Programmatic muddle management	(Kolovos et al. 2013)
P9	On Lightweight Metamodel Extension to Support Modelling Tools Agility	(Bruneliere et al. 2015)
P10	CEL: Touching software modelling in essence	(Lemma et al. 2015)
P11	Using Free modelling as an Agile Method for Developing Domain Specific modelling Languages	(Golra et al. 2016)
P12	A metamodelling framework for promoting flexibility and creativity over strict model conformance	(Hili 2016)
P13	JSMF: A flexible JavaScript Modelling Framework	(Sottet & Biri 2016)
P14	Safe model polymorphism for flexible modelling	(Degueule et al. 2017)
P15	A posteriori typing for model-driven engineering: Concepts, analysis, and applications	(Lara & Guerra 2017)
P16	An example is worth a thousand words: Creating graphical modelling environments by example	(López-Fernández et al. 2019)
P17	OctoUML: An Environment for Exploratory and Collaborative Software Design	(Vesin et al. 2017)
P18	On the quest for flexible modelling	(Guerra & de Lara 2018)
P19	Grass-Root Enterprise modelling: Issues and Potentials of Retrieving Models from Powerpoint	(Reiz et al. 2018)
P20	FlexiSketch: a lightweight sketching and metamodelling approach for end-users	(Wüest et al. 2019)
P21	CouchEdit: A relaxed conformance editing approach	(Nachreiner et al. 2020)
P22	Towards On-The-Fly Creation of Modelling Language Jargons	(Bider et al. 2021)
P23	Towards Facilitating the Exploration of Informal Concepts in Formal modelling Tools	(Gogolla et al. 2021)
P24	Facet-oriented Modelling	(Lara et al. 2021)
P25	From Object to Class Models: More Steps towards Flexible Modelling (Short Paper)	(Gogolla et al. 2022)
P26	From Informal Architecture Diagrams to Flexible Blended Models	(Jongeling et al. 2022)

questions. To limit this threat, both researchers independently performed the title and abstract screening of all papers resulting from the initial search. Out of 384 unique results, we had 27 cases where we did not immediately agree on their inclusion or exclusion. We discussed each of these and, in the end, through rigorously applying our inclusion and exclusion criteria, we solved the conflicts. Furthermore, by following the systematic literature guidelines, we are confident that we included the relevant literature.

Internal validity It concerns the validity of conclusions around causality between factors. The first step to mitigating this threat is the definition of our data extraction sheet with properties following our research questions. Furthermore, in our analysis, we focus on describing the technical aspects that are explicitly presented by the proposed solutions, thereby limiting the possible impact of unknown external factors, should they exist.

External validity It concerns the generalizability of the findings and the extent to which they are of interest to people external to the flexible modelling area. To ensure the completeness of our search, we complemented the initial automatic search with exhaustive forward and backward snowballing. In addition, we described in Section 2 the place of flexible modelling within the model-driven engineering landscape and the value of exploring the topic.

Conclusion validity It concerns with the validity of the conclusion from the gathered data. We limited this threat by iteratively refining the characteristics extracted from the primary studies. We also followed well-established guidelines for SLRs in software engineering (Kitchenham 2007). Furthermore, we documented our steps in our publicly available replication package (Jongeling 2023).

4. Publication trends

This section reports quantitative results of our search and selection process towards answering RQ1. We present publication trends in the area of flexible modelling in terms of publication years and venues of the collected primary studies.

4.1. Publication year

Figure 2 shows the primary studies as published per year. The average number of publications in the collected window is 1.7 per year, roughly stable throughout the studied period, with a slightly increasing trend. We thus see stable, but little, interest in the topic after discontinuation of the dedicated workshops as listed in Section 2.1.

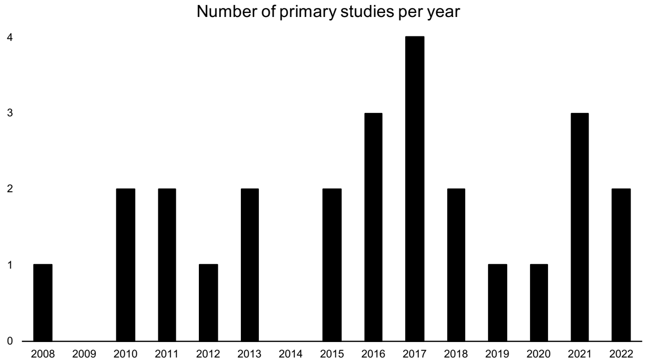

Figure 2 Publication trend

4.2. Publication venues

The primary studies have been published in workshops (10), conferences (10), and journals (6). Five venues have hosted at least two primary studies, these are listed in Table 2. From the table, we see a preference for general modelling/software engineering-related venues, and less interest in venues specialized in flexible modelling, or venues related to software architecture.

Table 2 Venues hosting at least two primary studies

Venue	#
XM/FlexMDE workshop at MODELS	4
ACM Transactions on Software Engineering and Methodology (TOSEM)	2
European Conference on Software Architecture (ECSA)	2
International Conference on Model Driven Engineering Languages and Systems (MODELS)	2
International Journal on Software and Systems Modelling (SoSyM)	2

5. Motivational characteristics and terminology

In this section, we discuss characteristics of primary studies related to the provided motivation and terminology used by the primary studies. The following are qualitative results towards answering RQ2.

5.1. Motivation

We categorize the motivations for flexible modelling approaches provided by the authors of the primary studies. We note that all these challenges are related to flexibility with respect to conformance, which ranges from no conformance, thus sketching or informal diagramming, to complete conformance, thus strict modelling. Correspondingly, arguments for flexibility are given from two directions, indicating the limitations of both extremes. We observe that a majority of approaches starts from models and seeks ways to allow more freedom (less rigid conformance) in the process of creating models and metamodels.

