Noname manuscript No.
(will be inserted by the editor)

Large Language Models in Model-Driven Engineering: A Systematic Mapping Study

Weixing Zhang · Bowen Jiang* · Yuhong Fu* · Haowei Cheng · Maximilian Hummel · Vincenzo Scotti · Nathan Hagel · Jialong Li · Georg Grossmann · Markus Stumptner · Regina Hebig · Daniel Strüber · Anne Koziolek

Received: date / Accepted: date
Clinical trial number: not applicable

Abstract The application of Large Language Models (LLMs) in Model-Driven Engineering (MDE) has emerged as a rapidly evolving research area. While existing systematic literature reviews have examined specific technical approaches, a comprehensive mapping of the broader research landscape (e.g., development trends) remains lacking. This study presents a systematic mapping study of LLM applications in MDE, analyzing 86 primary studies collected from five databases, covering publications from 2022 to early 2026. Guided by five research questions, we characterize the field across five dimensions: MDE task distribution and research contribution types, LLM technologies and interaction strategies, artifact representation and processing, validation practices, and publication landscape. Our findings reveal that current LLM4MDE research is heavily concentrated on Model Generation, while tasks such as Model Migration, DSL Engineering, and Metamodeling remain marginal. Most approaches rely on black-box OpenAI models accessed via remote APIs and adapted through prompt engineering, with fine-tuning and retrieval-augmented generation rarely employed. Inputs are predominantly natural-language arti-

Weixing Zhang
Karlsruhe Institute of Technology, Karlsruhe, DE, E-mail: weixing.zhang@kit.edu

Bowen Jiang*
Karlsruhe Institute of Technology, Karlsruhe, DE, E-mail: bowen.jiang@kit.edu

Yuhong Fu*
Adelaide University, Adelaide, AU, E-mail: yuhong.fu@adelaide.edu.au

Haowei Cheng
Waseda University, Tokyo, JP, E-mail: haowei.cheng@fuji.waseda.jp

Maximilian Hummel
Karlsruhe Institute of Technology, Karlsruhe, DE, E-mail: maximilian.hummel@kit.edu

Vincenzo Scotti
Karlsruhe Institute of Technology, Karlsruhe, DE, E-mail: vincenzo.scotti@kit.edu

Nathan Hagel
Karlsruhe Institute of Technology, Karlsruhe, DE, E-mail: nathan.hagel@kit.edu

Jialong Li
Waseda University, Tokyo, JP, E-mail: lijialong@fuji.waseda.jp

Georg Grossmann
Adelaide University, Adelaide, AU, E-mail: georg.grossmann@adelaide.edu.au

Markus Stumptner
Adelaide University, Adelaide, AU, E-mail: markus.stumptner@adelaide.edu.au

Regina Hebig
University of Rostock, Rostock, DE, E-mail: regina.hebig@uni-rostock.de

Daniel Strüber
Chalmers | University of Gothenburg, Gothenburg, SE & Radboud University, Nijmegen, NL, E-mail: danstru@chalmers.se

Anne Koziolek
Karlsruhe Institute of Technology, Karlsruhe, DE, E-mail: koziolek@kit.edu

[1] Equal contribution.

facts, while outputs are model-oriented but usually expressed in lightweight textual formats rather than native MDE exchange formats. Validation is centered on quantitative experimentation, with 42% of studies reporting no baseline and cost efficiency reported in fewer than one quarter of studies. The field has grown rapidly, from one paper in 2022 to 42 in 2025, with research concentrated in Europe and Canada and limited industry involvement. Based on these findings, we identify gaps and opportunities across task coverage, technical configuration, and evaluation practice, offering a knowledge map to guide future work in this cross-disciplinary field.

Keywords Large Language Models, Model-Driven Engineering, Systematic Mapping Study, Model Generation, Domain-Specific Languages, Prompt Engineering

1 Introduction

Large Language Models (LLMs) have become a technological game changer of the 21st century, to the point where they are the dominant technology associated with the term *artificial intelligence* [35][51]. From the introduction of the Transformer architecture in 2017 to the widespread application of models like GPT and BERT today, LLMs have moved from academic research to the core of industrial practice [81][122]. Today, LLMs can be defined as neural network models based on deep learning architectures that obtain strong language understanding and generation capabilities through pre-training on large-scale text data [27]. These models show unprecedented abilities in code generation, document understanding, requirement analysis, and system design [51][18].

The application of LLM in Software Engineering (SE) shows diverse forms and is becoming part of development practices. Automatic code generation tools help programmers improve efficiency [17] [111]. Smart testing tools assist with quality assurance [90]. Documentation generation systems support project maintenance [31].

Although LLMs show great potential in SE, their application in the specific field of model-driven engineering (MDE) is still in the exploration stage. MDE is an important branch of SE. It emphasizes model-centered development methods [95]. It improves software development efficiency and quality through model transformation, code generation, and automated analysis. However, traditional MDE methods face challenges such as high complexity of modeling languages, high barriers for domain expert participation, and high model maintenance costs [105][71].

Successful implementation of non-trivial MDE systems requires collaboration from experts in different domains [95][54]. These include software architects, domain experts, system analysts, and developers. In traditional MDE practice, these experts need to master specific modeling languages and tools such as UML, DSL, and metamodels. For example, in system modeling, architects need to use UML class diagrams and component diagrams [95]. In domain modeling, experts need to define specific domain-specific languages (DSL) [38]. In code generation, developers need to write complex model transformation rules.

Due to these complexities, researchers and industry have begun to explore the potential of LLMs in MDE. Due to their inherent capability to process natural language prompts and produce high-fidelity results, LLMs give rise to an extremely wide spectrum of application scenarios in MDE, including simplifications of the modeling process [9], automation of model transformation [70], intelligent code generation [62], and more intuitive human-computer interaction. These applications cover multiple aspects from requirements to model transformation, inter-model transformation, model to code generation, and automatic evaluation of model quality [3].

Although recent research has provided systematic literature reviews of technical methods for LLMs in MDE [29], a comprehensive mapping analysis is still lacking from a research landscape perspective. The development status, research distribution characteristics, technical trends, and research ecosystem of this field remain unclear. Specifically, it is still unclear how the research scale and growth trends in this field develop, the distribution of research activity levels across different types of MDE tasks, the characteristics of major academic publication venues, the participation patterns of active research institutions and teams, the usage distribution of various LLM technical methods, how MDE artifacts are represented and processed as inputs and outputs in LLM-based approaches, the validation strategies of existing methods, and the reproducibility support status of open-source tools and datasets.

To better understand these aspects, this paper provides an SE research map of LLM applications in MDE (LLM4MDE in short). This map can guide researchers and practitioners from artificial intel-

ligence and SE fields to understand the current research status and help reveal future challenges and opportunities.

2 Background

2.1 Model-Driven Engineering Fundamentals

Model-driven engineering (MDE, [95]) is a long-established research field of 25 years that envisions the use of models as primary development artifacts in software engineering. A model is an instance of a *modeling language*, traditionally a general-purpose modeling language such as UML, while the field shifted over time to dedicated domain-specific languages (DSLs) tailored to particular application domain. The roots of MDE are in *Model-Driven Architecture* (MDA), which envisioned a particular process of generative software development that moves from more abstract to more concrete models through a chain of transformations.

The value of automated transformations was recognized early in the field and led to the emergence of a variety of *model transformation* approaches, usually either focused on model-to-model transformations or code generation approaches. A wide set of different model transformation paradigms was developed, including declarative, imperative, reactive, graph-based languages and led to a sizable taxonomy, as captured in Czarnecki's and Helsen's seminal survey [26].

An additional asset of the use of models is the capability to perform analyses on the model before the system is fully developed, which can avoid costly errors and repair further downstream in the process. *Model validation* [37] is concerned with such analyses, which range from light-weight syntactic checks (e.g., identifying violations of multiplicity constraints) over testing to more involved semantic analysis based on formal representations (e.g., model checking).

The rise of DSLs for modeling also led to comprehensive activities on design principles and tools for *language engineering* in MDE [38]. Developing the core of a modeling language traditionally involves expressing the abstract syntax of the language as a *meta-model*, that is, a model for other models. A model language can have one or several concrete syntaxes (e.g., textual, graphical, tabular), which can be defined using available language workbenches. Language semantics can be defined through formal languages or pragmatically, e.g., through code generation. When a modeling language changes, e.g., by editing the meta-model, associated artifacts such as available models may need to be *migrated*, which has led to a sizable amount of work on model migration or co-evolution approaches [87].

In conventional MDE processes, *model generation* or creating the model is usually a manual activity. A relatively recent development of the field are approaches that assist the developer as they edit the model [67]. *Model repair* approaches can guide developers in fixing errors in the model, including, e.g., the repair of broken constraints. As a pioneering application area of AI approaches in MDE, *model completion* approaches can support a developer in identifying and missing elements, based on the current state of the model. A separate strand are automated *reverse engineering* approaches, which derive models from existing systems.

2.2 Large Language Models Overview

Large Language Models (LLMs) are probabilistic generative models of text based on Deep Neural Networks (DNNs) trained on massive amounts of data collected from diverse sources such as the web, books, and code repositories. LLMs are built using the Transformer architecture as backbone [106]: a DNN for sequence processing that can ingest discrete sequences of symbols (i.e., tokens that can represent individual characters, words, or sub-word units). The Transformer leverages the self-attention mechanism to capture complex long-range dependencies and allow the LLM to produce coherent and contextually meaningful text. The term "large" is not strictly defined (it generally reflects the rapid increase in model size over time), but foundational examples include *BERT* [27], *GPT* [81], *T5* [82], and their successors. Nowadays, LLMs are well known for their flexibility and capabilities, extending beyond text to support multiple input and output modalities (*Multimodal* models) such as images, audio, and video.

LLMs are typically developed in two stages. First, a *pre-training* phase exposes the model to large-scale corpora, enabling it to learn statistical patterns of language and world knowledge. After pre-training,

models are often *fine-tuned* to follow instructions or behave as conversational agents. Instruction-following fine-tuning adapts the model to respond to natural language prompts for specific tasks [21, 91], while chatbot-assistant fine-tuning further specializes it to interactive multi-turn dialogues [97]. These fine-tuning stages may rely on supervised examples or Reinforcement Learning from Human Feedback (RLHF). Besides assistant-style fine-tuning, other variants target specialized domains such as code generation or logical reasoning [45, 49, 88].

The majority of currently adopted LLMs adopt a *causal* (or *autoregressive*) architecture. These models generate text one token at a time, predicting the next token based on the entire previously observed context. Users interact with these models via *prompting*: they supply a prefix consisting of an instruction or query, and the LLM produces a completion representing its response. As tokens are generated, they are appended to the context and used to condition subsequent predictions, making the prompting process central to LLM-based applications.

Several prompting strategies were developed to improve LLM performance. *Few-shot learning* (or in-context learning) [11] provides task descriptions together with input–output examples to guide the model even on unseen tasks without further training (DNN weights update). Retrieval Augmented Generation (RAG) [43] mitigates the model's limited internal knowledge and static training cutoff by injecting external retrieved information directly into the prompt. Chain-of-Thought (CoT) reasoning [60] encourages the model to generate intermediate reasoning steps before producing the final answer, improving performance on multi-step or logically complex tasks. These prompting techniques are orthogonal and can be combined to further enhance reliability and task accuracy.

LLMs are commonly made accessible through fine-tuned assistant versions deployed as chatbots or agentic systems. In these settings, the model is provided with a system instruction describing its role and objectives, and may be equipped with tools such as search or code execution interfaces to act as an autonomous agent. Available models range from open-access, research-friendly models such as *Llama* [47], *Qwen* [114], *Mistral* [56], and *Gemma* [69], to large closed-access commercial models such as *GPT* [76] and *Gemini* [24]. The type of access significantly influences the level of controllability, transparency, and adaptability available to practitioners.

3 Study Design

We designed and conducted this study following established guidelines for systematic mapping studies [79, 80]. Based on these guidelines, this section describes the core aspects of our mapping study design. Specifically, we outline the research goal and driving questions and describe the steps of our research methodology. A complete replication package is publicly available in [112], enabling independent replication and validation of our study. The replication package contains the raw data from our search and screening phase, a list of selected primary studies, raw data extracted from each primary study, and the R scripts we developed for data exploration and analysis.

3.1 Research goal and questions

The primary goal of this systematic mapping study is to provide a comprehensive overview of the research landscape regarding the application of LLMs in MDE. Unlike existing systematic literature reviews that focus on technical approaches and methods [29], we aim to map the broader research ecosystem, i.e., understanding the current state, trends, and characteristics of LLM4MDE research.

This mapping study reveals what research topics and MDE tasks are being explored, what technical approaches and LLM technologies researchers are using, and how they validate their LLM-based approaches. We also examine the scale and distribution of research activity, including publication venues and institutional participation. By mapping these aspects, we can help researchers and practitioners understand where LLM4MDE currently stands and identify promising directions for future work.

To achieve this goal, we formulated the following research questions:

RQ1: What are the main research topics, application focuses, and research contribution types in LLM4MDE research? By answering RQ1, we can identify which MDE tasks researchers are focusing on when applying LLMs—such as model generation, transformation, or validation—and understand what application domains are being explored. This helps us see where research attention

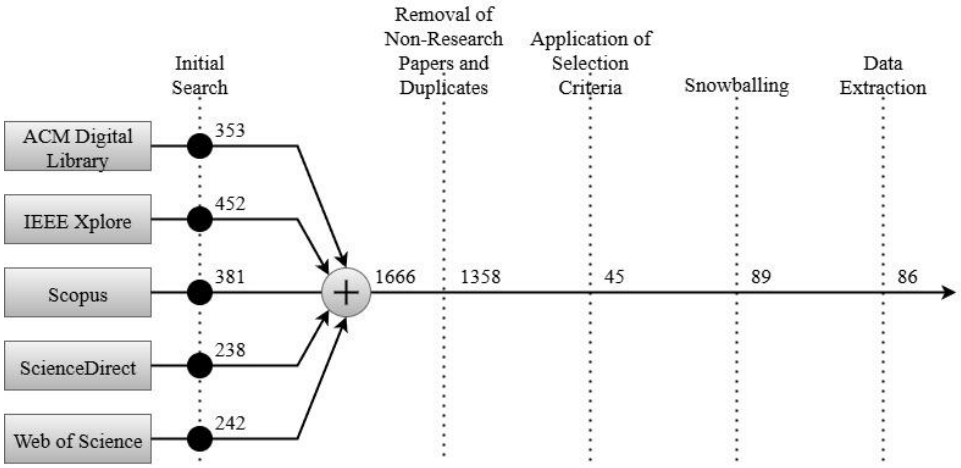

Fig. 1: Our search and selection process comprises five phases to search, identify, and process relevant publications.

is concentrated and which areas might be underexplored. The findings provide a foundation for understanding the scope and priorities of current work in LLM4MDE.

RQ2: What LLM technologies, interaction strategies, and system designs are employed in LLM4MDE research? By answering RQ2, we can see what specific LLM models researchers are using, how they access and configure these models, and what prompt engineering and fine-tuning strategies they apply. Understanding these technical choices reveals the current state of LLM adoption in the MDE community, and helps identify which technologies and configurations are gaining traction as well as which remain underexplored.

RQ3: How are artifacts represented and processed as inputs and outputs in LLM4MDE approaches? By answering RQ3, we can understand how researchers represent MDE models in formats suitable for LLM consumption, and what types of artifacts are used as inputs and outputs. This helps us identify the common practices and technical conventions for bridging MDE artifacts and LLM-based processing, and reveals which representation strategies are most prevalent in the community.

RQ4: How are researchers validating existing LLM-based approaches for MDE? By answering RQ4, we can see what aspects of LLM-based MDE approaches researchers actually evaluate and how they do it. We then compare these practices against existing LLM evaluation guidelines in SE to identify where gaps exist. Understanding these gaps is important for developing more comprehensive evaluation approaches in this emerging area. We also examine whether studies report the computational or financial cost of their approach, as token consumption and API costs are relevant to the practical viability of LLM-based solutions [8].

RQ5: What are the current publication scale, distribution patterns, and growth trends of LLM4MDE research? By answering RQ5, we can understand how the field has grown over time and where research is happening. This includes looking at publication trends to see if the field is expanding or stabilizing, identifying which conferences and journals are the main venues for LLM4MDE work, and mapping which institutions and geographic regions are most active in this area. These insights help us understand the maturity and global reach of LLM4MDE research.

3.2 Search and Selection Process

Fig. 1 illustrates the main phases of the search and selection process of this study, as well as the number of potentially relevant studies involved in each phase.

3.2.1 Initial Search

This study uses a multi-database search strategy to ensure complete literature coverage. Based on the recommendations by Kitchenham et al. in [58] regarding search strategies for systematic reviews in SE,

we first selected four databases: IEEE Xplore Digital Library, ACM Digital Library, SCOPUS, and Web of Science. IEEE Xplore and ACM Digital Library serve as the most important professional databases in SE. They can "ensure good coverage of important journals and conferences." SCOPUS and Web of Science work as general indexing systems. They provide broader academic literature coverage and meet the recommendation to use "at least two general indexing systems." Then we also include ScienceDirect to further expand the search scope and reduce the risk of missing relevant literature.

Listing 1: The search string.

```
1 ( system modeling OR system architect OR model driven engineering OR model-driven
    engineering OR MDE OR model driven development OR model-driven development OR MDD
    OR domain specific language* OR DSL*)
2 AND
3 ( large language model* OR LLM* OR generative AI OR pre-trained language model* OR
    foundation model* )
```

The search string used to query the above data sources is shown in Listing 1. In order to focus our attention on the main goal of the study, the search string was created based on our goal and research questions. Specifically, our search string consists of two main parts, which are linked by a logical *AND* relationship: the first part (line 1 in the list) targets the discipline of model-driven engineering, and the second part (line 3) targets the discipline of large language models. To have a high degree of confidence in the coverage of the search string, we (1) included keywords related to their corresponding aspects in both parts based on preliminary research and known preliminary research, (2) added synonyms for the added terms, (3) added alternative spellings and acronyms for the added terms, and finally (4) linked all the added terms by a logical *OR* relationship.

We applied the search string consistently across all databases, targeting titles, abstracts, and keywords. While the search logic remained uniform, we adapted the query format to meet the specific requirements of individual databases (refer to the replication package for more details). ACM Digital Library and ScienceDirect offer built-in article type filters, which we used to exclude non-research papers (e.g., editorials, prefaces). The search results from these two databases represent post-filtering counts. The search was conducted in September 2025, covering publications from January 2020 onwards to focus on the recent rapid development of LLM technologies. To ensure an unbiased selection process, we employed a rigorous approach. Two different authors independently evaluated all the papers, and the three senior co-authors thoroughly reviewed the entire selection process. The initial search yielded 353 papers from ACM Digital Library, 452 from IEEE Xplore, 381 from Scopus, 238 from ScienceDirect, and 242 from Web of Science, totaling 1666 potentially relevant studies.

3.2.2 Removal of Non-Research Papers and Duplicates

Due to the nature of the electronic databases and indexing systems, search results from IEEE Xplore, Scopus, and Web of Science included items that were not research papers, such as editorials, prefaces, keynote abstracts, and workshop summaries. While ACM Digital Library and ScienceDirect had already been filtered using their built-in article type functionality, the other three databases required manual screening. Consequently, in this phase, we manually remove 235 duplicates as well as 73 non-research results to obtain a coherent primary study of 1358 potentially relevant research studies. Eventually, the remaining papers from the various databases are combined into a single dataset.

3.2.3 Application of Selection Criteria

In this phase, we consider all the potentially relevant studies and filter them according to a set of well-defined selection criteria. The used selection criteria are reported below.

Inclusion Criteria

I1: Studies that apply, integrate, or evaluate LLMs for MDE tasks or activities.
I2: Studies are written in English.
I3: Studies including empirical evaluation, validation, or assessment of the proposed approach.
I4: Peer-reviewed studies published in journals, conferences, or workshops.

I5: Studies where LLMs play a substantial role in an MDE process.

Exclusion Criteria

E1: Secondary literature studies.
E2: Studies for which full text is not available.
E3: Studies where MDE is being used merely as an application example or LLMs are being used only for peripheral tasks.

Screening Process

We adopted an adaptive reading depth strategy to ensure efficiency and objectivity of the screening process [79]. For each study, we first examined the title. If the title clearly indicated that the study was unrelated to LLM or MDE, we excluded it directly. If the title alone was insufficient for judgment, we further read the abstract, introduction, and conclusion sections. For studies that still could not be determined for inclusion, we read the full text before making a decision.

To reduce potential bias, we randomly assigned the 1358 studies to seven researchers for review. Each study was screened independently by one researcher and verified by another researcher. When the two researchers disagreed, a third researcher was involved to discuss and reach consensus.

Following this screening process, we identified 45 studies from the 1358 potentially relevant studies that satisfied all inclusion criteria and did not meet any exclusion criteria.