Table 3 shows six identified types of motivation for flexible modelling. The first three consider informal diagramming as the starting point, hence *from diagramming*. Due to their informal nature, these diagrams can not be read nor processed by automated means. Therefore, their use is limited to communication between stakeholders and they cannot be used for automated manipulations. Moreover, in situations where both informal diagrams and models are used in the development process, knowledge between them cannot easily be shared. To benefit more from the knowledge captured in these diagrams, it is required to introduce some degree of conformance.

The next three types of motivation in Table 3 start instead from canonical modelling, hence *from modelling*. The rigidity of models and their need to strictly conform to metamodels is an often cited challenge, especially in the early phases of the design when creativity and freedom are highly valued and often required. Therefore, looser conformance is sought to give engineers the freedom to capture their ideas. Beyond the rigidity of the models themselves, the modelling process is also mentioned as too rigid to facilitate the reality of modelling practices. Particularly, the need to define metamodels up-front, thus fixing syntax and semantics, is not compatible with the common needs for incremental and iterative development of models and thus modelling languages. Lastly, some primary studies specifically refer to the rigidity of modelling tools when enforcing conformance and identify it as a barrier to the use of those tools for modelling tasks that would benefit from temporary non-conformance. A consequence of the latter is that informal tools and diagrams are used in parallel for easier sketching and communication, thus leading to the challenges categorized as "from diagramming" in Table 3.

5.2. Needs for and evaluations of flexible modelling

Here, we study the evidence provided for the needs for flexible modelling and for the evaluation of proposed approaches. A minority of primary studies (P2, P5, P7, P9, P11, P26) shows concrete use cases from which the needs for flexible modelling are elicited. Similarly, a minority of primary studies (P1, P5, P7, P11, P17, P20, P26) includes evaluations with actual users. From these seven studies, 4 include exclusively practitioners from industry but do not specify how many exactly (P1, P5, P7, P26) and P11 and P20 include partially practitioners (4 and 9, respectively). P11 and P20 also include others, 3 researchers and 8 master students, respectively. The evaluations of P17 include in total 2 postdocs, 13 PhD students, and 15 master students.

Overall, the majority of works is evaluated through tool prototypes and running examples. This indicates a need for more realistic use cases that can be evaluated and addressed by practitioners, since it is unlikely that approaches solely motivated by – and evaluated on – simple examples are applicable in industrial settings.

5.3. Terminology

During the review, we encountered several different terms used to refer to flexible modelling and related concepts. Table 4 shows terminology as used in the primary studies. The terms refer either to flexible modelling directly or to concepts related to it. There seems to be most agreement on the term flexible modelling and therefore we also used that term in this work.

We grouped the terms by the similarity of concepts they describe or by the context in which they are used. For example, sketch recognition and example-based metamodelling are both used for bottom-up metamodelling. The differences between the various terms is at first sight unclear, and can only emerge from discussing the various characteristics of the approaches as discussed in Section 6. Later in this paper (see Table 8), we distinguish the various mechanisms by which flexibility is achieved in the primary studies, and thus, what these terms refer to concretely.

6. Technical characteristics

In addition to the motivation types discussed in Section 5, the following are further qualitative results towards answering RQ2, focusing specifically on the technical characteristics of the approaches as presented in the primary studies. We first investigate the target scope of the approaches to be able to better identify their technical characteristics.

6.1. Targeted domains

First, we notice that the primary studies target various subdomains of software engineering, together covering a broad set of software engineering activities. The chosen application domain of a study determines to a large extent the functionalities provided by the approach, even though underlying principles of flexible modelling may be shared. Table 5 provides an overview of targeted domains as we identified them in the primary studies. Beyond general software design modelling and software architecture modelling, several approaches aim at more specific types of these domains such as object-oriented modelling or multilevel modelling and we have categorized them accordingly. In addition, there are approaches aiming at different domains entirely such as enterprise modelling or business analysis.

Most approaches are related to (graphical or diagrammatic) domain-specific modelling, which is a domain where flexible modelling can provide a rather natural way to interact. Commonly, these approaches facilitate the bottom-up creation of graphical domain-specific languages (DSLs) starting from sketched or informal input. In this way, the rigidity of requiring an up-front model definition is avoided in favour of more flexibility in the initial creation and evolution of the DSL, as well as in its usage.

6.2. Targeted development phase

Similar to the targeted domains, also various development phases are targeted by the primary studies, Table 6 provides an overview. If no particular focus is indicated in the studies, we classified them as targeting the general "development" phase. Other approaches are notably categorizable into focusing on language or software development, on early design phases or on the iterative development of languages or designs, aligned with agile principles.

A common argument for aiming at early design phases is the inherent vagueness and ambiguity of software requirements, architecture, and design at those stages. Hence, committing to a particular modelling language may hinder creativity and ability to communicate quickly and freely the ideas that need to be shared in this phase. Iterative approaches benefit from flexibility in terms of freedom and creativity of gathering informal input, and from the formality of expressing the designs in canonical models to provide automated model manipulations.

Indeed, a fundamental underlying reason for the need for flexibility may be that the abstraction level and intended scope of the model change over time; what is a good enough abstraction early on needs to be refined in later iterations. Then, keeping existing models up to date is clearly preferred over discarding them and creating new ones. Hence, flexibility is aiming at supporting this change of purpose of models over time, while building on previous versions rather than starting from scratch.

We note that some approaches are aiming at language development; in these cases, flexibility spans both the model and

Table 3 Motivations for flexible modelling

	Motivation type	Challenges overview	Primary studies
From Diagramming	Not automatically readable	Informal diagrams lack modelling advantages such as semantics, multiple views and consistency management	P2, P3, P26
	Not automatically processable	Persistence, maintenance and processing of information from informal diagrams into development artefacts	P1, P3, P7, P8, P10, P19, P20, P22, P26
	Not automatically combinable	Both informal diagrams and models are used, so there needs to be a way for them to be combined	P17, P18, P19, P20, P22
From Modelling	Rigidity of metamodels	Rigid metamodels restrict the creativity of modelling which is needed especially for understanding and early design	P1, P2, P3, P5, P7, P9, P10, P12, P13, P14, P15, P18, P20, P21, P22, P23, P24, P25
	Rigidity of process	Defining metamodels up-front does not allow for the required collaborative, iterative and incremental development of (graphical) (domain-specific) models and languages	P4, P6, P7, P8, P11, P12, P13, P16, P17, P18, P23, P24, P25
	Rigidity of tools	Enforcing strict conformance leads to using parallel tools for informal diagramming	P5, P7, P10, P21

Table 4 Terminology used in the primary studies, grouped by common terms

Related terms	Primary studies
Sketch recognition, Grass-root modelling, Example-based metamodelling (also “bottom-up metamodelling”)	(P1), (P19), (P16)
Flexible modelling, Flexible multilevel modelling, Flexibility in modelling tools/environments, Natural modelling	(P2, P3, P5, P6, P13, P14, P16, P18, P20, P23, P24, P25, P26), (P4), (P9, P12, P22), (P13)
Muddle management, Lightweight metamodelling	(P8), (P20)
Free modelling	(P11)
A-posteriori model typing, Relaxed conformance	(P15), (P21)
Modelling language jargons	(P22)
Facet-oriented modelling	(P24)

the metamodel levels. In these cases, metamodels are typically created bottom-up by first drawing model elements and defining corresponding metamodel elements later.

6.3. Types of artefacts

From the previous aspects, we noted that approaches target different domains and development phases. It follows that the types of development artefacts that flexibility efforts focus on differ across the primary studies, as shown in Table 7. We see in fact a close relationship between the targeted artefact and the targeted domain, as shown in Table 5.

Table 5 Targeted domains

Domain type	Primary study
Software design modelling	P1, P3, P7, P9, P17, P20
Business analysis	P2
Multilevel modelling	P4
Software architecture modelling	P5, P26
(graphical or diagrammatic) domain-specific modelling	P6, P8, P11, P14, P16, P21, P24
Object-oriented modelling	P10, P23, P25
General model-driven engineering	P12, P13, P15, P18
Enterprise modelling	P19, P22

Works focusing only on sketch recognition were excluded as per our inclusion criterion IC1, which requires studies to present a flexible modelling approach. Therefore, except for approaches offering flexible modelling starting from hand-drawn artefacts such as whiteboard sketches, all other considered artefacts are digital. Six approaches consider specifically informal diagrams as created in already existing general-purpose diagramming tools or in other formats such as spreadsheets. These approaches focus on non-experts who have typically already started diagramming (or modelling textually) in these tools and may benefit more from these informal representations if they

Table 6 Targeted development phases

Phase	Primary study
Early design	P1, P3, P7, P17, P19, P20, P22
Pre-requirement analysis	P2
Development	P4, P12, P15, P18, P23, P24, P25
Iterative design and development (agile)	P5, P9, P10, P13, P26
Early language development	P6, P16
Agile language development	P8, P11, P14, P21

were to be considered as models. Other approaches providing a similar bottom-up approach may start from graphical syntax examples of domain-specific languages, as expressed in dedicated tools or language workbenches. Approaches that provide flexibility for UML models, models in EMF, or general MDE models typically target modelling experts and provide flexibility, e.g. allowing less stringent conformance to the metamodel.

Thus, from the targeted artefact types emerge two directions of flexible modelling. On the one hand, some approaches start from informality and aim to gain benefits from adopting more strict modelling. On the other hand, we have modellers who wish to gain from the benefits of less strict conformance to introduce more informality in their models.

Table 7 Targeted artefact types

	Flexible artefact type	Primary study
Diagrams	"Whiteboard" sketches	P1, P7, P20
	Informal diagrams (in office tools, diagramming tools, or general-purpose tools)	P2, P3, P4, P8, P19, P26
	Architecture diagrams and requirements definitions	P5
	Graphical syntax examples of DSLs	P6, P11, P16, P21
Models	Metamodels	P9, P11
	UML (class/object)	P10, P17, P23, P25
	Models in EMF	P14
	General MDE models	P12, P15, P18, P24
	Textual models and metamodels	P13
	Fractal enterprise models	P22

6.4. Flexibility mechanisms

We discuss the mechanism by which flexibility is provided, independently of the targeted domains and artefact types. Table 8 lists the mechanisms as extracted from the primary studies. In the table, we grouped three main strategies for providing flexibility: "flexible input", "bottom-up metamodelling", and "flexible typing".

When considering input methods, some approaches accept hand-drawn sketches, for example on whiteboards, or created through drawing software. Other approaches allow the editing of modelling information through tables or lists that represent information from the models. The studies included here also provide mechanisms to manipulate these artefacts once they are translated to models, and sometimes also to synchronize between manual notations and the representation as a model. Thus, flexibility is primarily concerned with syntax.

Bottom-up metamodelling allows (typically graphical) notations to be created first and to keep track of a metamodel describing the used notations either simultaneously with the modelling activities or a posteriori. Creating a mapping between the concrete syntax and modelling concepts represented by it can be done in many ways. One way is manually, by which the modeller creates a mapping between the concepts, either by explicitly linking concepts or by creating annotations for informal elements to be mapped to model element counterparts. Another way is to record a mapping during the diagram editing, either manually or automatically. One approach proposes to discover these mappings by automated means, relying on shape recognition. Overall, all these approaches start from informally capturing particular information and then introducing formality by defining metamodel elements for them.

The other general strategy for flexibility that we discern is related to typing, for which various approaches have been proposed. One way is by not enforcing a particular type or by configuring the strictness by which a type should be checked. Thereby, the conformance relation may be controlled in a fine-grained way, for each type. Another similar approach is to allow elements to be assigned multiple types at the same time. This multiple typing, or dynamic retyping, allows conformance to multiple metamodels, opening further possibilities for flexibility. Another method providing flexibility is a posteriori typing, i.e., delaying assigning types to model elements. A posteriori typing provides flexibility by similar means as bottom-up metamodelling, i.e. by allowing the initial introduction of untyped elements. There are papers focusing specifically on the means by which to provide type inference, for example by means of classification algorithms (Zolotas et al. 2019) (this paper relies on the flexible modelling approach presented in P8 and is therefore not included as a primary study).