3.3 Snowballing

To reduce threats to construct validity and cover as many relevant studies as possible, we adopted a snowballing approach to supplement the automatic search described above [113]. Using the 45 studies identified in the previous phase as the start set, we conducted backward snowballing and forward snowballing. Backward snowballing examined the reference list of each seed study to identify potentially relevant research; forward snowballing examined papers citing each seed study through Google Scholar. Since some seed studies had a large number of upstream and downstream citations, we adopted a preliminary screening strategy during the collection phase. For each citation, if the title clearly indicated that the study was unrelated to LLM or MDE, we excluded it directly; if the title appeared relevant or could not be determined by title alone, we recorded it in a dedicated spreadsheet for further evaluation. This strategy is consistent with Wohlin's suggestion of conducting preliminary screening based on titles.

After candidate studies were collected, we first checked whether these studies duplicated any of the previous 1358 studies and confirmed whether they were research papers (excluding editorials, prefaces, and other non-research content). Subsequently, we applied the same inclusion and exclusion criteria as in Section 3.2.3, using the same execution approach: one researcher screened independently and another researcher verified. When the two researchers disagreed, a third researcher was involved to discuss and reach consensus. Given that LLM4MDE is a rapidly evolving area, forward snowballing retrieved a number of recent studies that had not yet appeared in peer-reviewed venues at the time of our search and were therefore excluded under criterion I4 (e.g., [121]). As most studies in this field were published after 2023, their citation relationships are still sparse. After conducting one round of snowballing on the start set, the newly identified studies primarily cited research that had already been covered, and the marginal benefit of additional iterations was limited. Therefore, the snowballing process terminated after one round.

Through the snowballing phase, we added 44 new studies, totaling 89 primary studies together with the 45 studies identified previously. However, among these 89 primary studies, some were journal extensions of preliminary studies already included. Therefore, we merged these 89 studies into 86 unique primary studies for subsequent analysis of research content.

3.4 Data Extraction

To collect data from the 86 primary studies, we designed a data extraction form comprising 43 fields organized into five facets, each addressing one research question. The complete list of fields, together with their dimensions and corresponding research questions, is summarized in Table 1. Detailed definitions and example values for all fields are provided in the replication package [112].

Table 1: Classification framework used in this study

RQ	Dimension	Fields
RQ1	Research Context	MDE Task Type, Specific Industry Domain, Modeling Language, Modeling Language Classification, Research Contribution Type
RQ2	LLM Configuration	LLM Model Used, LLM Access Categories, LLM Access Level
	LLM Interaction Strategy	Prompting Strategy, LLM Sampling Strategy, RAG Strategy, Fine Tuning Applied
	System Architecture	Autonomy Level, Execution Structure, Integration With MDE Tools
	Human Involvement	Human Involvement
RQ3	Input	Input Artifact Type, Input Artifact Categories, Input Data Type, Input Model Representation Format
	Output	Output Artifact Type, Output Artifact Categories, Output Data Type, Output Model Representation Format
RQ4	Evaluation Design	Evaluation Type, Evaluation Metrics, Baseline Comparison, Dataset Type, Dataset Size, Human Evaluation Included
	Rigor & Validity	Reproducibility Support, Cost Efficiency Evaluated, Threat To Validity Discussed, LLM Limitations Reported, Approach Limitations Reported
RQ5	Publication Metadata	Publication Year, Publication Venue, Venue Type, Venue Rank Or Tier
	Authorship & Collaboration	Author Affiliations, Author Countries, Collaboration Type, Funding Acknowledged

The data extraction form was developed by the research team through a collaborative process grounded in the research questions. For each research question, the relevant dimensions and fields were identified through team discussion, drawing on the authors' prior knowledge of both LLM and MDE research. The initial form was subsequently refined as primary studies were analyzed: new field values were introduced where existing categories proved insufficient, overlapping categories were merged, and ambiguous definitions were clarified. Whenever the form was revised, studies analyzed in earlier iterations were re-examined to ensure that all classifications reflected the updated definitions. This iterative approach to form development is consistent with established practices in systematic mapping studies [79].

Data extraction was conducted collaboratively by the research team. Each primary study was assigned to one team member, who read the paper in full and populated all fields of the extraction form. Each completed extraction was then independently reviewed by a second team member [80]. Disagreements identified during this review were resolved through discussion between the two team members; cases that remained unresolved were escalated to a senior author for arbitration. This practice of cross-checking extractions among multiple researchers is consistent with recommended strategies for data extraction in systematic mapping studies [80].

It is worth noting that some fields admit multiple values for a single study [80]. For example, a paper may apply more than one prompting technique, or target more than one MDE task type. In such cases, all applicable values were recorded. Where a field was not discussed or could not be determined from the paper, it was recorded as "not specified" and included in the analysis as a distinct value, as the frequency of non-reporting is itself an informative finding reflecting reporting practices in the field.

3.5 Data Synthesis

To synthesize the data extracted from the 86 primary studies, we applied frequency counting and co-occurrence analysis as the primary methods, following established practices in systematic mapping studies [79],[80]. Frequency counts were used to characterize the distribution of individual field values across the primary studies, while co-occurrence matrices were used to examine how values across related fields appear together within the same study. Where applicable, we also computed phi coefficients to quantify pairwise associations between binary validation practice dimensions (Section 4.4).

The classification schemes underlying our extraction fields vary in their source and degree of formalization. Several fields adopt existing taxonomies from the literature: *Research Contribution Type* follows the framework proposed by [99], extended with an additional category *Benchmark / Dataset* to account for contribution types prevalent in LLM research; *Input Data Type* and *Output Data Type* are classified

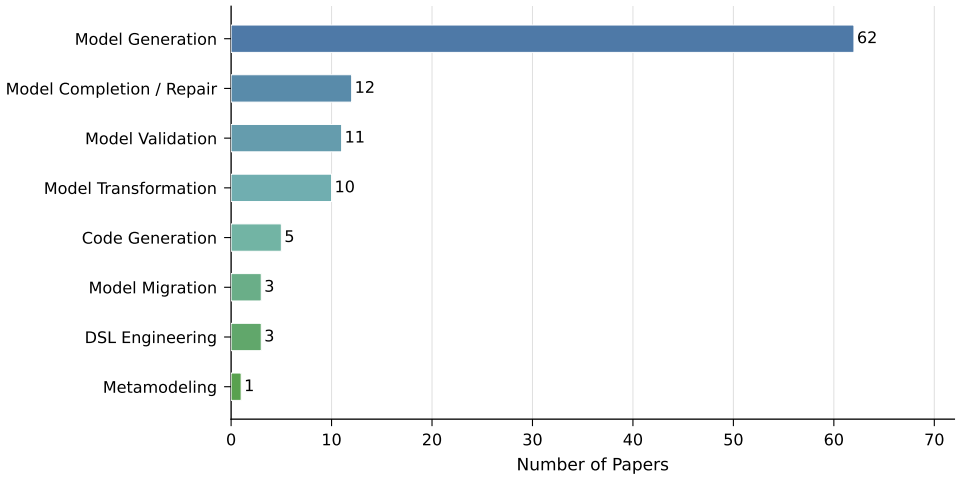

Fig. 2: Distribution of MDE task types across the selected papers. (N=86, labels may overlap)

according to ISO/TR 8344 [55]; and *Evaluation Type* follows the categories defined by [59]. The classification of *MDE Task Type* is grounded in established MDE literature and is introduced in Section 2.1. For fields where no established classification existed, we developed our own category schemes through iterative team discussion during the pilot extraction phase. These fields include *Modeling Language Classification*, *LLM Access Categories*, *LLM Access Level*, *Execution Structure*, *Human Involvement*, *Input Artifact Categories*, *Output Artifact Categories*, *Dataset Type*, and *Reproducibility Support*. For each of these fields, detailed definitions and example values are provided in the replication package [112].

Several of the above fields require the extractor to exercise judgment in selecting one or more values based on the content of the paper, rather than mechanically mapping reported information to a category. To support consistent judgment across extractors, we established cross-field consistency rules during the extraction process. For instance, if the MDE task performed by an LLM in a paper is classified as *Model Generation*, then the output of at least one LLM process in that paper must be a model (whether expressed as a serialized representation or a textual description). Rules of this kind were documented and applied throughout the extraction process to ensure that classifications across related fields remain mutually coherent.

4 Results

4.1 Research Topics and Application Focuses (RQ1)

To address RQ1, we analyze the research topics and application focuses of LLM4MDE studies across three dimensions: the MDE tasks that LLMs are applied to, whether the studies target a specific industry domain, and the modeling languages involved. We also examine the types of research contributions made by the primary studies.

As shown in Fig. 2, *Model Generation* is the most frequently addressed task, appearing in 62 of the 86 primary studies, while all other tasks each account for fewer than 15 papers. This indicates that current LLM4MDE research is heavily focused on generation, leaving tasks such as *Model Migration* (three), *DSL Engineering* (three), and *Metamodeling* (one) unexplored. As shown in Fig. 3, 76.7% of the papers address a *single task*, while 23.3% address *multiple tasks* in combination. The co-occurrence matrix in Fig. 4 reveals that the most frequent task pairings involve *Model Generation* combined with *Model Validation* (seven papers) and with *Model Transformation* (five papers), suggesting that researchers often couple generation with quality assurance or transformation processes. Other pairings are sparse, indicating that cross-task research remains limited.

Beyond task distribution, we also examine the application context of these studies. Fig. 5 shows that 64% of the primary studies (n=55) are not tied to a *specific industry domain*, while 36% (n=31) are applied to or motivated by an industrial context. This suggests that the majority of LLM4MDE

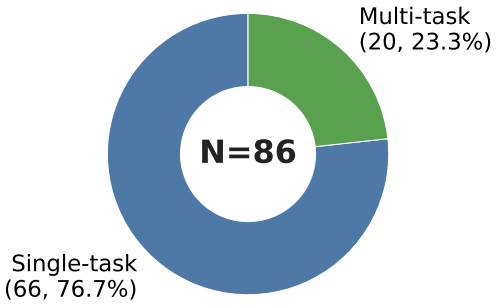

Fig. 3: Proportion of single-task versus multi-task papers.

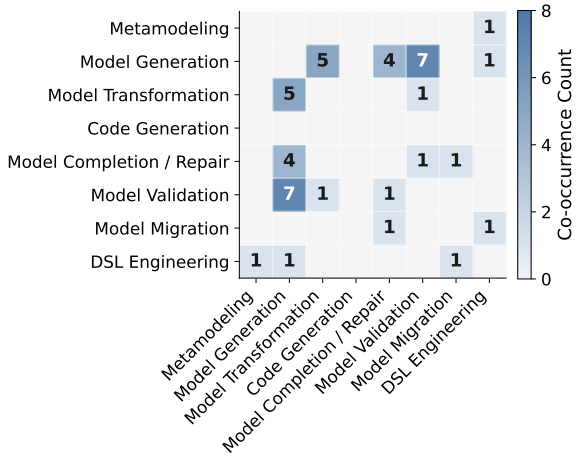

Fig. 4: Co-occurrence matrix of MDE task types.

research targets SE or MDE tasks in general, without grounding in a particular industry setting, while industry-specific applications remain a minority. The former category includes, for example, studies that assess LLM capabilities for generating UML class diagrams and assisting software modelers without targeting any specific sector [15]. Fig. 6 further breaks down industry applicability by task type. In terms of industry share within each task type, *Model Completion/Repair* shows the highest proportion (6/12, 50.0%), followed by *Code Generation* (2/5, 40.0%), *Model Generation* (23/62, 37%), and *Model Validation* (4/11, 36%).

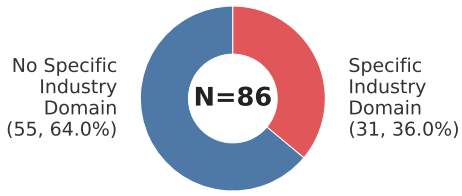

Fig. 5: Distribution of the selected papers by specific industry domain applicability.

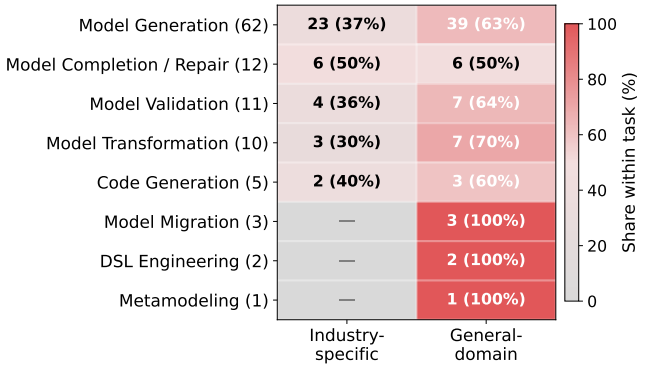

Fig. 6: Industry-domain applicability by MDE task type. Cell values report counts and within-task shares.

Regarding the modeling languages involved, Fig. 7 shows that *Standard General-Purpose Modeling Languages (GPML)* appear in 49 papers (57%), followed by *Domain-Specific Modeling Languages (DSML)* in 38 papers (44%) and *Metamodeling and Modeling Framework Languages (MFL)* in ten papers (12%). The dominance of GPML reflects a tendency to apply LLMs to well-established languages such as UML and SysML [73], for which large training corpora exist.

This distribution also varies across task types. As illustrated in Fig. 9, Model Generation relies primarily on GPMLs (35/62, 56.5%) and secondarily on DSMLs (27/62, 43.5%), with MFLs remaining marginal (6/62, 9.7%), whereas Model Validation is even more strongly concentrated on GPMLs (9/11, 81.8%).

The co-occurrence matrix in Fig. 8 shows that *GPML* and *DSML* co-occur in six papers, as do *GPML* and *MFL* in five papers, while *DSML* and *MFL* co-occur only twice, suggesting that work on metamodeling infrastructure and work on DSLs are treated as separate concerns in current research.

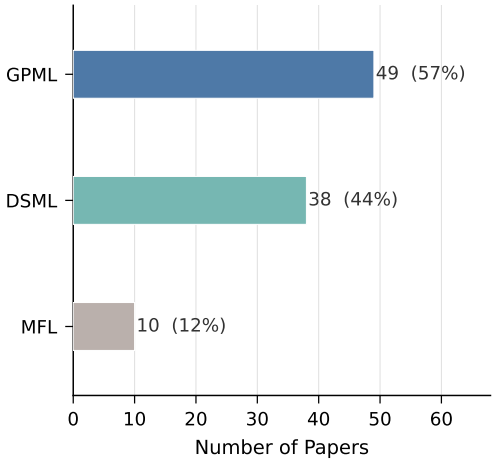

Fig. 7: Distribution of modeling language categories across the 86 primary studies (multiple categories allowed per paper).

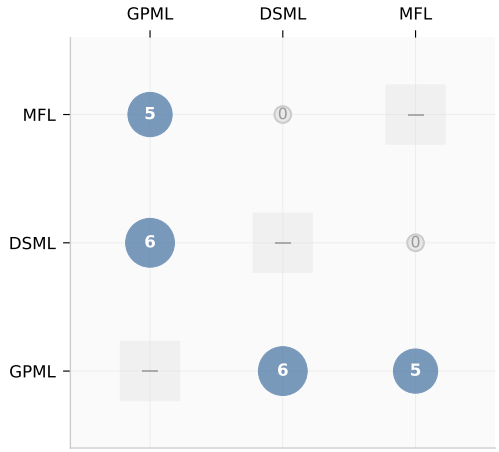

Fig. 8: Co-occurrence of modeling language categories across 86 selected primary studies.

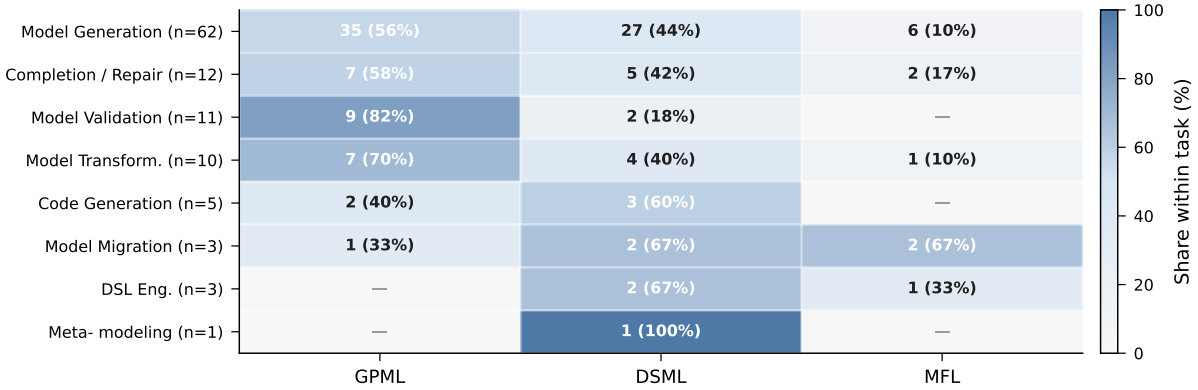

Fig. 9: Modeling language categories by MDE task type across 86 selected primary studies. Cell values report counts and within-task shares.

The low share of *MFL* is consistent with the finding in Fig. 2 that *Metamodeling* accounts for only one paper, confirming that LLM applications at the modeling language infrastructure level are limited to a small fraction of the primary studies.

It is also worth noting that *Procedure/Technique* is the dominant contribution type (53 papers, 62%), with 32 of these papers combining it with one or more additional contribution types, as shown in Fig. 10. *Tool/Prototype* is the second most frequent type (27 papers), followed by *Qualitative/Descriptive Model* (20 papers), *Benchmark/Dataset* (14 papers), *Report/Observation* (14 papers), and *Specific Solution/-Case Study Result* (ten papers). The prevalence of *Procedure/Technique* contributions, frequently accompanied by tool implementations, indicates that most LLM4MDE studies propose a concrete LLM-based method or technique alongside a tool prototype, such as [101]. Notably, *Empirical Model* and *Analytic Model* are absent from all 86 studies, suggesting that formal theoretical contributions have yet to emerge in this field.

These contribution types are not uniformly distributed across application domains (Fig. 11). Studies targeting a specific industry domain are more likely to contribute *Tool/Prototype* and *Qualitative/Descriptive Model* artifacts (39% and 42% of industry-domain studies, respectively), whereas *General-domain* studies more frequently contribute *Benchmark/Dataset* artifacts (22% vs. 6% in *Industry-specific* work). A similar contrast appears across task types. As shown in Fig. 12, *Model Transformation* papers are more likely to include a *Benchmark/Dataset* contribution (50.0%) than *Model Generation* papers (9%), while *Model Generation* remains more centered on *Procedure/Technique* contributions.

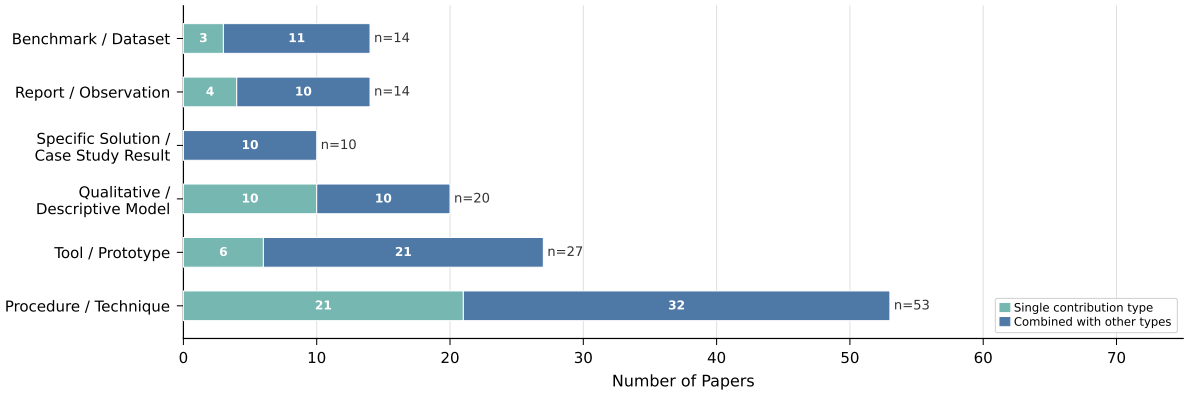

Fig. 10: Distribution of research contribution types across the 86 primary studies, classified following [99] and extended with *Benchmark / Dataset*. Each bar is divided into papers contributing solely that type (light blue) and papers combining it with one or more additional types (dark blue).

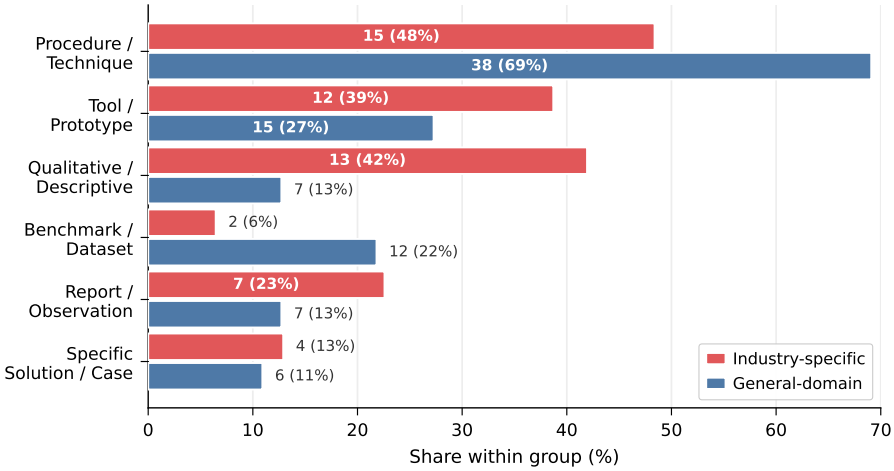

Fig. 11: Research contribution types by application focus across 86 primary studies. Labels report counts and within-group shares for industry-specific and general-domain studies.