6.5. Collaborative flexible modelling

Driven by the increasing complexity of software-intensive systems, modelling has become an inherently collaborative activity. Thus modelling approaches shall consider and support collaboration, be it synchronous (or real-time) or asynchronous. Four studies support real-time collaboration (P1, P7, P17, 20) and five studies support asynchronous collaboration (P6, P8, P9,

Table 8 Flexibility mechanism types

	Main flexible mechanism type	**Primary study**
Flexible Input	Sketch recognition and manipulation of inferred models	P1, P7, P17
	Editing models via tables/lists representing data from the model	P5
Bottom-up metamodelling	Bottom-up metamodelling through legend/palette defining mapping between informal diagram and modelling concepts	P2, P6, P11, P26
	Bottom-up metamodelling through recording mapping to model elements during diagramming	P3, P4, P12
	Bottom-up metamodelling through annotations to diagram elements to define model elements	P8, P16, P20
	Bottom-up metamodelling through automatically mapping the concrete syntax to abstract syntax	P19
Flexible Typing	Creating overview diagrams (vertical)/Filtering elements	P5, P9, P10
	Not enforcing typing or delaying (a posteriori) typing	P5, P15, P21, P22, P25
	Multiple typing or dynamic retyping	P9, P14, P24
	Configurable dynamic type-checking	P13, P18, P23

P16, P25).

We left out the majority of primary studies, which do not explicitly address how their approach fits in collaborative settings. When considering collaboration, some approaches consider real-time collaboration. Two types of real-time collaboration are mentioned, (i) in-person, such as collaborative sketching on whiteboards, and (ii) distributed, such as simultaneously editing the same file and seeing live updates. Other approaches do not consider real-time collaboration but rather the asynchronous editing of flexible artefacts by multiple engineers over time. Presumably, the approaches from papers not explicitly mentioning collaboration in their scope lean towards the latter form of collaboration. However, we observe that collaboration aspects are not a priority when presenting flexible modelling approaches.

6.6. Tool support

From the 26 primary studies, 13 share an implementation or tool. The majority of these are research prototypes with no further development or maintenance. The results of P14 are available as part of the tool Melange. The results of P20 are available as part of the tool FlexiSketch[1]. Other implementations have had no further updates since the publication, or have otherwise been superseded, for example Eugenia (P6) was discontinued in Epsilon 2.5[2]. P23 and P25 discuss the tool USE (UML-based Specification Environment), which is available[3]. P15 and P24 evaluate their approach on top of the tool MetaDepth, which is available[4].

The maturity level and availability of flexible modelling tools is low, and is an area for improvement in future research.

6.7. Reported limitations and unsolved challenges

In addition to the previously discussed technical characteristics, we are also interested in the limitations and unsolved challenges of the presented solutions as reported by the authors of the primary studies. Table 9 lists them. The first category we identified deals with *technical* limitations, related often to implementation details that are not fundamental to the approaches. This type of limitation is common in early research results and is not of interest in this study if it is only a matter of a lack of time or resources to extend a prototype implementation. Instead, in the remainder we focus on challenges beyond those related to extending existing implementations.

The second category combines challenges that are related to the flexible modelling *process*. Five primary studies mention challenges in providing automated assistance for users, possibly not modelling experts, to move along the path towards modelling when starting from informal diagrams. A co-occurring challenge is to evaluate the usefulness of such automated assistance to various types of users. Closely related to these two challenges, and a core aspect of flexible modelling processes, is deciding when to require rigidity and disallow informality, as also mentioned by five primary studies. Table 3, discerns two arguments: informal diagramming lacks modelling advantages, and modelling lacks the freedom of informal diagramming. Both are relevant, for different artefacts at different times during the engineering process. This depends on factors internal to the artefact such as its purpose and contents, and also on factors external to the artefact such as its relationships to other development artefacts and the way of working in a specific setting. For the former, three primary studies note that automated type inference is hindered when the artefacts are too informal, and therefore some semi-automated support with user guidance is needed. For the latter, one concrete challenge is to allow closer and real-time collaboration on the flexible artefacts between various stakeholders, and another broad challenge is to incorporate flexible modelling principles in existing modelling processes. All these challenges are related to the flexible modelling process and understanding the right amount of strictness that shall be enforced.

The third category that we identify in Table 9 relates to the *quality* of the defined languages and models conforming to them. A challenge when creating both of these bottom-up is to ensure

---

[1] https://www.ifi.uzh.ch/en/rerg/research/flexiblemodeling/flexisketch.html

[2] https://eclipse.dev/epsilon/doc/eugenia/

[3] https://www.db.informatik.uni-bremen.de/projects/USE-2.2.0/

[4] https://metadepth.org/

that model management tasks are still facilitated and to ensure that automated analysis can to some extent be performed. The applicability of a flexible modelling approach relies on these features since without them automated support is not achievable.

Lastly, we discuss challenges that are *fundamental* to flexible modelling. One question is whether the desired flexibility in the modelling process can be provided by a single tool or that it requires the integration of multiple tools to cover the entire spectrum from informality to formality as well as covering all needed degrees of flexibility between the two extremes (Kimelman & Hirschman 2011). A related aspect is the synchronization between various internal representations of flexible models, which needs a high degree of automation to not interfere with the modelling process. Furthermore, flexible modelling does not escape from the challenges of evolution, both of models themselves and of the tooling that supports flexible modelling, be them dedicated ones, or pre-existing tools for which the interfaces might change in future updates. A final intrinsic concern is the one dealing with the cognitive aspects of modellers. Flexible modelling has emerged as an approach to assist on the one hand non-experts in starting modelling bottom-up, and at the same time expert modelers to allow them to introduce more informality in models. The complexity of solutions should be carefully observed and evaluated in user studies, to ensure that the target audience is reached.