Answer to RQ1 LLM4MDE research is heavily concentrated on *Model Generation* (62 of 86 papers), while *Model Migration* (three), *DSL Engineering* (three), and *Metamodeling* (one) remain marginal. Most studies address a *single MDE task* (76.7%), and *multi-task* combinations are sparse; where they occur, *Model Generation* paired with *Model Validation* (seven papers) and with *Model Transformation* (five papers) are the most frequent. The majority of studies (64%) are not grounded in a specific industry domain, though *Model Completion/Repair* shows the highest industry share among single-task types (50%). Regarding modeling languages, *GPMLs* appear most frequently (57%), followed by *DSMLs* (44%) and *MFLs* (12%); *DSML* and *MFL* occur twice only, suggesting that work on DSLs and on metamodeling infrastructure are treated as separate concerns. In terms of research contribution, *Procedure/Technique* is the dominant type (53 papers, 62%), while *Empirical Model* and *Analytic Model* are absent from the entire primary studies.

4.2 LLM Technologies, Interaction Strategies, and System Designs (RQ2)

In answering RQ2, we analyze the technological profile of the LLM4MDE literature along eleven dimensions: LLM models used, LLM access categories, LLM access level, Prompting Strategy, LLM sampling strategy, retrieval-augmented generation (RAG) strategy, fine-tuning applied, autonomy level, execution

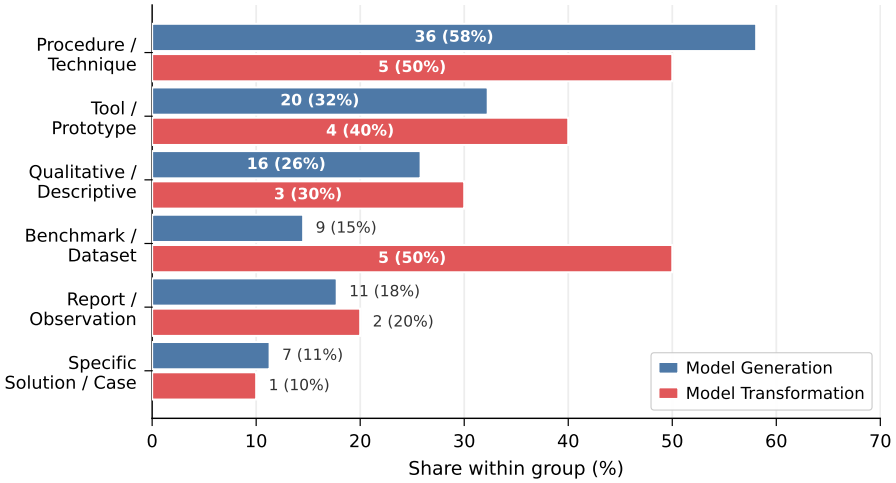

Fig. 12: Research contribution types in the MDE task groups *Model Generation* and *Model Transformation*. Bar lengths report within-group shares, and labels report counts and shares.

structure, integration with MDE tools, and human involvement in the LLM processes. These dimensions reveal how LLMs are selected, accessed, and orchestrated in current research practice. As some dimensions admit multiple values per study, category totals can exceed 86.

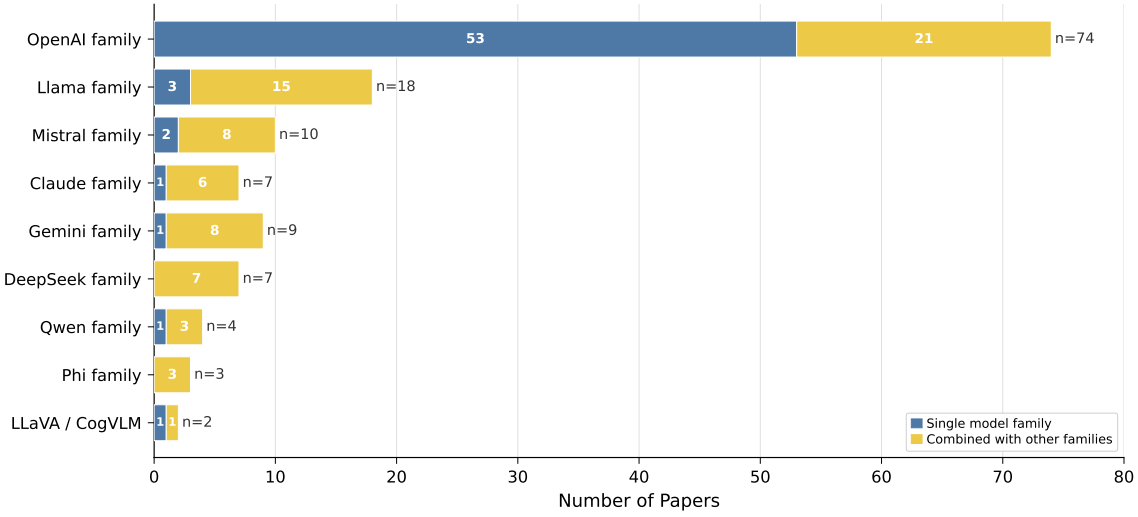

Fig. 13: Distribution of LLM model families used across 86 selected primary studies (Multiple labels allowed).

The model landscape is heavily dominated by the OpenAI family. As shown in Fig. 13, *OpenAI* models appear in 74 of the 86 studies (86.0%). Although the primary studies cover a broad range of model families, including *Llama*, *Mistral*, *Claude* [6], *Gemini*, and *DeepSeek*, their combined presence remains far smaller than that of *OpenAI* models. This imbalance is consistent with a broader pattern observed in LLM4SE research [51]. The dominance of *OpenAI* models coincides with the public releases of ChatGPT [75] and GPT-4 [1], which marked a period of rapid growth in LLM adoption across SE research. Regarding how these models are accessed, *remote APIs* are by far the dominant access mode, appearing in 47 studies, while *web interfaces* are reported in 20 studies and *inference engines* in 12 studies. *Self-hosted APIs*, in contrast, are observed in only three studies (Fig. 14). Turning to access level, *black-box* access is the dominant mode, appearing in 52 studies (60.5%), followed by *grey-box* access in 29 (33.7%) and *white-box* access in only eight (9.3%)(Fig. 15). These findings suggest that most

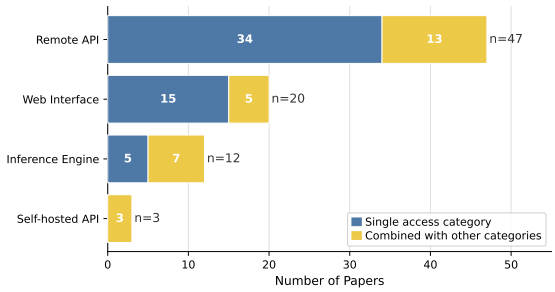

Fig. 14: Distribution of reported LLM access categories across 86 selected primary studies.

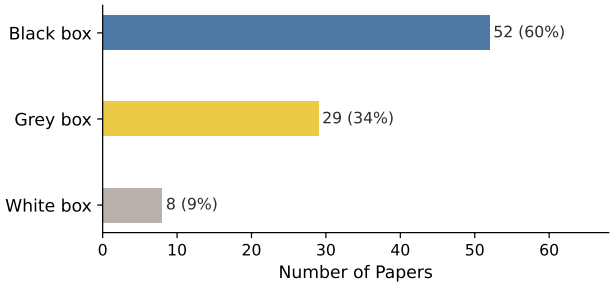

Fig. 15: Distribution of reported LLM access levels across 86 selected primary studies.

LLM4MDE approaches are built on top of externally hosted platforms, with limited use of self-managed deployment or model-level transparency.

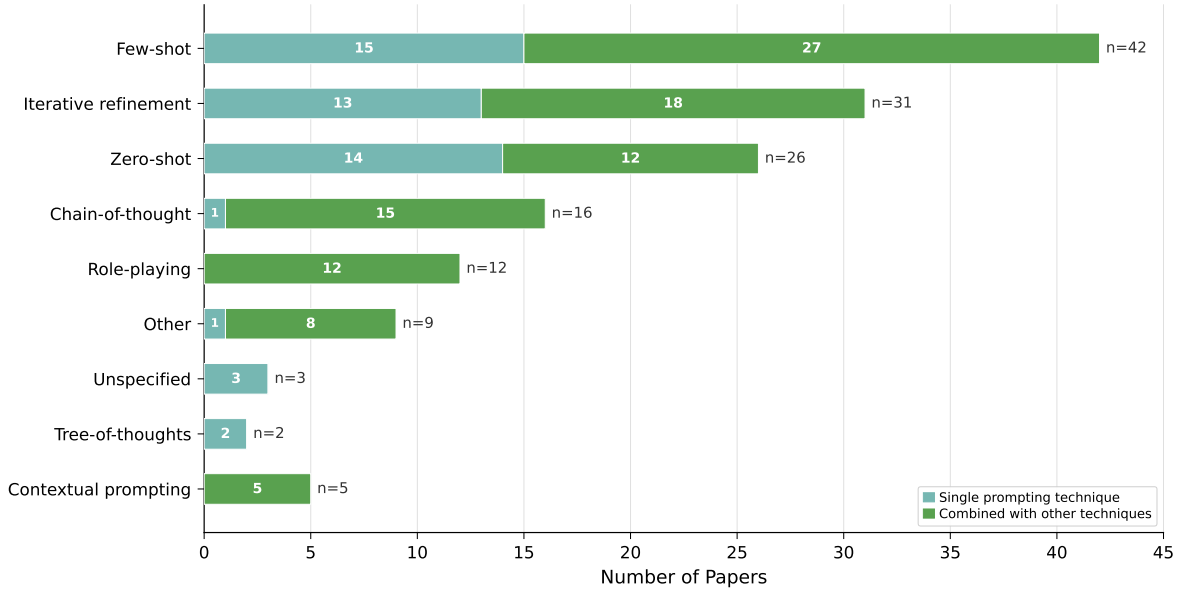

Fig. 16: Prompting strategies applied across 86 primary studies.

Prompt engineering is the primary mechanism through which studies adapt LLM behavior to MDE tasks. As shown in Fig. 16, *few-shot* prompting is the most frequently reported strategy, appearing in 42 studies, followed by *iterative refinement* in 31 and *zero-shot* prompting in 26. More deliberative or role-oriented strategies are less common, including *chain-of-thought* in 16 studies and *role-playing* in 12, while approaches such as *tree-of-thoughts* and *contextual prompting* appear only rarely. Beyond prompt engineering, other mechanisms for customizing model behavior are rarely employed. *Fine-tuning* is applied in only nine studies (10.5%), with 77 studies (89.5%) explicitly reporting *no fine-tuning* (Fig. 17). RAG strategies are less prevalent: 77 studies (90%) specify *no retrieval mechanism*, while *example retrieval*, *document retrieval*, and *metamodel retrieval* together account for only ten studies (Fig. 18). LLM sampling strategies are poorly reported: 56 studies (65%) specify *no sampling strategy* at all, and among those that do, *temperature-based sampling* is the most common approach, appearing in 24 studies (28%)(Fig. 19).

A further dimension concerns the level of autonomy given to these systems. As shown in Fig. 20, the reviewed literature is overwhelmingly *non-agentic*: 83 of the 86 studies (96.5%) fall into this category, whereas only three studies (3.5%) adopt an *agentic* design. This indicates that most approaches still operate within fixed, researcher-defined workflows in which the LLM is expected to generate outputs

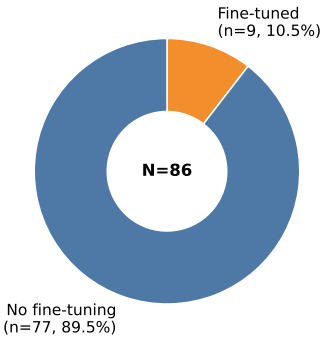

Fig. 17: The use of fine-tuning technology across 86 selected primary studies.

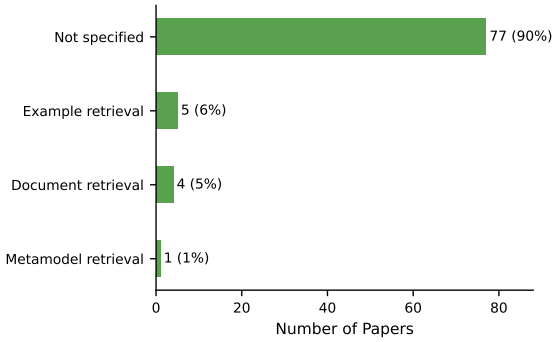

Fig. 18: Reported retrieval-augmented generation strategies across 86 selected primary studies.

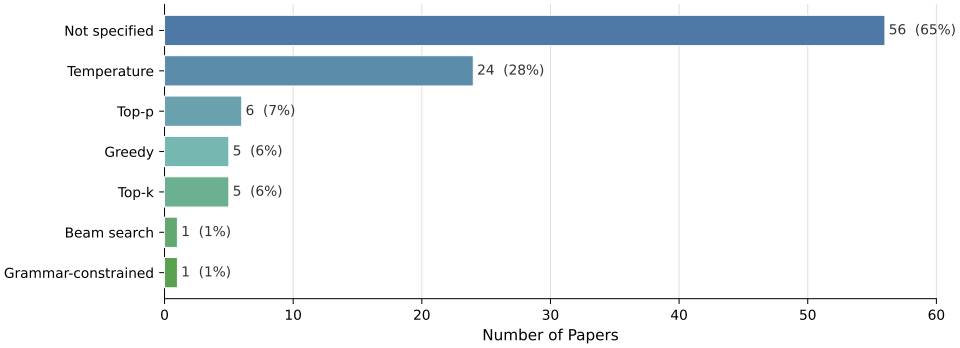

Fig. 19: Reported LLM sampling strategies across 86 selected primary studies.

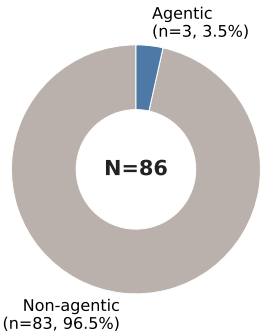

Fig. 20: Autonomy level of LLM approaches across 86 selected primary studies.

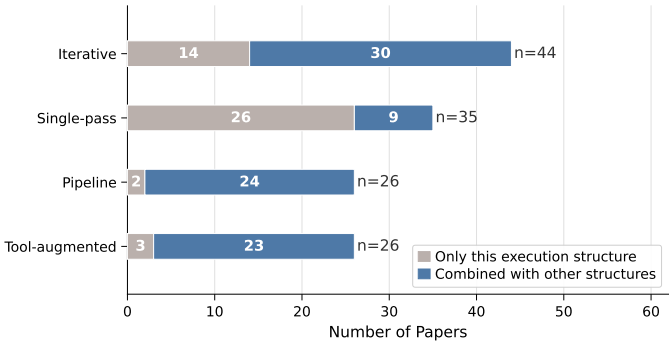

Fig. 21: Execution structures adopted across 86 selected primary studies.

inside a predetermined control structure. The three *agentic* studies are not evenly distributed across tasks: they cover *Model Generation*, *Model Transformation* combined with *Model Validation*, and *Metamodeling* combined with *DSL Engineering*, respectively. Two of the three are *multi-task* studies, and all three combine *agentic* behavior with *tool-augmented* execution and *human-in-the-loop* interaction, suggesting that, in the cases observed, *agentic* designs appear in scenarios that require coordination across multiple steps or artifacts rather than in routine *single-task* settings. Capabilities such as planning, autonomous redirection, or tool-driven decision making are rarely explored in current LLM4MDE research.

The predominance of *non-agentic* systems does not mean that these workflows are structurally simple. As shown in Fig. 21, *iterative* execution is the most common structure, appearing in 44 studies, while *pipeline* and *tool-augmented* structures are each reported in 26 studies. *Single-pass* execution appears in 35 studies. As shown in Fig. 22, execution structure and prompt engineering often co-vary. Among the 31

papers using *iterative refinement*, 25 also adopt *iterative execution*, and 25 of the 44 *iterative-execution* papers explicitly employ *iterative refinement*. By contrast, *chain-of-thought* prompting does not show a similarly strong association with *single-pass execution*: only six of 16 *chain-of-thought* papers are *single-pass*, while 11 are *iterative*. The clearer *single-pass* pairing is *zero-shot* prompting, which co-occurs with *single-pass* execution in 18 of 26 papers. Even when *agentic* autonomy is avoided, staged processing and iterative prompting remain common strategies for improving reliability and managing task complexity. These findings suggest that the dominant design logic in LLM4MDE is not autonomous agency, but structured orchestration around the model: the workflow itself, rather than the autonomy of the LLM, is where most of the methodological complexity in current approaches is concentrated.

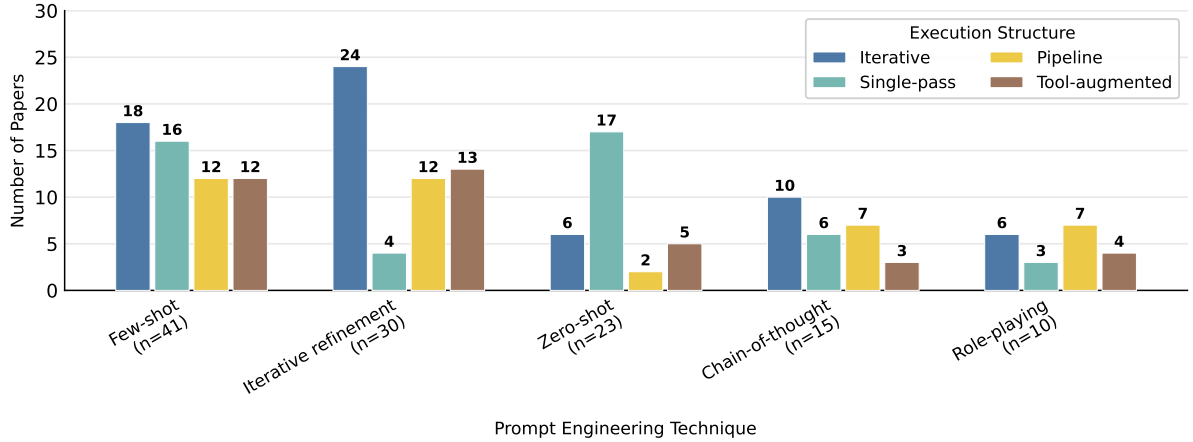

Fig. 22: Execution structures by prompting strategy across 86 selected primary studies.

We also examine how far these approaches move beyond standalone prototyping and into actual MDE environments. Fig. 23 shows that the large majority of the primary studies are implemented as *standalone* approaches: 75 out of 86 papers (87.2%) are not integrated with an existing MDE toolchain, suggesting that most current approaches treat LLMs as external components rather than as native capabilities of established MDE tooling.

Turning to the role of humans in these workflows, we found that human participation is common across the literature (Fig. 24). *Post-hoc human review* is the most common mode, appearing in 38 papers, followed by *human-in-the-loop* interaction in 34 papers. By comparison, only 11 papers report *no human involvement* at all, while ten papers rely on *human-provided input only*. In *post-hoc human review*, humans validate or approve LLM-generated outputs as a necessary step before they are considered complete. For example, in [86], expert reviewers assess the generated multidimensional schemata against ground truths after each experimental run. In *human-in-the-loop* interaction, humans intervene dynamically during the

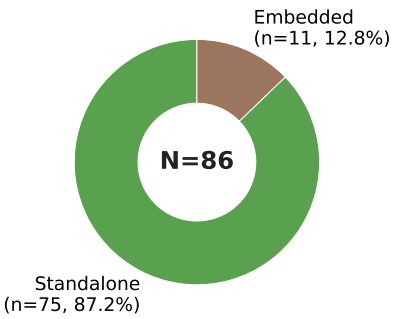

Fig. 23: The integration of LLM and MDE tools in 86 selected primary studies.

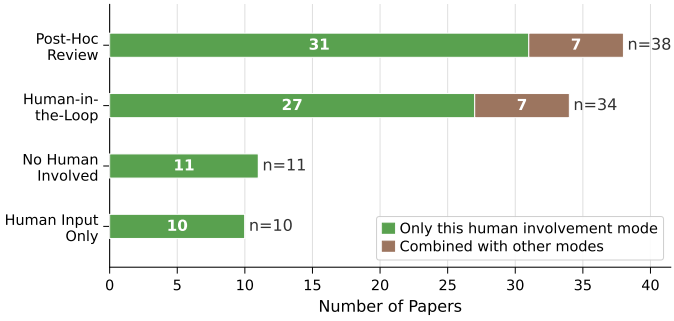

Fig. 24: The human involvement in the LLM process in 86 selected primary studies.

LLM process by providing iterative feedback or correcting intermediate outputs. For example, in [61], human analysts prune candidate modification ranges and review generated code before the workflow proceeds to the next stage. *Human-provided input only* refers to cases where humans supply the initial input but have no further involvement in the process. For example, in [120], once the prompt is finalized, the co-evolution workflow runs fully automatically without further human intervention. *Post-hoc human review* and *human-in-the-loop* interaction co-occur in seven papers, suggesting that in a subset of the literature, human participation is distributed across multiple stages of the workflow rather than limited to one stage. These patterns also vary across MDE task types, as illustrated in Fig. 25. *Code Generation*, *DSL Engineering*, *Meta-modeling*, and *Model Validation* have no paper coded as *No Human Involved*. The rates of *Human-in-the-Loop* (36%) and *Post-Hoc Human Review* (45%) of *Model Validation* are close to those of *Model Generation* (35% and 48%, respectively), indicating that validation is not more human-intensive than generation in relative terms. *Code Generation* shows a different profile: all five papers in this category involve *Human-in-the-Loop* interaction, and none report *No Human Involved*. The most automated profile appears in *Model Migration*, where two of three papers report *No Human Involved*, while among the better-represented categories. Across task types, human oversight remains a common feature of current LLM4MDE workflows, whether during generation or at the output review stage.

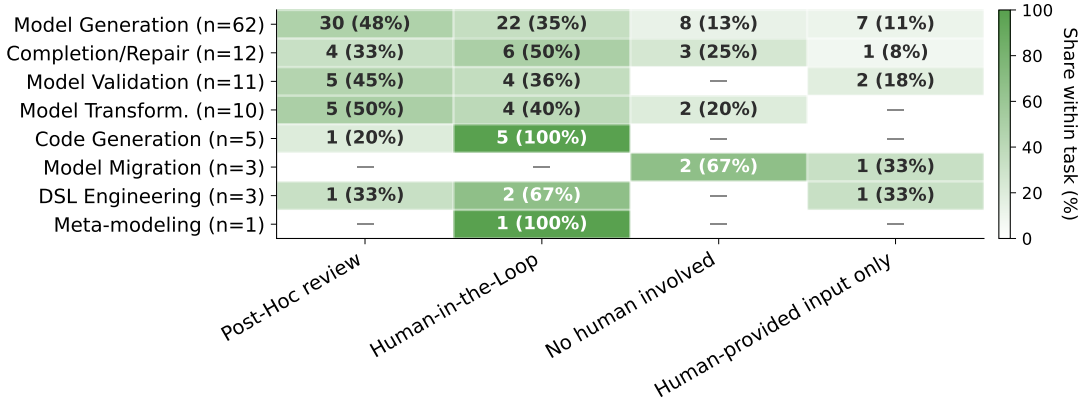

Fig. 25: Human involvement by MDE task type across 86 selected primary studies. Cell values report counts and within-task shares.

Answer to RQ2. The technological profile of current LLM4MDE research is characterized by reliance on externally accessed, black-box foundation models and prompt engineering, with limited engagement in fine-tuning, RAG, or tool integration. *OpenAI* models appear in 74 of 86 studies (86.0%), with the GPT-4 family alone accounting for 51 (59.3%). *Black-box* access is reported in 52 studies (60%). Among prompt engineering strategies, *few-shot* prompting is the most common (42 studies), followed by *iterative refinement* (31 studies) and *zero-shot* prompting (26 studies). By contrast, *fine-tuning* is applied in only nine studies (10.5%) and *RAG* in ten. At the system execution level, *iterative execution* is the most frequent structure (44 studies, 51%), and only three studies (3.5%) adopt an *agentic* design. This suggests that the dominant design logic in LLM4MDE is structured orchestration around the model rather than autonomous agency. Most approaches remain *standalone* prototypes (75 studies, 87.2%) rather than being *embedded in MDE toolchains*, and human participation through *post-hoc review* (38 studies, 44%) or *human-in-the-loop* interaction (34, 40%) is reported across the primary studies.

4.3 Artifact Representation and Processing in LLM4MDE Approaches (RQ3)

To answer RQ3, we examine how artifacts are represented as they enter and leave LLM processes in LLM4MDE research. Because individual studies may use more than one input artifact type, the input-

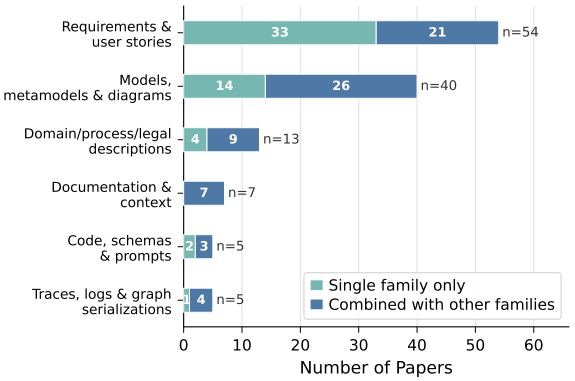

Fig. 26: Input artifact types across 86 selected primary studies. The stacked bars distinguish studies that use a type alone from those that combine it with other input artifact types.

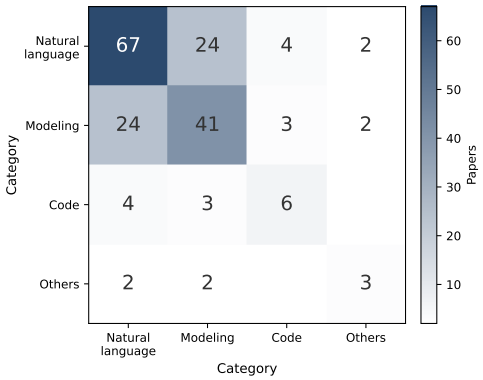

Fig. 27: Co-occurrence matrix for input artifact categories across 86 selected primary studies. Diagonal cells show category frequency; off-diagonal cells show how often two categories appear together.

side counts reported below are not mutually exclusive. They indicate whether a type appears in a study's LLM-based workflow, rather than implying that each study relies on a single input or that all reported inputs are provided to the LLM in one step.

On the input side, the literature remains oriented toward requirements-related text. As shown in Fig. 26, the largest input type is *requirements and user stories*, which appears in 54 of the 86 studies (63%). This type is not only common on its own, with 33 papers, but combining with other inputs is also common to see, appearing in 21 additional studies alongside model or contextual artifacts. *Models, metamodels, and diagrams* form the type with the second largest count, i.e., 40 papers (47%), but more often appear in combination with other input types (26 papers) than in isolation (14 papers). We categorized them, and the results are shown in Fig. 27. It shows that natural-language artifacts dominate the input artifact category, appearing in 67 studies (78%), whereas modeling artifacts appear in 41 studies (48%). Their co-occurrence is notable, with 24 studies using both, but combinations involving code remain rare. These results suggest that natural-language artifacts are the most common component of the input configurations reported in current LLM4MDE approaches, with model artifacts appearing regularly but more often in combination with other inputs than in isolation.

The data type and format results reinforce this finding. Fig. 29 shows that unstructured input is by far the most common data type, appearing in 70 studies (81%), compared with 35 studies (41%) using semi-structured input and only 12 studies (14%) using structured input. The most common to see mixed configuration is the combination of unstructured and semi-structured data, which occurs in 21 studies. Fig. 28 corroborates this: more than half of the studies, 48 out of 86 (56%), do not rely on any explicit model representation format at input time. Among the studies that do specify one, JSON and PlantUML are more common than XMI/Ecore. This suggests that studies often report them through prompt-compatible textual or lightweight structured forms rather than through explicitly stated native MDE encodings.

The output side covers a wider range of artifact types, but it remains centered on model-related results. As shown in Fig. 31, *models and diagrams* appear in 66 studies (77%), making them the dominant output family by a wide margin. This family is almost evenly split between standalone use, with 32 papers, and combined use, with 34 papers, indicating that model-oriented output is often the primary reported result, sometimes accompanied by other artifact types. Beyond this dominant family, the literature studies cover DSL grammars, instances, and formal specifications (18 studies), traceability- or suggestion-oriented outputs (17 studies), and code or executable artifacts (15 studies). The category-level view in Fig. 32 shows a similar distribution. Modeling artifacts appear in 68 studies (79%), far exceeding code artifacts with 15 studies (17%) and natural-language artifacts with 10 studies (12%). Even when outputs are combined, they usually remain within the modeling category rather than shifting away from it. Viewed across the full LLM process, this dominant output profile is consistent with the input-side finding that natural-language artifacts are the most frequent input type. As shown in Fig. 30, 55 of the 67 studies

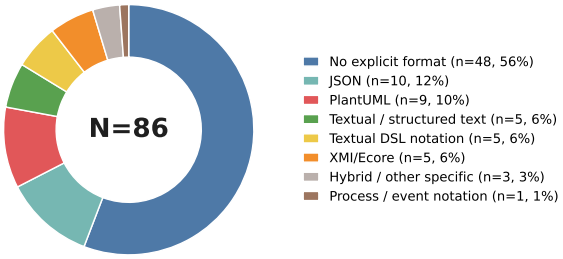

Fig. 28: Distribution of input representation formats. Categories are non-exclusive; the dominant slice corresponds to studies reporting no explicit model representation format, which does not imply the absence of model-related input.

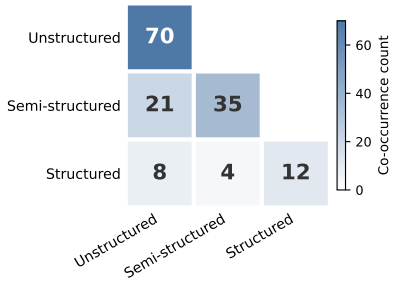

Fig. 29: Co-occurrence matrix for input data types. Larger bubbles indicate higher frequency; diagonal bubbles represent the number of studies containing each data type.

using natural-language artifacts as input (82%) still produce modeling artifacts as output, suggesting that current LLM4MDE approaches are largely organized around a text-to-model pattern rather than a model-to-model one.

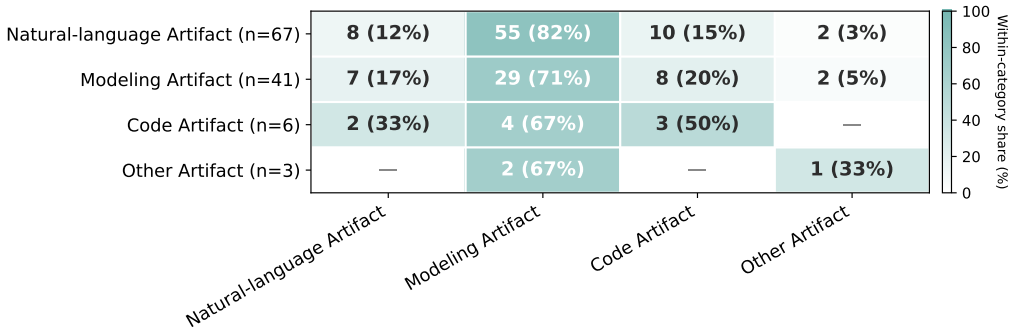

Fig. 30: Input-to-output artifact category alignment. Rows denote input categories, columns denote output categories in the same primary studies, and cell values report counts and within-input shares.

The output representation formats reflect the same tendency. Fig. 33 shows that semi-structured output is the dominant case, appearing in 59 studies (69%), whereas structured output appears in 20 studies (23%) and unstructured output in 16 studies (19%). Cross-type combinations are uncommon, which suggests that many approaches end with semi-structured rather than fully structured outputs. Fig. 35 makes this point more explicit. Although the output side is more explicit than the input side, *no explicit format* is still the largest family with 32 studies (37%). PlantUML, textual DSL notations, JSON, and other lightweight textual formats are more visible on the output side than on the input side, but XMI/Ecore remains a minority choice.

Comparing input and output data types reveals a similar degree of partial formalization. As shown in Fig. 34, 47 of the 70 studies with unstructured input (67%) produce semi-structured output, whereas ten of the 12 studies with structured input (83%) retain structured output. This indicates that LLM processes beginning from free-text artifacts often move toward semi-structured results, while LLM processes starting from structured artifacts are more likely to preserve that level of structure.

A similar pattern can be observed at the format level. As shown in Fig. 36, four of the five studies using *textual DSL notation* as input (80.0%) also output textual DSL, and five of the ten studies using *JSON* as input (50.0%) retain JSON at output. By contrast, only two of the five studies using *XMI/Ecore* as input (40.0%) preserve *XMI/Ecore* at output. This suggests that lightweight textual formats are more often maintained across LLM-based workflows than native MDE exchange formats in the studies examined. In

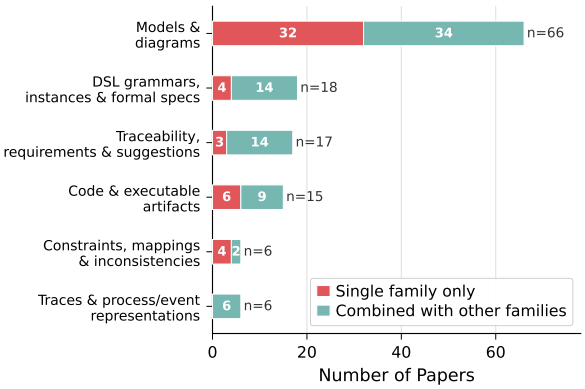

Fig. 31: Output artifact type families. The stacked bars separate studies in which a family is the sole output family from studies that combine it with other outputs.

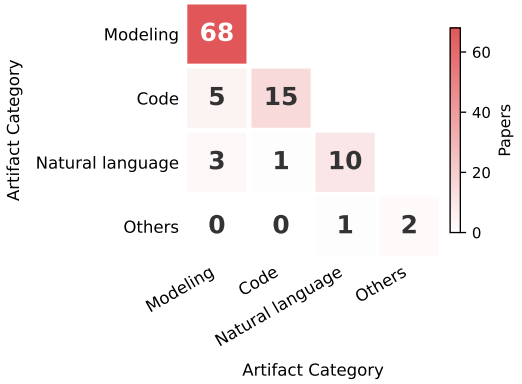

Fig. 32: Co-occurrence matrix for output artifact categories. Modeling artifacts dominate the diagonal and anchor most observed category combinations.

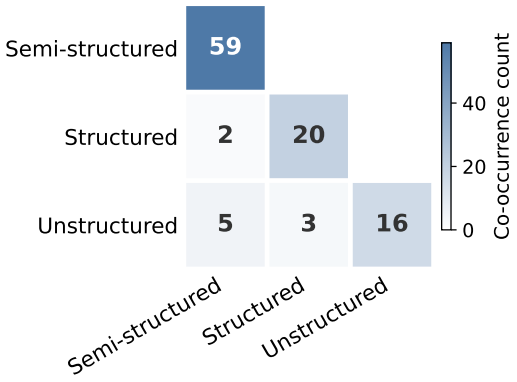

Fig. 33: Co-occurrence matrix for output data types across 86 selected primary studies. The diagonal emphasizes the predominance of semi-structured output over fully structured output.

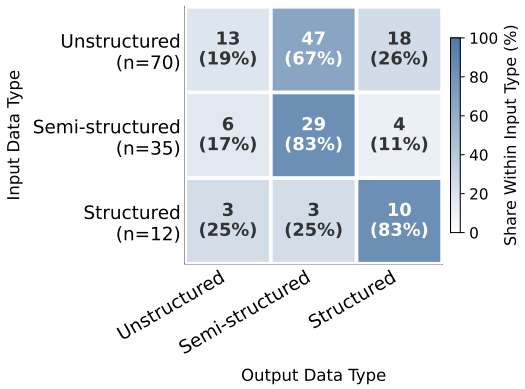

Fig. 34: Input-to-output data-type alignment across 86 primary studies. Rows denote input data types, columns denote output data types in the same primary studies, and cell values report counts and within-input shares.

other words, when explicit formats are used, lightweight textual representations are more common than native MDE exchange formats.

Answer to RQ3. Overall, the evidence suggests that current LLM4MDE processes are characterized by natural-language-dominant inputs and model-oriented outputs. Inputs are most often requirements, user stories, and other natural-language artifacts, sometimes supplemented with model context, while outputs are usually model-related but frequently expressed in PlantUML, JSON, textual DSLs, or other semi-structured formats rather than in native MDE exchange formats. The cross-stage results reinforce this pattern: studies with unstructured input often move to semi-structured output, whereas studies with structured input usually retain structured output; similarly, lightweight textual formats such as textual DSLs and JSON are more often preserved across the process than XMI/Ecore. These findings indicate that current LLM4MDE research more often represents artifacts in forms suited to LLM prompting than in forms directly reusable by MDE toolchains.

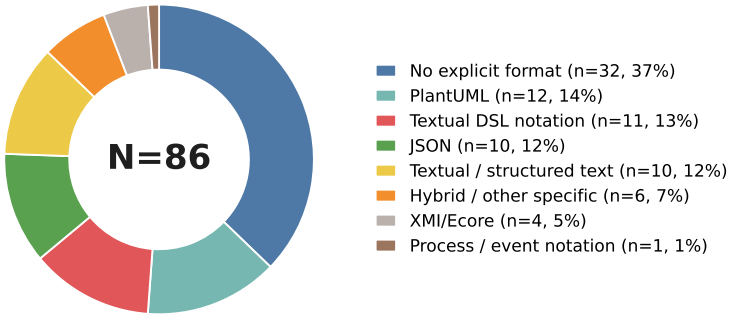

Fig. 35: Distribution of output representation format families across 86 selected primary studies.

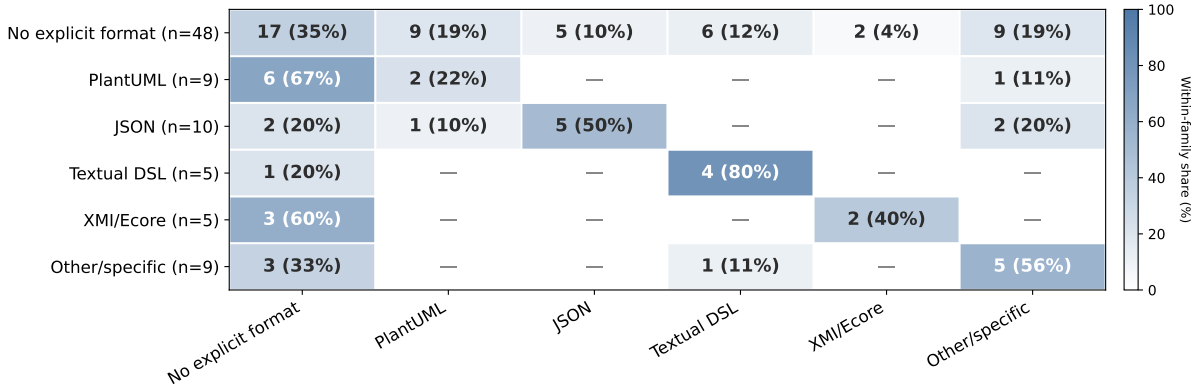

Fig. 36: Input-to-output representation-format alignment. Rows denote input format families, columns denote output format families in the same primary studies, and cell values report counts and within-input shares.

4.4 Validation Practices and Evaluation Design (RQ4)

To address RQ4, we examine how researchers validate LLM-based approaches for MDE by looking at the kinds of evaluation studies they conduct, the baselines they choose, the datasets and metrics they rely on, and a broader set of validation practices that reflect the maturity of empirical assessment. We then relate these validation choices back to the task types, prompting strategies, execution structures, and human involvement levels identified in RQ1, RQ2, and RQ3.

As shown in Fig. 37, current validation practice in LLM4MDE is centered primarily on quantitative experimentation. Among the 86 primary studies, 66 report quantitative experiments, more than comparison studies (32), case studies (21), or user studies (11), while controlled experiment and benchmarking studies appear in only one paper each. Because papers may adopt more than one evaluation type, these counts exceed 86. Overall, this distribution suggests that the field relies mainly on automatically computed measures over generated artifacts, whereas more human-centered forms of evaluation remain less common.

This emphasis on quantitative assessment is also reflected in baseline design. Fig. 38 shows that 36 papers (42%) do not include any baseline at all, evaluating the proposed approach in isolation. For example, Cámara et al. [15] assess ChatGPT's UML modeling capabilities by directly recording success rates and error patterns across prompting sessions, without comparing against any reference method. When a comparison is provided, the most common reference point is human or manual performance (19 papers), followed by traditional or rule-based approaches (14 papers). For example, Hachm et al. [50] compare their LLM-based transformation agent against a baseline agent using only generic transformation tools, measuring tool selection accuracy across 1,200 instructions. By contrast, only four papers

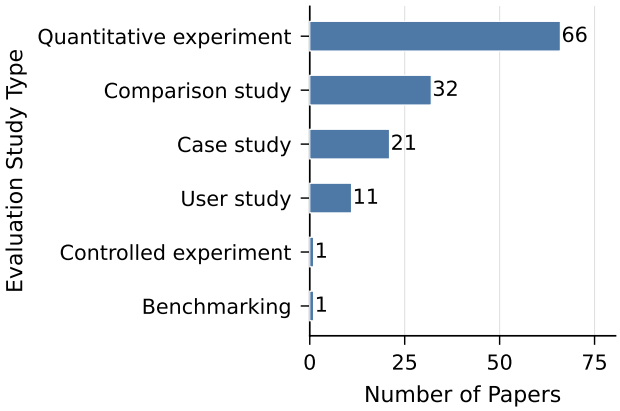

Fig. 37: Distribution of evaluation study types across the 86 primary studies (Multi-labels are allowed in a single paper).