7. Discussion

In this section, we provide our interpretation and reflections on the results and analysis presented in Sections 5 and 6.

7.1. Spectrum of flexibility

Based on the primary studies, we derived a spectrum of flexible modelling between sketching on one end and canonical modelling on the other. Along the spectrum we find varying levels of conformance to modelling languages ranging from none at all to complete. On one extreme, there is completely free-form sketching as we may find on paper or whiteboards, no palette of shapes nor semantics of any elements are pre-defined. One step closer towards modelling, we may encounter diagramming, as for instance in MS Visio diagrams or similar tools; yet no underlying semantics of the shapes is defined, but diagrams may use a pre-defined set of shapes that carry some implicit meaning. An enhanced version of that is an informal diagramming practice where semantics is defined for some of the pre-defined shapes. Further still, we may encounter models conforming to modelling languages, but partially supplemented with shapes not conforming to the modelling language, or untyped elements. Eventually, we end up at the other extreme, canonical modelling, where models completely conform to modelling languages. In summary, the spectrum reaches from no conformance to complete conformance to the syntax and semantics of a modelling language.

Diagrammatically, a flexible modelling spectrum can be seen as the line in Figure 3. In this spectrum, we include only the intermediate steps found in the primary studies rather than all possible ones. The lines marked A-E show the coverage found in the primary studies, as listed in Table 10. However, the ranges A-E do not mean that each approach supports all the intermediate steps. For example, approaches covering practices from sketching to canonical modelling may only cover sketch recognition of shapes and canonical (e.g., UML) modelling. P9 and P10 are not included because they cover an orthogonal, vertical, direction of flexibility, related to the abstraction level of the model.

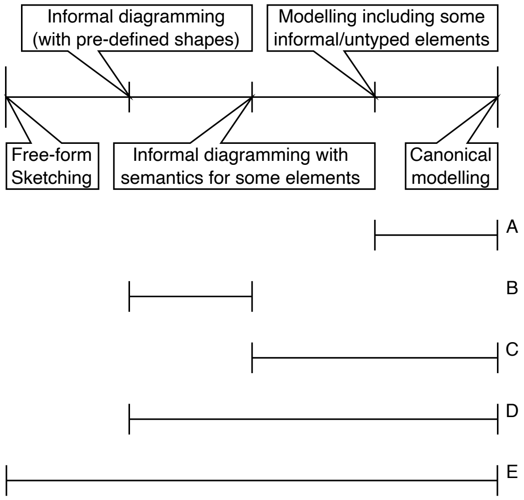

Figure 3 Modelling practices visualized as a spectrum between sketching on one extreme and canonical modelling on the other.

Table 10 Locations on modelling spectrum, as in Figure 3

Category	Primary studies
A	P12, P13, P14, P15, P22, P23, P24, P25
B	P2, P3, P26
C	P4, P11, P21
D	P5, P6, P8, P16, P18, P19
E	P1, P7, P20

From the approaches listed in Table 8, many support only one direction along the spectrum of Figure 3. Most commonly (P1, P2, P3, P6, P8, P12, P16, P18, P19, P20), flexibility is supported only from left to right along this spectrum, allowing for formalising informal elements. A few approaches (P22, P23, P25) support only flexibility from right to left, allowing for informal extensions to formal elements. Three approaches (P14, P15, P24) are rather stable on the spectrum, since the models always stay formal, although it could be said that they support bidirectional flexibility by allowing flexibility in typing. The other approaches from Table 8 (P4, P5, P7, P11, P13, P17, P26) support to some extent flexibility in both directions along the spectrum. That is, they provide bottom-up support, starting from some form of informal input they help the user transit to more formal notations.

Table 9 Limitations, unsolved challenges, and open questions

	Limitation, challenge, or question	**Primary study**
Technical	Technical limitations of implementations or user interface design (not fundamental to approach)	P1, P2, P3, P16, P21
Process	Supporting (non-expert) users in the path towards modelling and measuring how well the tool performs at that	P2, P5, P11, P18, P20
	Deciding the right time to move away from flexibility and introduce rigidity	P6, P8, P11, P23, P25
	(Automated) Type inference, possibly hindered by informality	P13, P19, P26
	Closer and real-time collaboration between software engineers and domain experts	P6
	Incorporating flexible modelling in the modelling process	P18
Quality	Model management (making approaches work and extending analysis functionality)	P15, P16, P17, P24
	Language definition and quality	P6, P11
Fundamental	Single tool versus integration of multiple for flexible modelling	P5
	Synchronizing representations	P5, P9
	Evolution of flexible modelling editor	P16, P17
	Cognitive difficulty	P22, P26, P24

From the motivations in Table 3, we know that there are equally many challenges reported with the rigidity of modelling. Therefore, we believe that only one-directional support for flexibility does not suffice. To cover the full range of flexibility, we envision that an approach shall provide flexibility in both directions along this spectrum, allowing for both bottom-up approaches and for approaches allowing informality to be introduced in combination with formal modelling. Moreover, recognizing that both are needed at different times during development, flexible modelling should provide the possibility of freely moving back and forth along the spectrum at any time in the development.

The spectrum in Figure 3 is a way to reason about the different levels of conformance. There are two studies (Bruneliere et al. 2015; Lemma et al. 2015) that do not fit this spectrum because they provide flexibility in an orthogonal, or "vertical" (Jongeling & Ciccozzi 2023), manner. That is, flexibility consists in allowing changing the level of completeness of a model, rather than the level of its conformance. Which dimensions shall be considered to meet the flexibility needs in practice is an open research question. Besides conformance and completeness, an additional dimension could for example be uncertainty (Troya et al. 2021).

Uncertainty, in particular epistemic uncertainty, is common in the same engineering phases as the need for informal and incomplete modelling and there are benefits to making this uncertainty explicit (Zhang et al. 2016). Moreover, uncertainty is linked to the other aspects of flexibility since higher uncertainty may lead to the inability to complete a certain model, and it may similarly be difficult to capture uncertain aspects using complete conformance. Instead, to express uncertainty, informal notations or annotations may be very useful. These may then be refined in later development phases. Therefore, uncertainty may be seen similarly to conformance as a dimension of flexibility that shall be supported to express, navigate, and manipulate flexible artefacts.

7.2. Open research challenges from the primary studies

We summarize open research challenges as reported by the primary studies, towards answering RQ3.

How to achieve successful flexible modelling guidance Modelling complex systems is a cognitively intensive task. Therefore, any approach aiming to enhance the abilities of the modelers should focus on not increasing the difficulty of modelling with limited additional gains. Indeed, sketching or informal diagramming is often chosen precisely because of its initial ease of use. We consider informal diagramming to cover a number of steps along the flexible modelling spectrum in Figure 3. Informal diagrams are those for which either no semantics is defined or they are not expressed in a form that allows for their automated manipulation. For example, an informal diagramming tool such as MS Visio or draw.io can support the creation of UML models exactly following the syntax and semantics as outlined in the UML standard. However, we would still consider such diagrams as informal because these tools do not consider what is drawn as a *model*. Querying or transforming these diagrams is not supported, since they are not seen as models; tools recognize and store different shapes, at most, but there are no semantics associated with these shapes. Hence, types and other modelling information need to be captured in informal diagrams for them to be processable as models.

To benefit fully from both informal diagramming and canonical modelling, there is a need for flexible modelling tools to provide guidance in the flexible modelling process. In particular, there is value in providing automated suggestions on when to loosen or tighten conformance. For example, users could be allowed to freely edit diagrams and get suggestions on the best fitting metamodel element. Alternatively, the tool could suggestion required extensions of the metamodel to formalize the freely added elements. Moreover, automated suggestions may be provided to help engineers introduce more formality in informal diagrams. This guidance shall also ensure language quality and the ability to use model management features. An example approach in this direction allows engineers to pro natural language instructions to a chatbot to transform informal models (that contain bits that do not conform to the metamodel) to models conforming to the metamodel (Pérez-Soler et al. 2019).

How to support collaborative flexible modelling Secondly, modelling is a collaborative effort and flexible modelling shall support collaboration between multiple stakeholders involved in modelling various aspects of complex systems. It is an open question if collaboration shall be real-time, or if other support for contributions across multiple stakeholders and throughout the development cycle is needed. A related open question is if flexible modelling support shall be provided by one single tool, or if it should be seen as an emerging functionality from the combination of multiple tools, as first brought up in (Kimelman & Hirschman 2011). In both cases, synchronizing various representations of models is an underlying challenge.

7.3. Other open research challenges

We consider open research challenges in addition to those presented in Section 7.2, towards answering RQ3. The following research challenges are a result of our own analysis and interpretation of the existing body of knowledge.

Requirements for flexible modelling in industrial settings Technical support for flexible modelling emerged as a challenge partially from the primary studies. There is a seeming contradiction in the requirements for flexible modelling technology, as it needs to provide flexible and domain-specific support while also being general-purpose enough to be used across different settings. Moreover, the tooling needs to be maintained and evolved according to specific demands of particular settings in which it is used. Out of 26 primary studies, 6 provide evidence of the need for flexible modelling from concrete use cases and only 7 show evidence of usefulness to human evaluators (Section 5.2). Without a broader set of use cases and evaluations, there is a risk that proposed solutions do not meet the requirements of practical usability and usefulness.

In our experience, industrial settings often bring about the needs for the informal side of the spectrum in Figure 3. The need for informality is commonly cited in the primary studies too, see Table 3, as well as the limitations of diagramming. There is a need to understand the settings in which flexible modelling can help and what features it shall provide to improve industrial practices. An open question is if the usage scenarios only consider these two motivations for flexibility, or if others are considered too. One concrete question is to which extent the various dimensions of flexibility are relevant in practice; examples of these dimensions are conformance, completeness, and uncertainty.

Flexibility in multi-paradigm modelling From our observations, current industrial modelling interests are in model-based systems engineering (MBSE) (Walden 2015), multi-view modelling (Cicchetti et al. 2019) and multi-paradigm modelling (Vangheluwe et al. 2002). In our experience with industrial projects and settings, using models as a central artefact and source of truth for design decisions, such as in MBSE, is much more common than the generation of code from models, as is often the target of research in the canonical modelling area. During the development of complex software systems, stakeholders from different disciplines create models (be them informal or canonical) for their own domain and using the most suitable tooling for it. Consequently, integrating knowledge from different disciplines is a difficult challenge.

In industrial practice, multi-view modelling is by default synthetic, distributed, collaborative, and asynchronous. Moreover, due to the inherent complexity of the modelled systems, models are often incomplete, inconsistent, and fraught with uncertainty. Flexible modelling can be a way to support early capturing of this information, as well as early validation and processing of the produced flexible artefacts. As such, flexible modelling may be better considered as an emergent property of integrating multiple tools providing some aspect of flexibility, so that they together support a flexible workflow.

There are multiple challenges arising from the integration of models. What is modelled, as well as the purpose of the models, can be constantly changing. That brings about a challenge in the further usage of these models, since the models created for one purpose may not be valid when considering them for an evolved purpose. Moreover, there may be unexpected emerging properties from combining multiple flexible artefacts, especially when the amount of modelled information is changing. Nevertheless, to benefit fully from modelling in these settings, it is usually useful to provide cross-model analysis.

Key takeaways To conclude this discussion, we list the following key takeaways from this study:

1. Flexible modelling emerges from both the limitations of free-form diagramming and the strictness of canonical modelling.
2. We identified a spectrum of flexible modelling from free-form sketching to canonical modelling and describe how various proposed methods cover parts of this spectrum. Further work on flexible modelling is needed to enable free movement back and forth along the spectrum.
3. Further research effort is needed to understand whether (and if yes, in what settings) flexibility support shall be provided by one single tool or a set of independent tools.
4. Further research effort is needed to understand more about industrial settings that may benefit the most from flexible

modelling, to understand industrial requirements, and to bridge the gap between industrial practices and the theoretical research on modelling approaches.