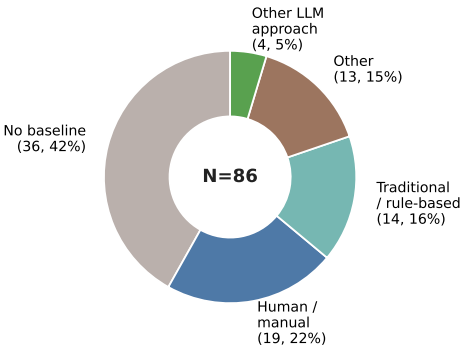

Fig. 38: Distribution of baseline comparison categories across the 86 primary studies.

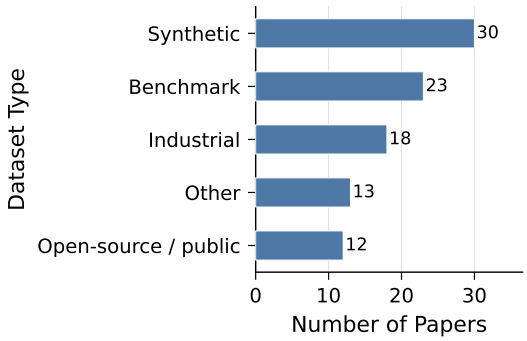

Fig. 39: Distribution of dataset types used for evaluation of the proposed approaches across the 86 primary studies (Multi-labels are allowed in a single study).

compare against other LLM-based approaches, while 13 papers rely on other forms of comparison. Taken together, these results suggest that many studies provide limited evidence about relative improvement, which makes it harder to compare results across studies and to assess claimed advances.

A similar pattern appears in the choice of datasets used to assess the proposed LLM-based approaches. As shown in Fig. 39, benchmark and synthetic datasets are the most frequently used categories (more than 20 papers each), while industrial datasets appear in 18 papers. Open-source or public datasets are used in 12 papers, 13 papers rely on other datasets and another seven papers do not specify the dataset type. This pattern suggests that researchers often favor controlled settings in which evaluation conditions can be managed more easily, although the use of industrial data in 18 studies shows that real-world applicability is examined in a subset of the literature.

The metrics reported in the literature reflect both general-purpose performance assessment and domain-specific evaluation criteria. Fig. 40 shows that Accuracy is the most common metric (31 papers), followed by syntactic correctness (29), semantic correctness (26), F1-score (22), and precision and recall (20 each). The most frequently reported metrics therefore combine standard information-retrieval measures with criteria specific to MDE, reflecting concern with both the statistical quality of the generated output and its conformance to modeling language constraints. At the same time, the appearance of metrics such as completeness (seven), cosine similarity (four), consistency (three), and hallucination rate (two) shows that a smaller set of studies also examines the semantic fidelity and reliability of LLM-generated models.

Looking beyond datasets and metrics, the broader validation practices summarized in Fig. 41 show uneven adoption across methodological dimensions. Eighty-one papers mention approach limitations,

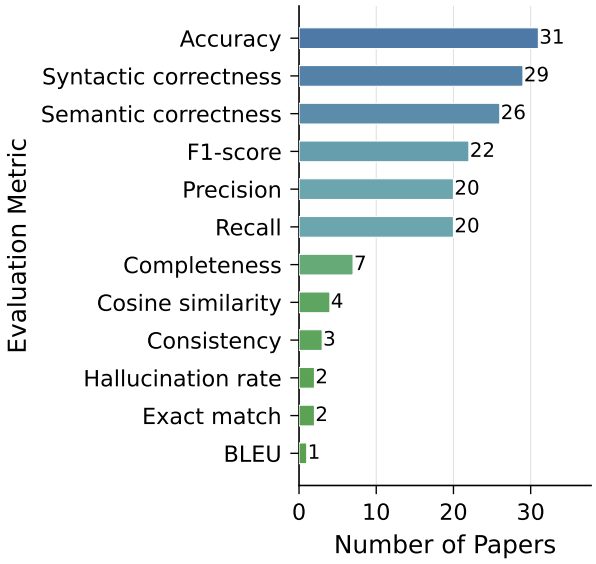

Fig. 40: Most frequently reported evaluation metrics across the 86 primary studies.

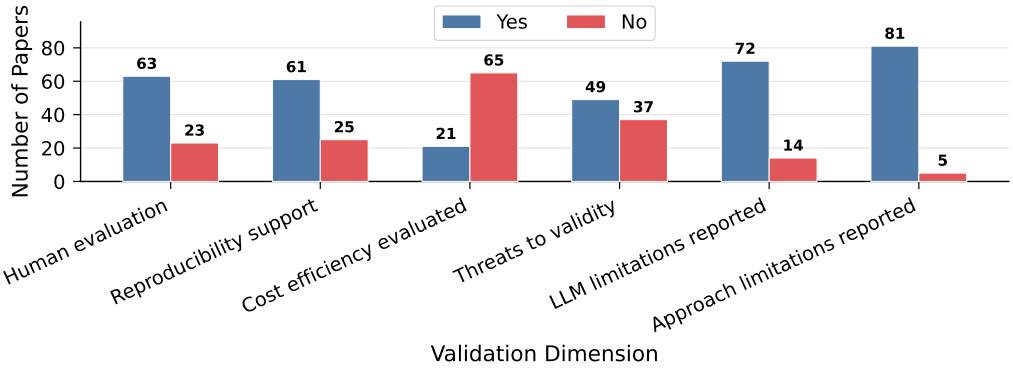

Fig. 41: Overview of six validation practices across the 86 primary studies.

making it the most widely adopted practice, while 72 papers mention LLM-specific limitations, 63 papers report human evaluation, and 61 papers report reproducibility support.

Threats to validity is provided in 49 papers, meaning that around half of the primary studies still offers no clear threats to validity discussion. The least frequently reported dimension is cost efficiency: only 21 papers evaluate the computational or financial cost of their approach, whereas 65 do not. Given the practical importance of deployment costs in industrial MDE settings, this omission is consistent with concerns raised in recent guidelines for empirical SE studies involving LLMs [8].

When these validation practices are examined together, only a few clear co-occurrence patterns appear. Fig. 42 shows that the strongest relationship is between reporting threats to validity and reporting reproducibility support ($\phi = 0.48$), suggesting that papers providing reproducibility materials are also more likely to discuss validity threats. A more moderate positive relationship appears between reporting approach limitations and LLM limitations ($\phi = 0.40$), indicating that papers discussing LLM-specific shortcomings usually also discuss broader limitations of the proposed approach. All other pairwise correlations are weak ($|\phi| < 0.35$), which suggests that the remaining practices are generally adopted independently rather than as part of a consistent validation package.

A complementary cross-tabulation within RQ4 further shows that evaluation design varies with both study type and dataset choice, as illustrated in Fig. 43. Quantitative experiments are conducted mainly on controlled datasets, especially benchmark datasets (23/63, 37%) and synthetic datasets (17/63, 27%), whereas case studies more often use industrial data (8/21, 38%). This suggests that controlled experimentation and real-world scenario validation still play partly distinct roles in the literature. Dataset type is

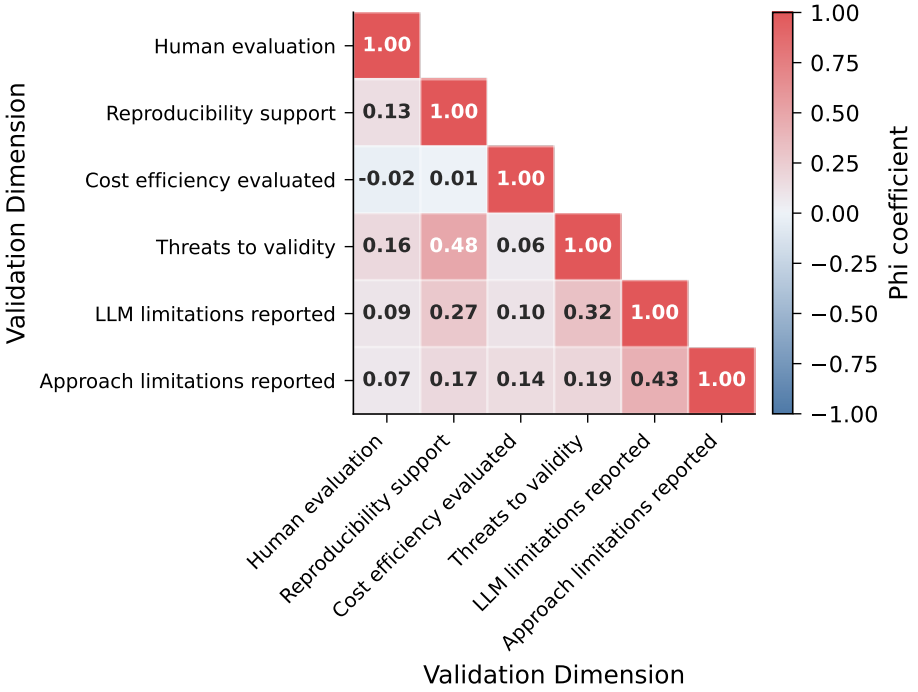

Fig. 42: Phi coefficient correlation matrix between the six validation practice dimensions.

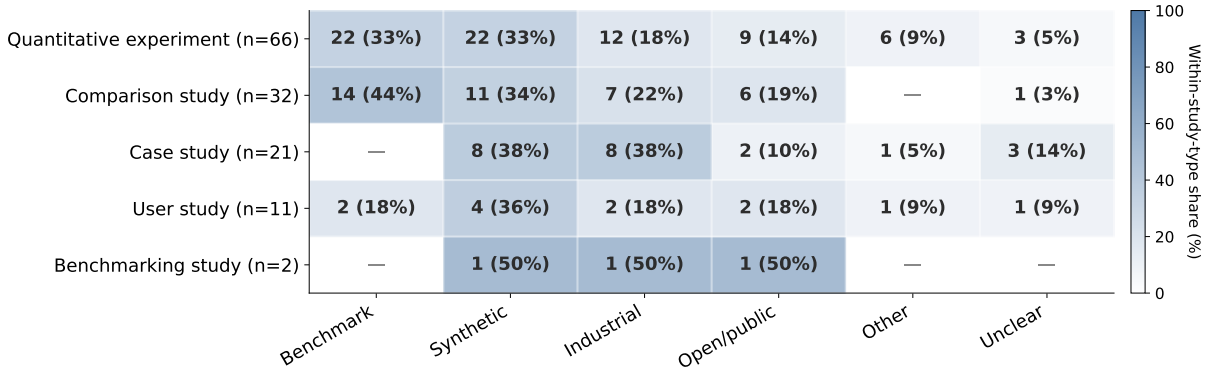

Fig. 43: Dataset types by evaluation study type. Cell values report counts and within-study-type shares.

also related to baseline design, as shown in Fig. 44. Studies using benchmark datasets more often compare against traditional or rule-based baselines (7/23, 30.4%) and less often omit baselines altogether (4/23, 17.4%), whereas studies using synthetic datasets are more often evaluated without any baseline (16/30, 53.3%). The highest rate of missing baselines appears in studies with unspecified or unclearly reported dataset types, where six of the seven papers (85.7%) provide no baseline. Case studies also report fewer validation elements overall: 14 of 21 case studies (67%) use no baseline, and only 19.0% discuss threats to validity, compared with 43% and 67%, respectively, for quantitative experiments. Overall, comparative validation is more common in studies using standardized datasets, whereas case-based evaluations and studies with unclearly specified datasets more often omit baselines and threats-to-validity discussions.

To extend this view beyond RQ4 itself, we next relate these validation patterns to the dimensions examined in RQ1, RQ2, and RQ3.

Fig. 45 shows that validation practices are not distributed evenly across MDE task types. Model Transformation papers achieve the highest reproducibility support among all task types (100%) and also maintain a relatively high rate of human evaluation (80%). Model Validation papers discuss approach limitations in all cases (100%) and also discuss LLM limitations at a comparatively high rate (91%). Code

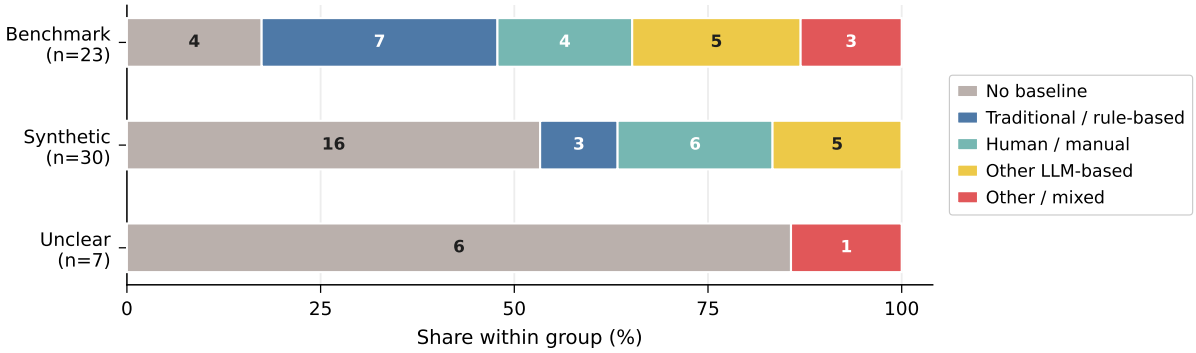

Fig. 44: Baseline comparison profiles in key dataset groups. Bar widths report within-dataset-group shares, and segment labels report counts.

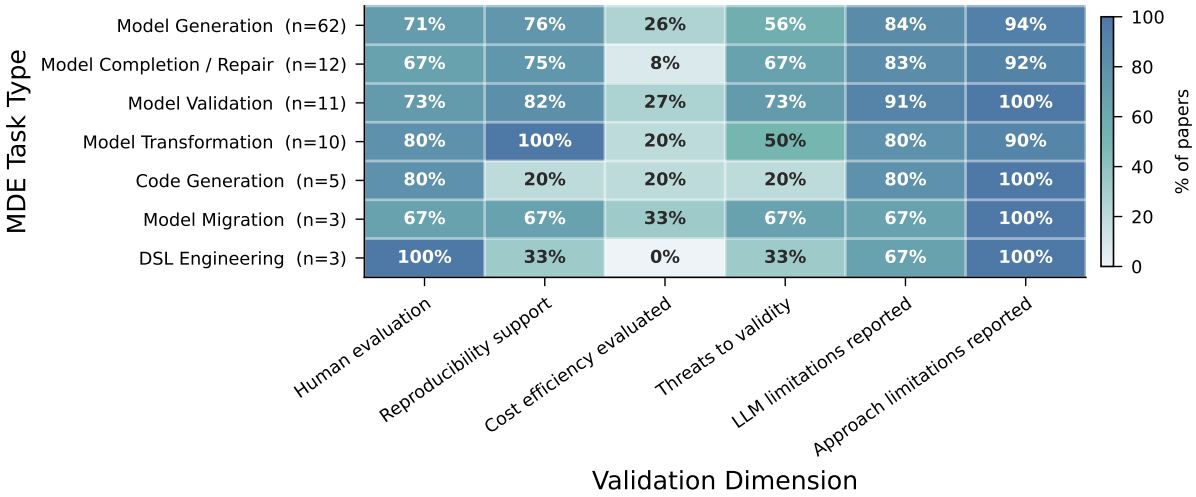

Fig. 45: Percentage of papers reporting "Yes" on each validation dimension, broken down by MDE task type (RQ1 × RQ4).

Generation papers, in contrast, report low reproducibility support (20%) and discuss threats to validity in only 20%, despite reporting approach limitations in all cases (100%). Across all task types, however, cost efficiency remains low, ranging from 0% to 33%, which suggests that this practical dimension is underexplored regardless of the task being addressed.

The limited use of baselines also persists when the results are broken down by task type. As shown in Fig. 46, Model Generation, which is the largest category with 62 papers, still includes 25 papers (40%) with no baseline comparison, although 17 compare against human or manual performance. Model Transformation and Model Validation show more frequent use of traditional or rule-based baselines and human-performance comparisons. Even so, comparisons against other LLM-based approaches remain rare across all task types, indicating that systematic benchmarking among competing LLM solutions is still uncommon.

A similar pattern appears when evaluation is viewed through the lens of prompt engineering. Fig. 47 shows that few-shot prompting, the most common technique, is evaluated mainly through quantitative experiments (32 papers), followed by comparison studies (18), user studies (eight), and case studies (five). Iterative refinement, zero-shot prompting, Chain-of-thought prompting, and Role playing similarly rely on quantitative experiments as the dominant evaluation type, though the relative proportions of other evaluation types vary. User studies, controlled experiments, and benchmarking are present across several prompting categories, but they remain a minority in all of them.

Execution structure is likewise associated with different choices of evaluation data. As shown in Fig. 48, iterative execution, which appears in 44 samples, uses the widest variety of dataset types,

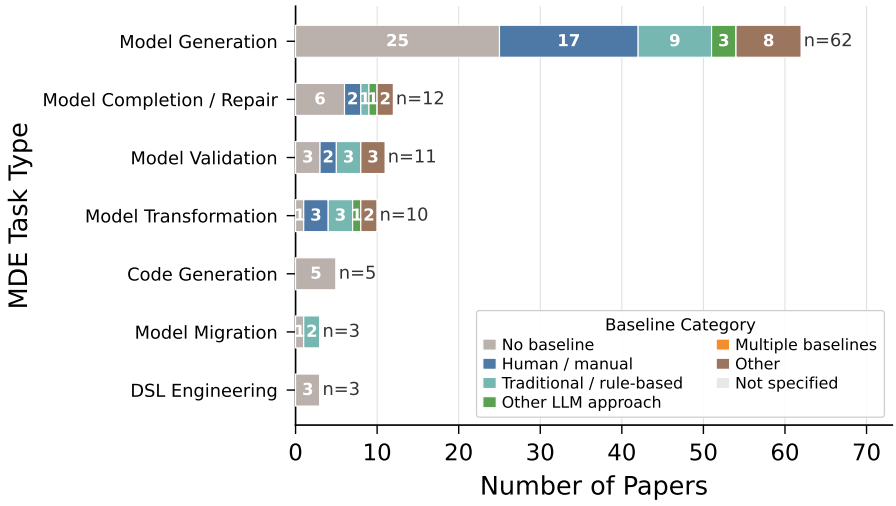

Fig. 46: Baseline comparison categories by MDE task type (RQ1 × RQ4).

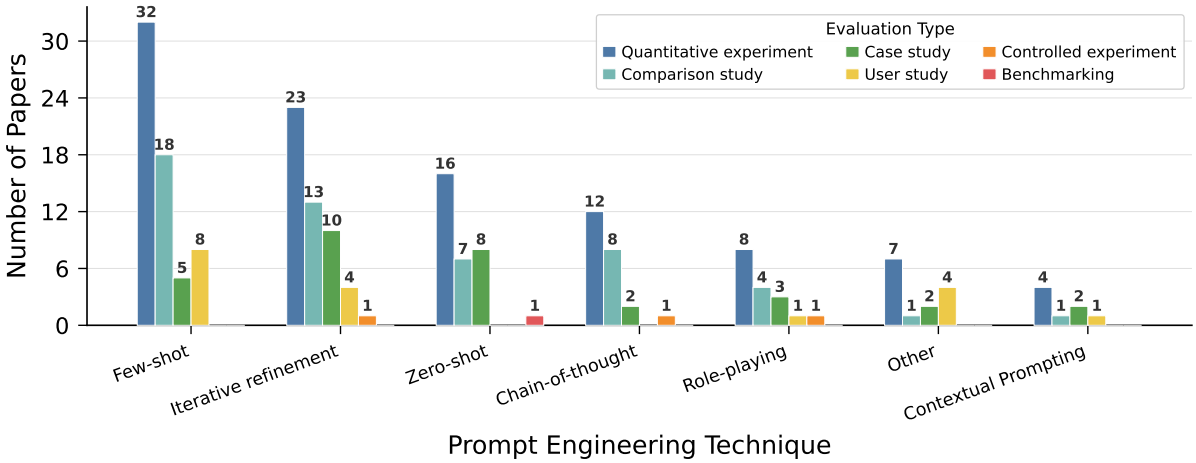

Fig. 47: Distribution of evaluation study types across the most common prompting strategies (RQ2 × RQ4).

with a relatively balanced distribution across benchmark (16), synthetic (nine), industrial (six), and open-source (four) datasets. Pipeline-based approaches (26 samples) are more often associated with industrial datasets. Single-pass execution (35 samples) follows a pattern closer to the overall distribution, with benchmark (nine) and synthetic (12) datasets being the most common choices. Tool-augmented approaches (12 samples) also rely mainly on benchmark (five) and synthetic (eight) datasets. Overall, more complex execution structures appear to be evaluated across a broader range of settings, whereas simpler structures are assessed mainly in more controlled environments.