5. Further research effort is needed on approaches and applications of combining knowledge captured in multiple flexible models in multi-view or multi-paradigm settings.

8. Conclusion

In this paper, we reported on a systematic literature review on flexible modelling approaches. We have seen motivations explaining that strict conformance of modelling can be too burdensome for development, and that complete freedom of sketching lacking conformance to a language hinders automated support to query and manipulate artefacts. To meet both needs, mechanisms have been proposed in both directions, either allowing for more freedom or more strictness. Flexible modelling can be seen as a spectrum along which these practices are placed from the one extreme, complete free-form sketching, to the other, canonical modelling with full conformance to a modelling language.

The collected approaches are to a limited extent motivated by practical needs and rarely evaluated in real scenarios. From this observation, we derived open needs and challenges other than those identified by the literature, e.g. gathering industrial usage scenarios and corresponding requirements for flexible modelling approaches and integrating flexible modelling approaches in other engineering processes. It is part of our future work to identify and describe more concrete industrial settings that may benefit from flexible modelling, towards a guidance process that helps in understanding what kind of and to what extent flexible modelling is needed.

Other future works include addressing the open research challenges highlighted in the primary studies, among others related to automated guidance, collaboration, and tool support for flexible modelling. Moreover, an interesting future work direction lies in the combination of flexible artefacts from different domains, in multi-paradigm modelling, and model-based systems engineering settings.

Acknowledgments

This research was partially funded by Software Center (https://software-center.se) and the Swedish Research Council (VR) through ORPHEUS (rn. 2022-03408).

References

Baltes, S., & Diehl, S. (2014). Sketches and diagrams in practice. In *FSE* (pp. 530–541). doi: 10.1145/2635868.2635891

Bartelt, C., Vogel, M., & Warnecke, T. (2013). Collaborative creativity: from hand drawn sketches to formal domain specific models and back again. In *MoRoCo@ECSCW* (pp. 25–32).

Bider, I., Perjons, E., & Bork, D. (2021). Towards on-the-fly creation of modeling language jargons. In *ICTERI* (pp. 142–157).

Brambilla, M., Cabot, J., & Wimmer, M. (2017). *Model-driven software engineering in practice*. Morgan & Claypool Publishers. doi: 10.1007/978-3-031-02549-5

Bruneliere, H., Garcia, J., Desfray, P., Khelladi, D. E., Hebig, R., Bendraou, R., & Cabot, J. (2015). On lightweight metamodel extension to support modeling tools agility. In *ECMFA* (pp. 62–74). doi: 10.1007/978-3-319-21151-0_5

Chen, Q., Grundy, J., & Hosking, J. (2008). SUMLOW: early design-stage sketching of UML diagrams on an e-whiteboard. *Software: Practice and Experience*, *38*(9), 961–994. doi: 10.1002/spe.856

Cicchetti, A., Ciccozzi, F., & Pierantonio, A. (2019). Multi-view approaches for software and system modelling: a systematic literature review. *SoSyM*, *18*, 3207–3233. doi: 10.1007/s10270-018-00713-w

Correia, F. F., & Aguiar, A. (2013). Patterns of flexible modeling tools. In *PLoP 2013* (pp. 1–17).

Degueule, T., Combemale, B., Blouin, A., Barais, O., & Jézéquel, J.-M. (2017). Safe model polymorphism for flexible modeling. *Computer Languages, Systems & Structures*, *49*, 176–195. doi: 10.1016/j.cl.2016.09.001

Demuth, A., Lopez-Herrejon, R. E., & Egyed, A. (2011). Cross-layer modeler: a tool for flexible multilevel modeling with consistency checking. In *FSE* (pp. 452–455). doi: 10.1145/2025113.2025189

Di Ruscio, D., Pierantonio, A., & De Lara, J. (2012). Summary of the Extreme Modeling Workshop. In *XM@MODELS* (pp. 1–2). doi: 10.1145/2467307.2467308

Gogolla, M., Clarisó, R., Selic, B., & Cabot, J. (2021). Towards facilitating the exploration of informal concepts in formal modeling tools. In *MODELS (Companion)* (pp. 244–248). doi: 10.1109/MODELS-C53483.2021.00044

Gogolla, M., Selic, B., Kästner, A., Degrandow, L., & Namegni, C. (2022). From object to class models: More steps towards flexible modeling (short paper). In *FPVM at STAF* (Vol. 3250). CEUR-WS.

Golra, F. R., Beugnard, A., Dagnat, F., Guerin, S., & Guychard, C. (2016). Using free modeling as an agile method for developing domain specific modeling languages. In *MODELS* (pp. 24–34). doi: 10.1145/2976767.2976807

Guerra, E., & de Lara, J. (2018). On the quest for flexible modelling. In *MODELS* (pp. 23–33). doi: 10.1145/3239372.3239376

Hili, N. (2016). A metamodeling framework for promoting flexibility and creativity over strict model conformance. In *FlexMDE* (Vol. 1694, pp. 2–11).

Jongeling, R. (2023). *Flexible modelling - SLR datasheet.* doi: 10.5281/zenodo.10225254

Jongeling, R., & Ciccozzi, F. (2023). Towards supporting malleable architecture models. In *MDE4SA.*

Jongeling, R., Ciccozzi, F., Cicchetti, A., & Carlson, J. (2022). From informal architecture diagrams to flexible blended models. In *ECSA* (pp. 143–158). doi: 10.1007/978-3-031-16697-6_10

Kimelman, D., & Hirschman, K. (2011). A spectrum of flexibility-lowering barriers to modeling tool adoption. In *FlexiTools.*

Kimelman, D., Ossher, H., van der Hoek, A., & Storey, M.-A. (2010). SPLASH 2010 workshop on flexible modeling tools. In *OOPSLA (Companion)* (pp. 283–284). doi: 10.1145/1869542.1869614

Kitchenham, B. e. a. (2007). *Guidelines for performing systematic literature reviews in software engineering.* Technical report.