Finally, Fig. 49 connects validation practice to the role that humans play in the approach itself. Papers with post-hoc human review report human evaluation most often (92%), followed by human-in-the-loop approaches (76%). Reproducibility follows a different pattern: post-hoc human review (84%) provide the highest levels of reproducibility support, whereas human-in-the-loop approaches show the lowest rate (56%), possibly because tighter human integration introduces variability that is more difficult to document or replicate.

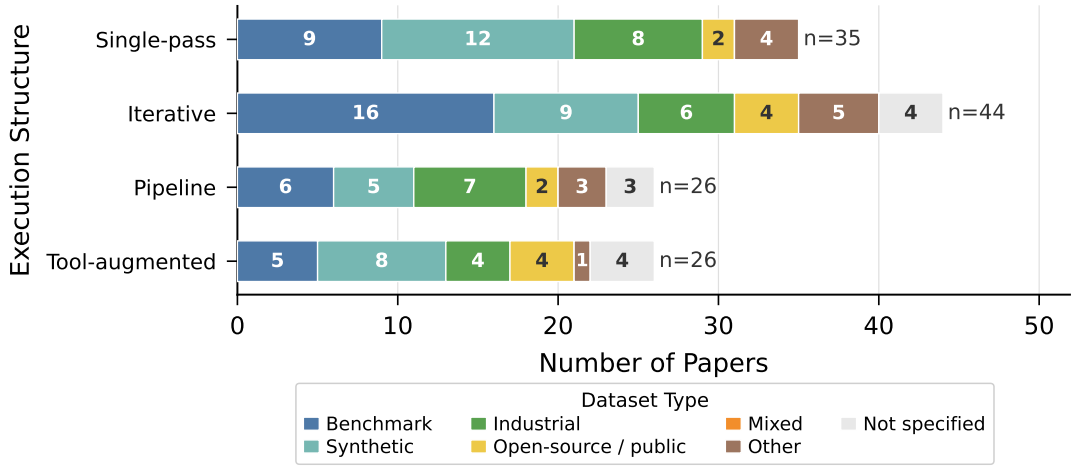

Fig. 48: Dataset types used for evaluation by execution structure (RQ2 × RQ4).

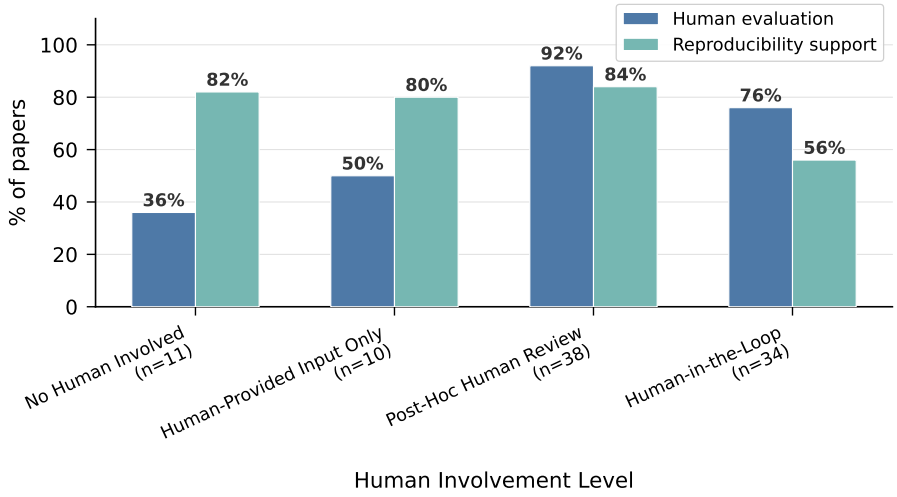

Fig. 49: Percentage of papers including human evaluation and providing reproducibility support, by human involvement level (RQ3 × RQ4).

Answer to RQ4. Validation in LLM4MDE is centered primarily on quantitative experimentation, which appears in 66 of the 86 papers and is most often reported through accuracy, syntactic correctness, and semantic correctness. Comparative evaluation remains limited: 36 papers (42%) use no baseline, and only four compare against other LLM-based approaches. Evaluation data come mainly from synthetic and benchmark datasets (30 and 23 papers), with industrial datasets appearing in 18 studies. Within RQ4 itself, quantitative experiments are concentrated on benchmark and synthetic data, whereas case studies more often use industrial data; studies using benchmark datasets are also more likely to include traditional or rule-based baselines, while studies using synthetic or unclearly specified datasets more often omit baselines. Among broader validation practices, reporting approach limitations is common (81 papers, 94.2%), reporting LLM limitations is available in 72 papers (83.7%), and cost efficiency is examined in only 21 papers (24%). Reproducibility support and threats-to-validity discussion ($\phi = 0.48$), as well as reporting LLM limitations and approach limitations ($\phi = 0.43$), both show moderate positive associations. When these patterns are related to RQ1, RQ2, and RQ3, Model Transformation reports high level of reproducibility support, few-shot and zero-shot prompting are still assessed mainly through quantitative experiments, and post-hoc human review is associated with the highest rate of human evaluation (92%) and also the highest reproducibility support (84%).

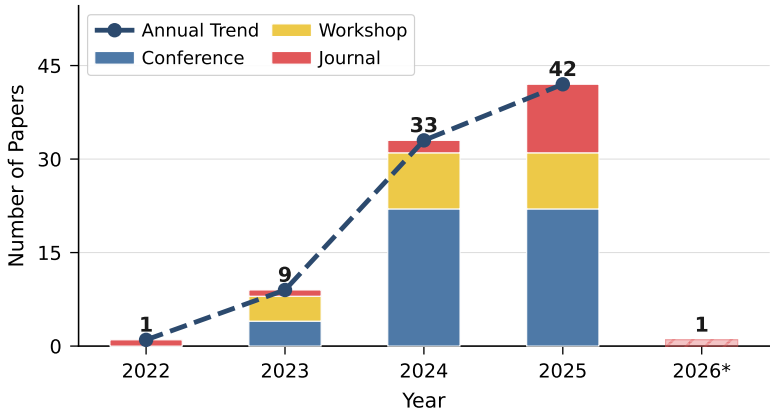

Fig. 50: Annual publication trend, stacked by publication venue category. Conference denotes Conferences (main track), Workshop denotes the combined category of Companion tracks and workshops, and Journal denotes Journals. **The bar for 2026* represents one paper collected up to January 2026.**

4.5 Publication Landscape and Research Community (RQ5)

In RQ5, we analyze the publication landscape of LLM4MDE along six dimensions: publication growth over time, geographic distribution, venue type and venue quality, collaboration structure, international collaboration, and funding sources [30]. Together, these dimensions describe the size of the literature, where it is published, which communities contribute to it, and how the area is currently organized.

Fig. 50 shows that LLM4MDE is a recent research area that has grown quickly. The primary studies contain one paper in 2022, nine in 2023, 33 in 2024, and 42 in 2025. The steepest increase occurs between 2023 and 2024. This period coincides with the release and wider availability of highly capable general-purpose LLMs, especially GPT-4 in March 2023 and Llama 2 in July 2023 [1, 103]. Growth first came mainly through conference and companion/workshop venues. Journal growth becomes more visible only in 2025. Relative to the broader LLM4SE literature, which had already reached 395 papers by January 2024 and was still described as being at an early stage [51], LLM4MDE appears to have developed later and at a smaller scale, despite our search extending to early 2026.

The geographic distribution is uneven. Fig. 54 shows that Canada and Germany lead with 15 papers each, followed by France with 13, and Sweden with 10. The most active countries are concentrated mainly in Europe, with additional contributions from Canada, China, and the United States. Because the counts are based on author affiliations, one paper may be credited to multiple countries. The figure therefore captures country participation rather than a mutually exclusive count of paper production. Cross-country collaboration is examined separately through the international co-authorship network. Geographic participation also intersects with research focus in revealing ways. As shown in Fig. 51, using broad regions on a participation basis, European-affiliated papers cover the widest range of task types: besides Model Generation (43/63, 68.3%), they include ten papers on Model Transformation, ten on Model Completion / Repair, eight on Model Validation, five on Code Generation, three on DSL Engineering, two on Model Migration, and the only Metamodeling paper. By contrast, Asia-Pacific participation is more concentrated on Model Generation (13/14, 92.9%), while Fig. 52 highlights that North American participation shows a lower share of industry-specific studies (2/19, 10.5%) than Europe (25/63, 39.7%) or Asia-Pacific (6/14, 42.9%). The regions also differ in language focus: as shown in Fig. 53, North American papers are evenly split between GPMLs and DSMLs (10 papers each), whereas European papers contain the largest number of MFL-related studies (8 papers). These descriptive patterns suggest that the geographic concentration of the field coincides with differences in topical breadth and industrial grounding.

The venue profile shows that LLM4MDE is still published mainly in conferences. As shown in Fig. 56 and Table 2, 48 of the 86 studies (55.8%) appear in conference proceedings, 22 (25.6%) in companion tracks and workshops, and 16 (18.6%) in journals. Venue rankings are distributed across tiers. Conference proceedings are distributed across CORE/ICORE tiers, with 20 papers in A*/A venues, 18 in B/C venues, and ten in unranked venues. All companion-track and workshop papers are unranked under

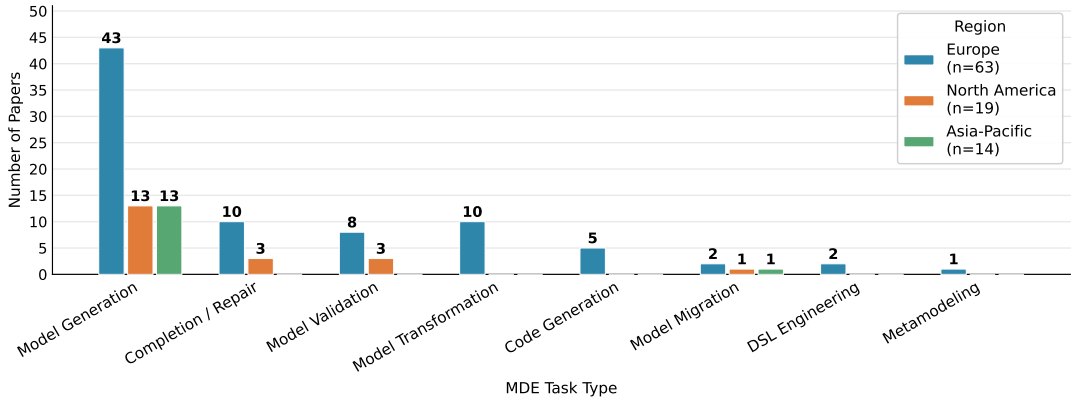

Fig. 51: Task-type profile by region (RQ5 × RQ1). Bubble area reports within-region shares based on participation counts, and labels report counts.

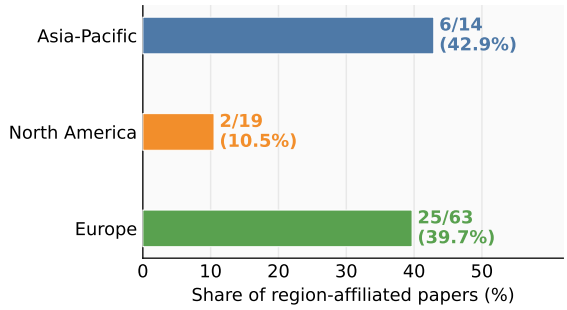

Fig. 52: Industry-specific share by region (RQ5 × RQ1). Bars report the number and proportion of region-participating papers coded as industry-specific.

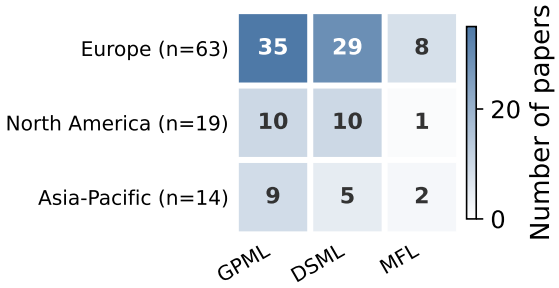

Fig. 53: Modeling-language profile by region (RQ5 × RQ1). Bubble area reports within-region shares based on participation counts, and labels report counts.

CORE/ICORE. The journal subset is smaller, but 13 of the 16 journal papers appear in JCR Q1 or Q2 venues. Conference quality tiers are based on the ICORE2026 conference rankings, and journal quality tiers are based on the quartiles reported in the 2025 release of Journal Citation Reports (JCR) [22, 25]. The most frequent venues are closely tied to the modeling and SE communities. MODELS-C is the single most frequent venue with 11 papers, followed by the MODELS main conference with six papers. Among conference proceedings, the most frequent venues are MODELSWARD (seven papers), MODELS (six papers), and ER (four papers). Among journals, *Software and Systems Modeling* (SoSyM, three papers) and *Information and Software Technology* (IST, two papers) appear most often. Venue tier also correlates with notable differences in validation practice, as shown in Fig. 55. Papers in CORE A*/A conferences and JCR Q1/Q2 journals (33 papers) are more likely than the remainder of the literature (53 papers) to include a baseline comparison (22/33, 67% vs. 27/53, 51%) and to discuss threats to validity (22/33, 67% vs. 27/53, 51%). They are also more often evaluated through quantitative experiments (29/33, 88% vs. 34/53, 64%). By contrast, the differences are small for human evaluation, cost-efficiency analysis, and industrial-dataset use. Even within the higher-tier subset, one third of the papers still use no baseline (11/33, 33%). This suggests that papers in higher-tier venues more often report certain validation elements, especially baselines and threats to validity, but do not differ consistently across all validation dimensions.

The collaboration profile remains fairly localized. As shown in the left panel of Fig. 59, single-institution papers are the largest group, with 44 of the 86 papers (51%). They are followed by international collaborations with 25 papers (29%) and domestic multi-institution collaborations with 17 papers (20%). International collaboration is therefore more common than domestic multi-institution collaboration. Industry participation is limited: only seven papers involve industry-academia collaboration, distributed

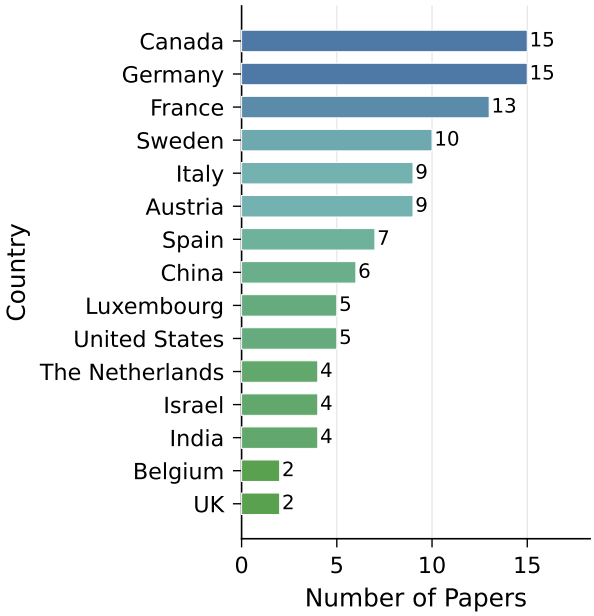

Fig. 54: Geographic distribution of LLM4MDE research: top 15 countries by paper count, based on author affiliations. A paper is credited to multiple countries when its co-authors have affiliations in different countries.

Table 2: Most frequent publication venues.

Venue	Full Name	Quality	#
Conference proceedings			
MODELSWARD	Int'l Conf. on Model-Based Software and Systems Engineering	CORE C	7
MODELS	Int'l Conf. on Model Driven Engineering Languages and Systems	CORE A	6
ER	Int'l Conf. on Conceptual Modeling	CORE A	4
ICSE*	Int'l Conf. on Software Engineering	CORE A*	2
RE	Int'l Requirements Engineering Conference	CORE A	2
MSR	Int'l Conf. on Mining Software Repositories	CORE A	2
Companion tracks & workshops			
MODELS-C	Int'l Conf. on Model Driven Engineering Languages and Systems: Companion Proceedings	—	11
REW	Int'l Requirements Engineering Conf. Workshops	—	4
LLM4SE	Workshop on Large Language Models for Generative Software Engineering	—	2
Journals			
SoSyM	Software and Systems Modeling	JCR Q1	3
IST	Information and Software Technology	JCR Q1	2

*Includes one NIER-track paper.

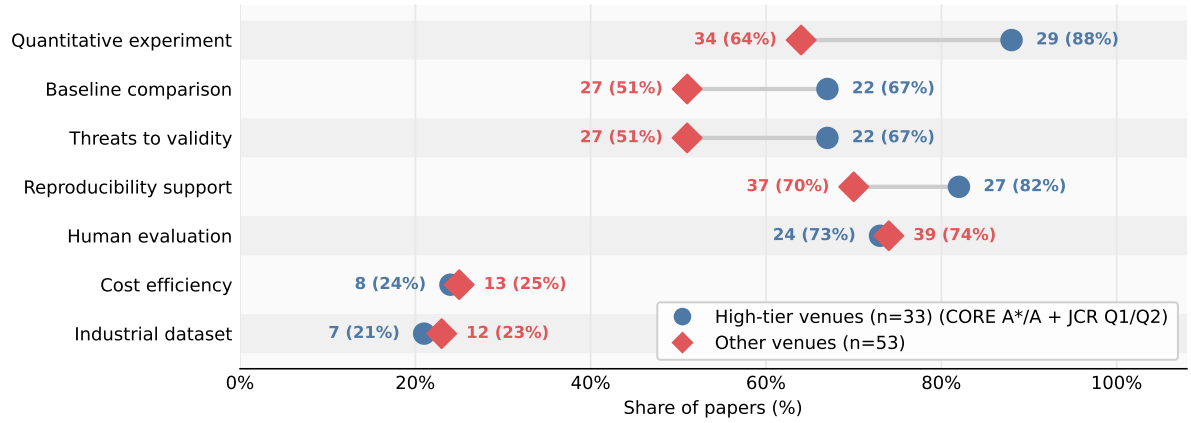

Fig. 55: Validation practices by venue tier (RQ5 × RQ4). High-tier venues comprise CORE A*/A conferences and JCR Q1/Q2 journals; cell values report counts and within-tier shares.

across domestic multi-institution (six) and international collaboration (one). The temporal view in the right panel of Fig. 59 shows that international collaboration increases from three papers in 2023 to eight in 2024 and 13 in 2025, while domestic multi-institution collaboration remains smaller. All seven industry-academia collaborations appear only in 2025. Collaboration structure is also associated with distinct technical choices, as illustrated in Fig. 57. Internationally collaborative papers most often use OpenAI models (24/25, 96.0%) and are usually implemented as standalone approaches (22/25, 88.0%), with little use of tool-augmented execution (2/25, 8.0%) and no agentic designs in this subset. Single-institution papers more often report tool-augmented execution (20/45, 44%), fine-tuning (6/45, 13%), and embedded MDE-tool integration (7/45, 16%). Domestic multi-institution papers are also mostly standalone (15/16, 94%). These descriptive patterns suggest that more customized or integrated prototypes are currently

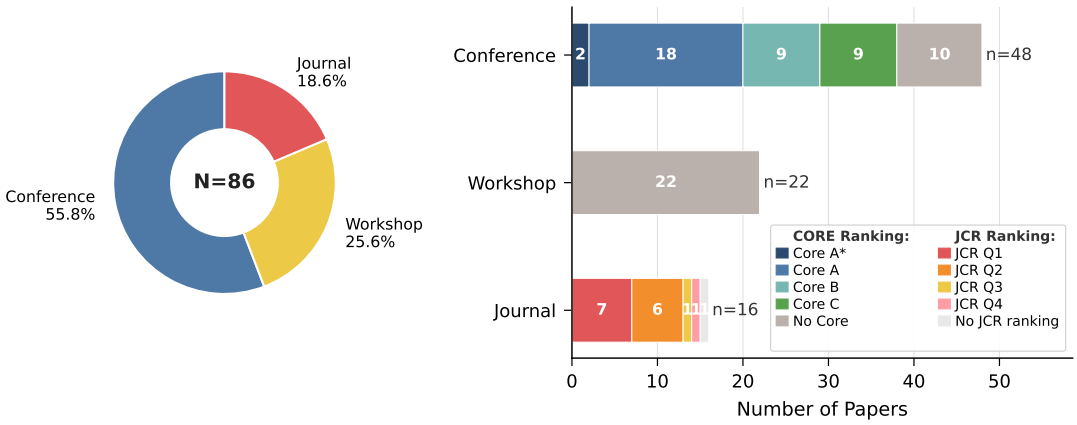

Fig. 56: Venue category and publication-quality profile. *Conference* denotes Conferences (main track), *Workshop* denotes the combined category of Companion tracks and workshops, and *Journal* denotes Journals. `Left:` distribution across publication venue categories. `Right:` CORE and JCR quality-tier breakdown within each venue category.

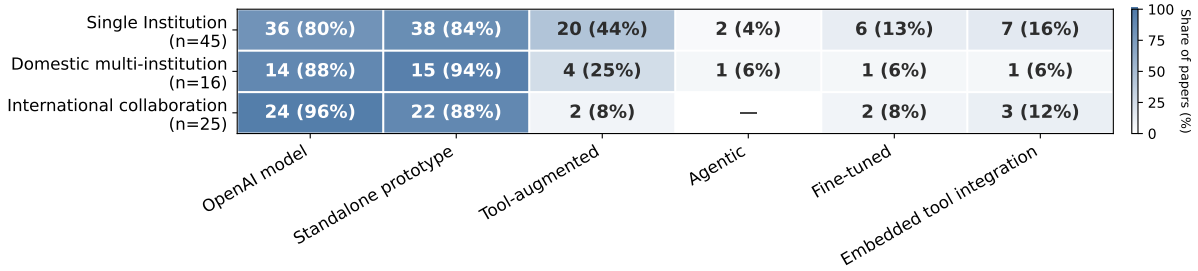

Fig. 57: Technical design choices by collaboration structure (RQ5 × RQ2). Cell values report counts and within-collaboration-group shares.

reported more often in single-institution settings, whereas cross-institution collaborations more often rely on standard externally accessed model configurations.

A related pattern emerges when venue profile is viewed through the lens of collaboration structure. As shown in Fig. 58, CORE A*/A conferences and JCR Q1/Q2 journals contain a larger share of international collaborations, whereas unranked conference papers are mostly single-institution papers. Industry-academia collaboration is observed only in 2025 in the current sample: all seven such papers are from that year, and six of them are domestic multi-institution collaborations, suggesting that industry involvement is still limited in the current literature.

Fig. 60 gives a more detailed view of international collaboration by country. The left panel shows that leading countries do not participate in international collaboration to the same extent. Germany and France have a larger share of single-institution publications, whereas Canada, Sweden, Spain, and Italy have a larger share of internationally collaborative papers. The right panel shows that international collaboration is concentrated in a small number of recurring country pairs rather than being broadly distributed across the full network. The strongest collaboration is between Canada and Spain (four papers), followed by Canada and Sweden (three papers), and then Austria and Italy (two papers), Germany and Sweden (two papers), and Israel and the Netherlands (two papers). The international co-authorship structure is therefore concentrated in a small number of recurring country pairs rather than broadly distributed across the network.

To examine how the field is supported, we classify acknowledged funding sources into four main categories: *Gov./Public Grant*, *Industry*, *University/Institutional*, and *None Mentioned*. In this scheme, *Gov./Public Grant* includes ministries, research councils, and public agencies; *Industry* refers to commercial or industrial funding; and *University/Institutional* covers internal institutional support. The category *None Mentioned* includes papers that do not report funding information and should not be

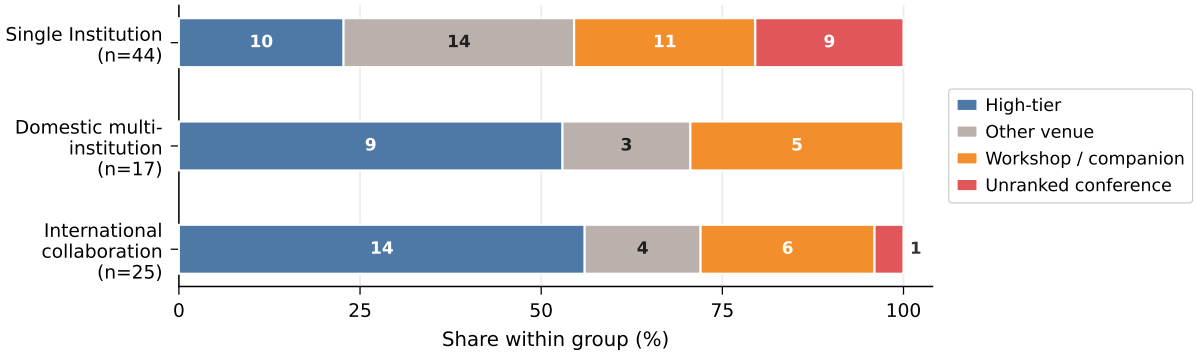

Fig. 58: Venue profile by collaboration structure (RQ5 orthogonal-analysis). Bar widths report within-collaboration-group shares across high-tier venues, other venues, workshops/companion tracks, and unranked conferences, and segment labels report counts.

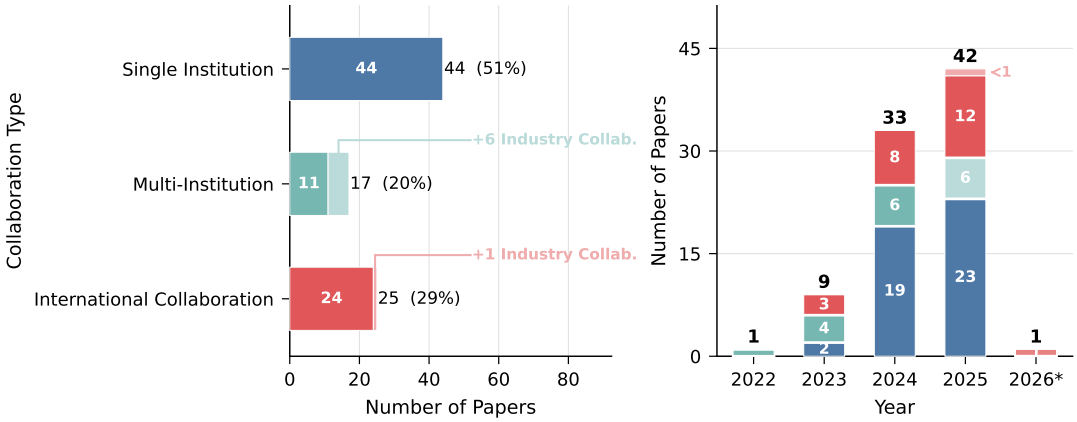

Fig. 59: Collaboration structure across the 86 selected primary studies. `Left:` overall distribution by collaboration type; lighter bars indicate the subset involving industry-academia collaboration within each group. Institution involves both the industry and academia. `Right:` temporal evolution of collaboration types. The 2026* bar represents one paper collected up to January 2026.

read as evidence of no funding. When a paper acknowledges more than one funding source category, we code it as a mixed category, such as *Gov.+University* or *Gov.+Industry*. This classification follows broad sectoral distinctions commonly used in research funding statistics [10, 74].

As shown in the left panel of Fig. 61, the most common pattern is missing funding information: 45 of the 86 studies (52%) do not mention any funding source. Among the papers that do report funding, public support is the dominant pattern. Thirty studies (35%) report only government/public funding, and a further five studies report a combination of government/public funding with university/institutional funding. By contrast, university/institutional funding alone appears in only three papers (3%), and industry funding is reported only in combination with public funding (three studies, 3%); no paper reports industry funding alone. The temporal breakdown in the right panel of Fig. 61 shows that papers funded by *Gov./Public Grant* increase from five in 2023 to nine in 2024 and 15 in 2025. However, *None Mentioned* remains the largest category in both 2024 (19 papers) and 2025 (23 papers). Industry remains limited in both funding and collaboration: only three papers report any *Industry* funding, and only seven papers involve industry-academia collaboration.

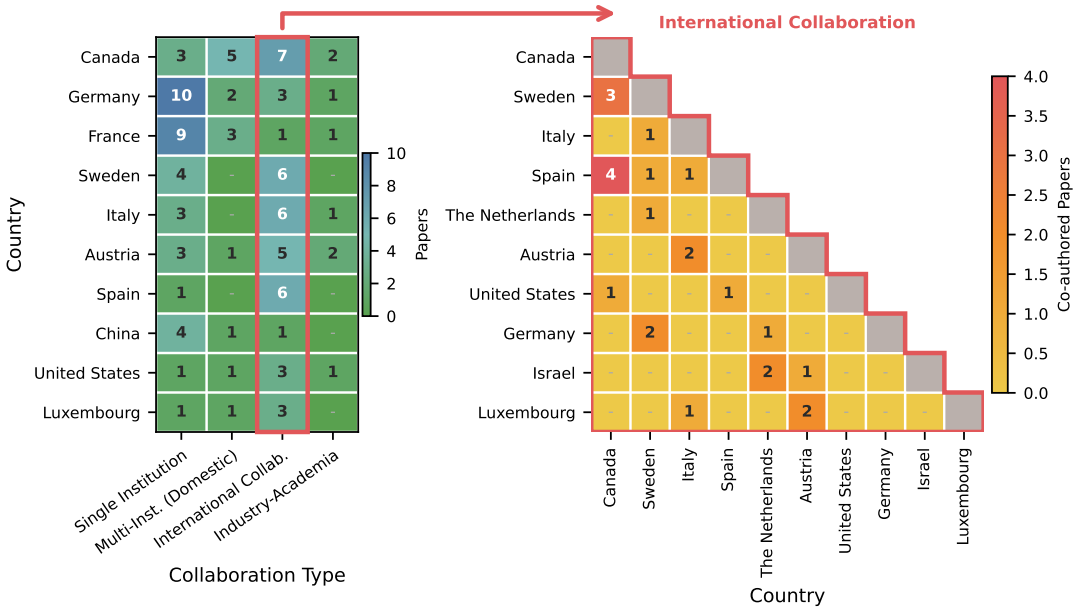

Fig. 60: International co-authorship patterns. *Left:* country by collaboration type heatmap for the top 10 countries by paper count; the highlighted column marks the internationally collaborative subset. *Right:* symmetric country co-authorship matrix for papers involving international collaboration.

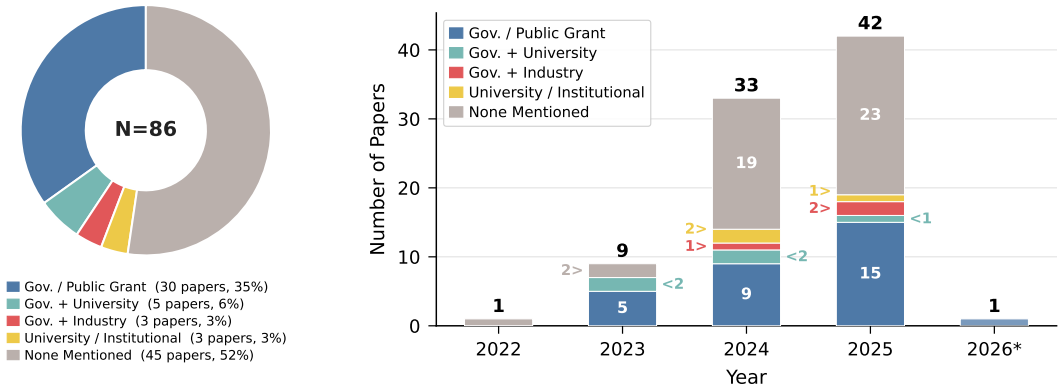

Fig. 61: Funding source distribution across the 86 selected primary studies. `Left:` overall distribution across five funding categories. `Right:` temporal evolution of funding categories. The 2026* bar represents one paper collected up to January 2026.

Answer to RQ5. The publication landscape of LLM4MDE shows fast recent growth, from one paper in 2022 to 42 in 2025. Most papers are published in conference proceedings or companion/workshop venues, while journal output becomes more visible in 2025. The literature is concentrated mainly in Europe and Canada. Collaboration is still dominated by single-institution academic work, although international collaboration has increased over time and is more common in JCR Q1/Q2 journals and top-ranked conferences than in unranked conference venues. Funding is reported inconsistently, but reported funding is dominated by government and public grants. Industry involvement remains limited in both funding and collaboration.

5 Discussion

5.1 Evaluation Recommendations for LLM4MDE Approaches

From a top-down perspective, the evaluation of LLM4MDE approaches can be viewed along three aspects: study setup and reporting, evaluation metrics and quality criteria, and comparison and validation design. Our review suggests that current LLM4MDE research still shows limitations in each of these aspects. Based on these findings, we offer a set of preliminary recommendations intended to inform future efforts toward establishing community-wide evaluation guidelines for LLM4MDE research. We do not claim these recommendations as definitive or exhaustive; they draw on our mapping findings together with existing frameworks from the empirical SE and MDE communities, and should be read as a starting point for community discussion rather than a prescriptive standard.

The need to spread the word on following the Baltes' guidelines LLM-based studies are sensitive to factors such as model version, access mode, prompts, interaction procedure, and decoding settings. Baltes et al. [8] propose eight guidelines for empirical software engineering studies with LLMs, covering the declaration of the LLM role, reporting of model version and configuration, documentation of prompts and interaction logs, reporting of tool architecture, use of human validation, inclusion of open-LLM baselines, selection of suitable benchmarks and metrics, and reporting of limitations and mitigations. Our results indicate that reporting remains incomplete in several respects. Prompts, interaction details, and sampling settings are often only partially reported, and cost information is absent in 76% of the studies (RQ4). These omissions limit reproducibility and make performance differences harder to interpret.

A lack of model-oriented metrics in evaluations Current LLM4MDE studies often report metrics such as accuracy, F1-score, and syntactic correctness (RQ4). These metrics are useful, but they do not always align with the kinds of artifacts that LLM4MDE approaches produce. As shown in RQ3, modeling artifacts appear in 79% of the studies as output artifacts. Yet model-oriented criteria such as completeness, consistency, and correctness are assessed explicitly only in a small part of the literature. In many studies, outputs are evaluated mainly as textual results rather than as engineering artifacts that must function within broader MDE workflows.

A lack of explicit baselines Validation design also varies across the literature: nearly half of the studies do not include any baseline (RQ4), comparisons against other LLM-based approaches remain rare (RQ4), comparisons with open-source LLMs are also uncommon (RQ2), and human evaluation is often not structured around explicit criteria or rubrics.

Recommendations These gaps should not be interpreted as showing that MDE lacks evaluation foundations. In contrast, MDE already has established work on artifact-oriented quality assessment and empirical evaluation of model-driven development claims [44, 78]. What the current LLM4MDE literature suggests, rather, is a partial disconnect between these practices and the recent growth of LLM-based studies. The task is therefore not to define evaluation principles from scratch, but to connect LLM4MDE evaluation more closely to both empirical LLM-study rigor and artifact-oriented MDE evaluation.

To address these issues, we organize a set of preliminary recommendations for evaluating LLM4MDE approaches into a two-layer framework. The first layer focuses on general empirical rigor in LLM-based studies in the MDE context. The second layer focuses on artifact-sensitive evaluation, with the goal of complementing generic output metrics with criteria that better match MDE artifacts and workflows. For the second layer, we draw on the SWEBOK model-analysis framework [110], in particular its five criteria for analyzing models: *completeness*, *consistency*, *correctness*, *traceability*, and *interaction*. These criteria make evaluation more sensitive to the roles that generated artifacts play in MDE settings and provide clearer constructs for organizing expert judgment.

Table 3 summarizes the resulting two-layer framework. Layer 1 (G1–G8) adapts the recommendations of Baltes et al. [8] to the MDE context, while Layer 2 (G9–G13) introduces five MDE-specific model-analysis criteria derived from SWEBOK [110]. Table 4 then shows how the second-layer criteria apply differently across output artifact types identified in RQ3, and which techniques and metrics are appropriate in each case.

Table 3: A two-layer framework of preliminary evaluation recommendations for LLM4MDE approaches.

Guideline	Recommended Practice for LLM4MDE research
Layer 1 — General Empirical Rigor	
G1. Declare LLM role	Specify the role of the LLM in the MDE workflow (e.g., model generator, model transformation synthesiser, constraint validator); justify the LLM model choice used for the specific type of MDE task and integration scenario.
G2. Report LLM model version, configuration & customization	Document the LLM model name, version, and all parameters configured in the experiments; in the case of tool integration (e.g., Eclipse plugin, web platform), record API endpoint URL, access mode, and access dates.
G3. Report tool architecture	Document the full MDE tool stack (e.g., EMF [102], Sirius [107], Xtext [34], ATL [57]), including the integration point with the LLM; in the case of RAG processes, include metamodel or model fragment retrieval; in the case of agent systems, include tool invocation and reasoning mechanisms.
G4. Document prompts & logs	Provide complete prompts with injected metamodel definitions, grammar definitions, or model fragments; report the number of iterations in prompt engineering; report representative input-output logs in the case of the modelling task.
G5. Use human validation	Use human validation based on clear MDE analysis and evaluation criteria; for model artifact results, use rubrics based on Layer 2 guidelines (G9-G13); include inter-rater reliability results.
G6. Use open LLM as baseline	Compare to open-source LLMs and traditional MDE methods (e.g., grammar-based generator, ATL rule set, search-based tool) for the same task.
G7. Use suitable benchmarks & metrics	Select metrics based on output artifact type (Table 4); discuss construct validity for the task; complement generic metrics with MDE analysis and evaluation criteria (Layer 2).
G8. Report limitations, cost & sustainability	Report token usage and API call cost results; report threats in MDE tasks: hallucinated model elements, constraint violations for metamodels, scalability for metamodel and model size, prompt sensitivity to modeling language syntax.
Layer 2 — MDE-Specific Model Analysis Criteria	
G9. Assess Completeness	Assess whether all required elements, relationships, and constraints are generated or preserved; main focus for model generation and migration tasks (see Table 4).
G10. Assess Consistency	Detect inconsistency within and across artifacts using techniques such as OCL validation [14] and model differencing [28]; important for model repair and validation tasks.
G11. Assess Correctness	Assess syntactic correctness (metamodel conformance checking) and semantic correctness (domain correctness) separately; apply to model-artifact outputs.
G12. Assess Traceability	Verify whether links between requirements, models, code, and transformations are preserved or recoverable; important for model transformation, model migration, and code generation tasks (see Table 4).
G13. Assess Interaction	Evaluate artifact behavior in MDE workflows (transformation chains, synchronisation, simulation); mainly relevant for model-artifact outputs in multi-step MDE pipelines.

Overall, these recommendations suggest that evaluation in LLM4MDE should shift from output-level evaluation alone to artifact-level and workflow-level evaluation. They also suggest that human evaluation should be tied to explicit quality constructs rather than used only as an informal judgment mechanism. More consistent use of first-layer recommendations, together with better reproducibility support, cost reporting, and baseline comparisons, would improve the comparability of LLM4MDE studies. We offer these recommendations as a starting point for future community efforts to establish more comprehensive and widely adopted evaluation guidelines in this field.

5.2 The Role of Human Involvement in Validation Practice

Fig. 49 shows that human evaluation and reproducibility support rates differ across human involvement levels. Papers with *post-hoc human review* report the highest rate of *human evaluation* (i.e., a validation step in which humans score or judge LLM-generated outputs) at 92%, and also the highest *reproducibility support* at 84%. By contrast, *human-in-the-loop* papers score lower on both dimensions (76% and 56%), while papers with no *human involvement* show a *reproducibility support* rate of 82

For the lower reproducibility in *human-in-the-loop* approaches, one possible explanation is that each human intervention in the loop, such as pruning candidate outputs or correcting intermediate results, constitutes a decision point that is difficult to document fully; prompt logs and interaction logs alone cannot adequately capture these human judgment steps. This resonates with the guideline proposed by

Table 4: Artifact-sensitive evaluation criteria organized by LLM output artifact type.

Output artifact	LLM4MDE tasks	Applicable Layer 2 guidelines	Recommended techniques & metrics
Model artifact	Model Generation; Model Completion/Repair; Model Transformation (M2M); Model Migration; Metamodeling; DSL Engineering (metamodel); Model Validation (OCL output)	G9 (Completeness) and G11 (Correctness) are primary to model generation tasks, while G10 (Consistency) is primary to model repair tasks, and G12 (Traceability) and G13 (Interaction) are most critical to model transformation and migration tasks.	Metamodel conformance checking [77]; OCL constraint checking [14]; reference-based model differencing [28]; trace link analysis [92]; model instantiation and validation [33]. *Metrics:* parsability/conformance rate; model element precision/recall/F1; constraint satisfaction rate; structural similarity ratio; trace link coverage.
Code artifact	Code Generation; Model Transformation (M2T); DSL Engineering (grammar)	G12 (Traceability) is applicable to the output to ensure model-to-code or model-to-grammar correspondence. G9–G11 and G13 are not applicable to code outputs. The correctness and completeness of the output should be judged by code-specific criteria.	Compilation and test-suite execution [18, 83]; source model to code coverage analysis; grammar well-formedness checking; instance parsability checking; CodeBLEU-based similarity assessment [85]. *Metrics:* compilation success rate; Pass@1/Pass@k [18]; CodeBLEU [85]; model element coverage ratio; parsability rate.
Natural language artifact	Model Validation (output report); Model Generation (descriptive output)	Although G9–G13 do not directly apply to the output itself, they are relevant as evaluation criteria. In particular, human evaluation (G5) should be guided by the relevant model analysis criteria (G9–G13) in determining whether the output is accurate and complete.	Human evaluation of adequacy, factual consistency, and hallucination. [52, 63, 66] *Metrics:* adequacy score; hallucination rate [52]; human preference score; inter-rater agreement [23, 68].

Baltes et al. [8] on documenting interaction logs: *human-in-the-loop* settings are exactly the scenarios in which this guideline is hardest to satisfy.