Kolovos, D. S., Matragkas, N. D., Rodríguez, H. H., & Paige, R. F. (2013). Programmatic Muddle Management. *XM@MODELS*, *1089*, 2–10.

Lara, J. D., & Guerra, E. (2017). A posteriori typing for model-driven engineering: Concepts, analysis, and applications. *TOSEM*, *25*(4), 1–60. doi: 10.1145/3063384

Lara, J. D., Guerra, E., & Kienzle, J. (2021). Facet-oriented Modelling. *TOSEM*, *30*(3), 1–59. doi: 10.1145/3428076

Lemma, R., Lanza, M., & Mocci, A. (2015). Cel: Touching software modeling in essence. In *SANER* (pp. 439–448). doi: 10.1109/SANER.2015.7081854

López-Fernández, J. J., Garmendia, A., Guerra, E., & de Lara, J. (2019). An example is worth a thousand words: Creating graphical modelling environments by example. *SoSyM*, *18*, 961–993. doi: 10.1007/s10270-017-0632-7

Nachreiner, L., Raschke, A., Stegmaier, M., & Tichy, M. (2020). CouchEdit: a relaxed conformance editing approach. In *MODELS (Companion)* (pp. 1–5). doi: 10.1145/3417990.3421401

Ossher, H., Bellamy, R., Amid, D., Anaby-Tavor, A., Callery, M., Desmond, M., … Swart, C. (2009). Business insight toolkit: Flexible pre-requirements modeling. In *ICSE (Companion)* (pp. 423–424). doi: 10.1109/ICSE-COMPANION.2009.5071043

Ossher, H., Bellamy, R., Simmonds, I., Amid, D., Anaby-Tavor, A., Callery, M., … Krasikov, S. (2010). Flexible modeling tools for pre-requirements analysis: conceptual architecture and research challenges. In *OOPSLA* (pp. 848–864). doi: 10.1145/1869459.1869529

Peltonen, J., Felin, M., & Vartiala, M. (2010). From a freeform graphics tool to a repository based modeling tool. In *ECSA (Companion)* (pp. 277–284). doi: 10.1145/1842752.1842804

Pérez-Soler, S., Guerra, E., & de Lara, J. (2019). Flexible modelling using conversational agents. In *MODELS (Companion)* (pp. 478–482). doi: 10.1109/MODELS-C.2019.00076

Reiz, A., Sandkuhl, K., Smirnov, A., & Shilov, N. (2018). Grass-root enterprise modeling: Issues and potentials of retrieving models from powerpoint. In *PoEM* (pp. 55–70). doi: 10.1007/978-3-030-02302-7_4

Robles, G., Ho-Quang, T., Hebig, R., Chaudron, M. R., & Fernandez, M. A. (2017). An extensive dataset of UML models in GitHub. In *MSR* (pp. 519–522).

Rose, L. M., Kolovos, D. S., & Paige, R. F. (2012). Eugenia live: a flexible graphical modelling tool. In *XM@MODELS* (pp. 15–20). doi: 10.1145/2467307.2467311

Sottet, J.-S., & Biri, N. (2016). JSMF: a flexible JavaScript modelling framework. In *FlexMDE*.

Störrle, H. (2017). How are conceptual models used in industrial software development? A descriptive survey. In *EASE* (pp. 160–169). doi: 10.1145/3084226.3084256

Troya, J., Moreno, N., Bertoa, M. F., & Vallecillo, A. (2021). Uncertainty representation in software models: A survey. *SoSyM*, *20*(4), 1183–1213. doi: 10.1007/s10270-020-00842-1

Vangheluwe, H., Lara, J., & Mosterman, P. (2002, 01). An introduction to multi-paradigm modelling and simulation. *AIS*.

Vesin, B., Jolak, R., & Chaudron, M. R. (2017). OctoUML: an environment for exploratory and collaborative software design. In *ICSE (Companion)* (pp. 7–10). doi: 10.1109/ICSE-C.2017.19

Walden, D. D. e. a. (2015). Systems engineering handbook: A guide for system life cycle processes and activities.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *EASE* (pp. 1–10). doi: 10.1145/2601248.2601268

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media.

Wüest, D., Seyff, N., & Glinz, M. (2019). FlexiSketch: a lightweight sketching and metamodeling approach for end-users. *SoSyM*, *18*, 1513–1541. doi: 10.1007/s10270-017-0623-8

Zarwin, Z., Bjekovic, M., Favre, J.-M., Sottet, J.-S., & Proper, H. A. (2014, July). Natural modelling. *JOT*, *13*(3), 4:1-36. Retrieved from http://www.jot.fm/contents/issue_2014_07/article4.html doi: 10.5381/jot.2014.13.3.a4

Zhang, M., Selic, B., Ali, S., Yue, T., Okariz, O., & Norgren, R. (2016). Understanding uncertainty in cyber-physical systems: A conceptual model. In *ECMFA* (Vol. 9764, pp. 247–264). Springer. doi: 10.1007/978-3-319-42061-5_16

Zolotas, A., Matragkas, N., Devlin, S., Kolovos, D. S., & Paige, R. F. (2019). Type inference in flexible model-driven engineering using classification algorithms. *SoSyM*, *18*, 345–366. doi: 10.1007/s10270-018-0658-5

About the authors

Robbert Jongeling is an associate senior lecturer in software engineering at Mälardalen University. His research includes industry-academia collaborations in which the aim is to improve support for flexible and continuous model-based software and systems engineering. You can contact the author at robbert.jongeling@mdu.se or visit https://robbert.science.

Federico Ciccozzi is an associate professor in software engineering at Mälardalen University, department of Innovation, Design and Engineering. He leads the Automated Software language and Software engineering research group and is the Head of Research Education in Computer Science and in Engineering. You can contact the author at federico.ciccozzi@mdu.se.