For the difference in *human evaluation* reporting rates, one possible explanation is that when humans are already embedded in the workflow, researchers may treat this as an implicit quality assurance mechanism and therefore consider a separate human evaluation phase unnecessary.

These two patterns point to a deeper issue: current research lacks a systematic connection between the two roles of *human involvement*, i.e., as a workflow design decision and as an evaluation strategy. Future work should consider the impacts on reproducibility and evaluation completeness when making human involvement design choices.

5.3 Threats to Validity

We discuss threats to validity following the standard classification of construct, internal, external, and conclusion validity.

Construct Validity. One threat concerns how we operationalized "MDE task" and "LLM-based". The boundary of what counts as MDE is not always obvious, and borderline cases required judgment. For instance, a study on LLM-based Simscape simulation modeling [53] was excluded as it does not involve core MDE activities, while a study on LLM-based goal modeling with GRL [16] was included because goal modeling constitutes a form of requirements engineering that produces formal models. Similarly, distinguishing "LLM-based" from broader "AI-based" or "NLP-based" approaches was not always clear-cut.

A second threat concerns our classification schemes for data extraction. Some taxonomies were taken from existing work *Research Contribution Type* follows [99], *Input Data Type* and *Output Data Type* follow ISO/TR 8344 [55], *Evaluation Type* follows [59], and *MDE Task Type* is grounded in established MDE literature (Section 2.1) but others were developed by our team through iterative discussion during pilot extraction. These include *Modeling Language Classification*, *LLM Access Categories*, *LLM Access Level*, *Execution Structure*, *Human Involvement*, *Input Artifact Categories*, *Output Artifact Categories*, *Dataset Type*, and *Reproducibility Support*. Alternative categorizations are certainly possible and could lead to different interpretations. We tried to reduce this risk by grounding classifications in prior work where possible, applying cross-field consistency rules to ensure coherence among related fields, and documenting all definitions and example values in the replication package [112]. Similarly, the field *Prompting Strategy* combines structural prompting techniques (e.g., few-shot, chain-of-thought) and iterative prompt refinement strategies under a single dimension, following common practice in the LLM4SE literature; finer-grained distinctions, as proposed by Schulhoff et al. [96], were not applied during extraction.

Internal Validity. The main internal validity threat is subjectivity when applying the inclusion and exclusion criteria. To address this, we randomly assigned the 1358 candidate studies to seven researchers, where each study was screened independently by one researcher and verified by another. Disagreements were resolved by involving a third researcher. The same dual-review procedure was used during snowballing. We also adopted an adaptive reading depth strategy [79] to keep screening decisions consistent. During data extraction, interpretation of paper content may vary across extractors. We mitigated this through a structured extraction form, cross-field consistency rules, and calibration sessions among team members during the pilot phase. As with any systematic review, publication bias is also a concern, since studies reporting negative or inconclusive results are less likely to appear in the literature.

External Validity. Several factors limit the generalizability of our findings. The field of LLM-based MDE is moving quickly, and some results may already be outdated by the time of publication. Our restriction to English-language papers (criterion I2) may have excluded relevant work in other languages. The research questions and fields we chose, while intended to be broad, do not cover every possible angle; future work could examine aspects such as user experience, industrial adoption barriers, or the cost of LLM usage in MDE workflows. Finally, most of the included studies present academic prototypes rather than industrial deployments, so our findings may not transfer directly to practice.

Conclusion Validity. The most obvious threat is search completeness: we may have missed relevant studies. To reduce this risk, we searched five databases (IEEE Xplore, ACM Digital Library, Scopus, ScienceDirect, and Web of Science) following the recommendations of Kitchenham et al. [58], refined our search string through pilot searches, and supplemented the automated search with one round of backward and forward snowballing [113]. Another concern is that some sub-categories in our analysis contain only a few studies, making observed patterns less reliable; we have avoided drawing strong conclusions from such small groups. Lastly, given how fast the field is growing, quantitative trends such as the year-over-year increase in publications depend on where the search cutoff falls, and readers should keep this in mind.

5.4 Future Research Agenda

Based on our observations, we sketch the following research agenda.

A systematization of LLM errors, prompting design, and learning curves when addressing MDE tasks
Although this mapping study covers five dimensions, including research distribution, LLM technology and configuration, artifact representation, validation practices, and publication landscape, several aspects that may be of value to the research community were not included in the analysis. First, while this study recorded whether the selected papers report LLM limitations, it did not systematically investigate the types of errors produced by LLMs in MDE tasks (e.g., hallucinated model elements) or their distribution, and such an analysis may be useful for how to apply LLMs in MDE. Second, although RQ2 documents the frequency of prompting strategies, the content and structure of prompts themselves (e.g., how metamodel information is encoded in prompts) were not examined, and an investigation at this level may offer practical guidance on prompt design. In addition, the experience of researchers and practitioners using LLM4MDE tools, including usability, learning curve, and the difficulty of integrating these tools into

existing MDE workflows, falls outside the scope of this mapping study, but is relevant to understanding the adoption barriers in this field. These aspects should be considered in future work.

Addressing the benchmark problem Our results also map the evaluation practices in existing LLM4MDE research. The majority of LLM4MDE studies do not include any baseline comparison, and the development of benchmarks and datasets has been limited: the datasets used in existing research consist primarily of synthetic and benchmark datasets, most of which are constructed by the researchers themselves, with no standardized evaluation resources widely recognized by the community. In addition, the reporting of cost efficiency is severely lacking. Future work could consider developing guidelines for how to apply LLMs to MDE tasks, covering aspects such as cost efficiency reporting and baseline establishment, with the preliminary recommendations proposed in Section 5.1 serving as a starting point. Future work could also consider constructing standardized benchmarks and datasets to support systematic comparison across different approaches and to facilitate the reproduction of experimental results.

Tool chain integration Our results also shed light on the current state of industrialization: LLMs are implemented as standalone approaches in most of the studies, without integration with existing MDE toolchains; the industry-specific share of Model Validation (36.4%) is close to that of Model Generation (37.1%) and lower than that of Model Completion/Repair (50.0%), which does not align with the expectation given its direct relevance to quality assurance, suggesting that current industry participation is driven more by productivity-oriented concerns. Future work should strengthen collaboration with industry and promote the transfer of LLM4MDE approaches to practical engineering settings.

Widening the coverage of MDE tasks, task combinations, and LLM usage The data presented in Section 4 maps the distribution of LLM applications across MDE tasks and the technical configurations adopted. First, current LLM4MDE research is heavily concentrated on Model Generation, while research on several important MDE tasks, such as Metamodeling, remains extremely limited. Furthermore, LLM4MDE studies predominantly focus on single MDE tasks, with multi-task combination studies being rare and systematic cross-task research being absent. Second, the technical configurations adopted in current LLM4MDE research are highly homogeneous. Models beyond OpenAI/GPT have not seen broader application in the MDE tasks, and important technical approaches, including white-box access, fine-tuning, and RAG remain unexplored in MDE contexts. Future work could extend the LLM4MDE research in two directions: first, by applying LLMs to other MDE tasks than just Model Generation, especially model migration, DSL engineering, and metamodeling, where research is currently scarce; second, by introducing more diverse technical configurations, including other LLM models than just OpenAI/GPT and techniques such as fine-tuning and RAG, to assess their applicability across different MDE tasks.

At a more specific level, we identify two directions of MDE research that have received little attention when it comes to LLMs.

Exploring the potential of LLMs in multi-level modeling (MLM) and multi-view modeling (MVM) While existing work has explored the application of LLMs to two-level modeling tasks, the potential of LLMs in multi-level modeling (MLM) [7] and its integration with multi-view modeling (MVM) [84], remains unexplored. Recent advances in this space, such as the C-SLICER framework [39, 40], which combines multi-level and multi-view modeling to address the complexity of industrial systems, highlight the complexity of modeling tasks that future LLM4MDE research must address, including multi-level instantiation and cross-level constraint propagation across an arbitrary number of modeling levels [41]. Multi-view modeling demands the maintenance of consistency across heterogeneous, overlapping perspectives of the same system [42]. These challenges exceed the two-level assumption underpinning most current LLM4MDE approaches. LLMs could contribute to several aspects of such integrated frameworks: (1) assisting in the automated construction of multi-level hierarchies, thereby lowering the steep learning curve associated with MLM formalisms; and (2) facilitating cross-view consistency checking by leveraging their capacity for semantic understanding to detect and suggest resolutions for inter-view conflicts. Given that industrial modeling scenarios inherently demand both multiple hierarchical levels and multiple stakeholder perspectives, as reflected in standards such as ISA-95 and IEC 62264, investigating the synergy between LLMs and integrated MLM-MVM approaches represents a direction that has received little attention but is directly relevant to industrial modeling practice.

Exploring the potential of LLMs in software language engineering. The findings of this study also provide inspiration for applying LLMs in software language engineering. Our prior work on metamodel-grammar co-evolution has shown that a rule-based approach can automate grammar adaptation after metamodel changes, but the adapted grammar still exhibits a gap relative to the target grammar [119][118]. One possible approach is to use LLMs to close this gap: given the grammar generated from the metamodel before evolution and the adapted version of this grammar, and also the grammar generated from the metamodel after evolution, an LLM could learn the adaptations in the previous version and replay it on the newly generated grammar, automatically obtaining a grammar that carries the same adaptations as the previous version while conforming to the evolved metamodel. We also see an opportunity in DSL-style customization. Our earlier work on converting Xtext-generated grammars to Python-style DSLs required semi-manual effort and was limited to a single target style and the metamodel-driven scenario [117]. An LLM-based approach could support both grammar-driven and metamodel-driven development, accepting natural-language descriptions of domain knowledge and style preferences to generate or adapt grammars for arbitrary target styles. We also intend to leverage LLMs to address challenges encountered during the development of a textual editor for EAST-ADL [32], i.e., schema version incompatibility and insufficiently informative error messages [116]. LLMs could contribute here by leveraging semantic understanding to bridge version differences and to generate context-sensitive diagnostic messages that guide users toward concrete corrections.

6 Related Work

6.1 Methodological Foundations for Systematic Studies

Our systematic mapping study follows the process defined by Petersen et al. [79], who proposed a five-step method for mapping studies in SE: defining research questions, conducting search, screening papers, keywording using abstracts, and data extraction. Their keywording technique, reading abstracts to identify concepts reflecting each paper's contribution and clustering them into a classification scheme, informed the spirit of our classification framework development, which proceeded through team discussion grounded in the research questions and was iteratively refined as primary studies were analyzed. We further adopted the updated guidelines by Petersen et al. [80], which consolidate lessons from a meta-study of existing mapping studies. In particular, we drew on their recommendations regarding snowballing as a complementary search strategy, multi-stage screening with explicit inclusion and exclusion criteria, and the use of visual matrix representations, including bubble plots and heatmaps, for presenting mapping results. The ACM SIGSOFT Empirical Standards [2] were consulted as a reference for reporting practices. Because our survey concerns the use of LLMs in an SE subdomain, characteristics of LLM-based research, closed-source models, non-deterministic outputs, and rapidly evolving model versions introduce validity concerns that conventional mapping study guidelines do not address. Several recent works have begun to examine these concerns, and they informed how we interpreted the primary studies in this mapping.

Sallou et al. [89] pointed out that LLM evaluation faces a more subtle problem, because models such as ChatGPT are trained on public code repositories and academic papers and may already have seen, or even memorized, the benchmarks used to test them. They cited evidence that LLM performance on well-known datasets (e.g., HumanEval, with $> 69\%$ branch coverage) can be drastically higher than on datasets absent from training corpora (e.g., SF110, at 2%). Beyond data leakage, they also discussed the lack of reproducibility caused by output variability and non-transparent model updates by providers, and proposed mitigation strategies such as metamorphic testing on inputs and mandatory metadata reporting.

Wagner et al. [108] extended this discussion at WSESE 2025 with a position paper arguing for dedicated evaluation guidelines, structured around baseline selection and configuration reporting. This effort was subsequently expanded by Baltes et al. [8] into a collaborative guideline document involving 22 researchers. Rather than listing all eight of their guidelines here, we note the aspects most relevant to our work: the requirement to report model versions and configurations, the recommendation to disclose prompts and interaction logs, and the emphasis on including open LLMs as baselines to support reproducibility. They also introduced a taxonomy of seven LLM-related study types (e.g., LLMs as annotators, as judges, as benchmarking targets), which helped us categorize how different primary studies in our mapping positioned their use of LLMs. Their observation that temperature-zero settings do not

guarantee deterministic outputs is a reminder that we should treat reproducibility claims in many of the papers we surveyed with caution.

6.2 Large Language Models in Software Engineering

LLMs have rapidly reshaped SE, enabling new capabilities across the entire development lifecycle, from requirements engineering to code generation, testing, maintenance, and system adaptation. Recent surveys and empirical studies provide a consolidated view of how LLMs are used in SE, emerging research trends, and open challenges. Overall, LLMs are becoming integral to SE practice and research. While code-related tasks currently dominate, applications in requirements engineering, architecture, testing, and system adaptation are accelerating. Nonetheless, fundamental challenges must be addressed before LLMs can be adopted in safety-critical or industrial-level settings.

6.2.1 Overview

A first wave of foundational surveys provides broad overviews of LLM applications in SE. Fan et al. present an early conceptual survey outlining the breadth of SE tasks impacted by LLMs and identifying hallucination and reliability as central challenges [35]. Hou et al. conduct the first large-scale Systematic Literature Review (SLR) of LLM4SE, analyzing 395 papers from 2017-2024 and providing a detailed taxonomy of models, tasks, prompting techniques, and evaluation methods [51]. Zhang et al. provide an extensive survey of code-focused LLMs, summarizing 62 code models and their use across 16 SE task categories [115]. Görmez et al. complement these findings with a systematic mapping study visualizing research themes and trends [46]. Collectively, these works show that while LLM4SE initially centered on code-related tasks, applications to requirements engineering, architecture, and testing have rapidly expanded. Nevertheless, a clear trend is the emergence of *code-specific* LLMs (e.g., Codex [19], CodeT5 [109], StarCoder [65], DeepSeek-Coder [48]), which significantly outperform general-purpose models in syntactic correctness, repair quality, and execution reliability [104].

Prompt engineering has become a first-class research area in SE. Sasaki et al. provide the first SLR on prompt engineering patterns for SE, identifying 21 reusable prompting strategies and showing their impact on downstream performance [93]. Prompt-based optimization (e.g., CoT, self-refinement, auto-prompting) is increasingly preferred over fine-tuning due to cost-efficiency and broader applicability.

Amalfitano et al. define a four-part classification of generative AI augmentation in SE (Copilot, GenAIware, Teammate, Robot), highlighting hybrid workflows where humans and LLMs collaborate [5]. The classification depends on what and how is the AI-based augmentation realized, integrating also the direction of SE4AI rather than focusing solely the AI4SE one. About the SE4AI topic, Scotti et al. propose an LLM Scripting Language (LSL) enabling structured, possibly verifiable interaction patterns for LLM-based tools [98]. Conversely, on the AI4SE topic, Li et al. outline how LLMs enhance monitoring, analysis, and planning in self-adaptive systems [64].

6.2.2 Applications Across the Software Engineering Lifecycle

LLM-assisted requirements engineering is one of the fastest-growing areas. Cheng et al. review 238 GenAI-for-RE studies, showing use cases in elicitation, classification, ambiguity detection, and validation [20]. Norheim et al. identify key challenges in applying LLMs to requirements tasks: correctness, verifiability, and domain adaptation [72]. Ferrari and Spoletini propose a two-way roadmap integrating formal methods with LLMs to improve correctness, fairness, and trustworthiness of generated requirements [36].

Research on LLMs for software architecture is emerging but growing rapidly. Schmid et al. provide the first dedicated SLR on LLMs in architecture engineering, identifying tasks such as design decision classification, pattern detection, and architecture generation from requirements [94]. Current approaches mainly rely on simple prompting, while structured multi-step reasoning and architectural consistency checks remain underexplored.

Code generation is the most mature and extensively studied area. Umre et al. show that code-specific LLMs outperform general-purpose models across code generation, translation, summarization, and repair tasks [104]. In program repair, multi-agent LLM processes and auto-prompting strategies have demonstrated strong performance. Moreover, LLMs proved useful even with underresourced languages.

In fact, Alhanahnah et al. show that dual-agent LLM systems outperform state-of-the-art techniques for repairing declarative Alloy specifications [4].

LLMs support numerous testing tasks, including unit test generation, assertion inference, regression test selection, vulnerability detection, and summarization of failure reports. Large surveys (e.g., Hou et al., Zhang et al.) highlight strong results in unit test generation and bug explanation but emphasize that reproducibility and evaluation quality remain major challenges [51, 115]. LLMs also facilitate software maintenance tasks such as code review, traceability link recovery, bug localization, and commit message generation.

6.2.3 Limitations and Research Challenges

All surveys identify *hallucinations* [12] as the primary obstacle to trustworthy LLM-based SE. Generated artifacts often appear syntactically correct, but embed subtle semantic errors, making them difficult to detect without verification. General mitigation techniques include retrieval-augmented generation, grammar-constrained decoding, stepwise prompting, and post-hoc verification using static analysis or automated testing.

Reproducibility constitutes a systemic crisis in LLM4SE research. Siddiq et al. analyze 640 studies and document widespread reproducibility smells, including missing artifacts, poorly specified environments, opaque prompts, absent datasets, and reliance on closed models [100]. Artifact badges correlate only weakly with actual reproducibility quality.

The absence of standardized evaluation frameworks hinders comparability across studies. Benchmarks rapidly become outdated, and many datasets are proprietary, small, or unrepresentative. In general, surveys call for transparent prompt reporting, shared datasets, open evaluation processes, and life-cycle-aware benchmarks that reflect real-world SE constraints.

6.3 Large Language Models in Model-Driven Engineering

As a subfield of SE, MDE has similarly seen growing interest in LLM applications. As noted in the introduction, Di Rocco et al. [29] present a systematic literature review of 14 primary studies (2019–mid-2024), examining LLM applications in tasks such as model completion, model generation, OCL constraint generation, and DSL specification, and proposing a research agenda addressing hallucination mitigation, benchmarking, and scalability. Burgueño et al. [13] take a broader automation perspective, surveying enabling technologies across formal methods, search-based techniques, knowledge engineering, and artificial intelligence, and noting that LLMs have partially alleviated the persistent data scarcity challenge in MDE. While these works provide complementary technical and automation-oriented perspectives on LLM4MDE, our mapping study aims at a broader classification of the research landscape, with particular attention to how LLMs are configured and orchestrated in practice, including model selection, access modes, retrieval augmentation, execution structures, and the representation of input and output artifacts, aspects that neither of these prior works covers systematically.

7 Conclusion

This paper provides a systematic mapping study of LLM applications in MDE, analyzing 86 primary research papers published between 2022 and early 2026. Guided by five research questions, we characterized the research landscape from dimensions such as MDE task distribution, LLM techniques and interaction strategies, artifact representation, validation practices, and publication patterns. The results show that while the field is developing rapidly, it remains highly focused on model generation, primarily relying on black-box OpenAI models accessed via remote APIs and adapted through prompt engineering. Furthermore, the generated model-oriented outputs are more often represented in lightweight text formats rather than MDE's native exchange formats. Validation mainly focuses on quantitative experiments, lacking baseline comparisons and reproducibility support. Most methods remain standalone prototypes with limited integration with existing MDE toolchains.

Several directions warrant attention in future work. Although task types such as model migration, DSL engineering, and meta-modeling are closely related to the MDE workflows, their research remains

unexplored. Agentic system design and deep integration with existing MDE toolchains represent untapped opportunities. In terms of evaluation, the broader adoption of MDE-specific quality standards, the implementation of systematic baseline comparisons including open-source LLMs, and more standardized cost reporting will improve the comparability and rigor of future studies. The two-tiered evaluation framework proposed in Section 5.1 provides a starting point for these efforts.

Declarations

Fundings

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - SFB 1608 - 501798263 and supported by the pilot program Core Informatics at KIT (KiKIT) of the Helmholtz Association (HGF). It was also funded by Vetenskapsrådet, grant 2021-04881_VR.

Ethical Approval

Not applicable

Informed Consent

Not applicable

Data Availability Statement

Our code and data generated for this article are made available at `https://osf.io/g5by9/overview?view_only=5c10c1e56be3480d8d25e017b4276f7a` for reference.

Conflict of Interest

The authors have no competing interests to declare that are relevant to the content of this article.

Author Contributions

Hereafter, we report the author contributions following the CRediT taxonomy[1].

- Conceptualization: Weixing Zhang, Regina Hebig, Daniel Strüber, Anne Koziolek
- Data curation: Weixing Zhang, Bowen Jiang, Yuhong Fu, Haowei Cheng, Maximilian Hummel, Vincenzo Scotti, Nathan Hagel, Jialong Li, Regina Hebig, Daniel Strüber, Anne Koziolek.
- Investigation: Weixing Zhang, Bowen Jiang, Yuhong Fu, Haowei Cheng, Maximilian Hummel, Vincenzo Scotti, Nathan Hagel, Jialong Li, Regina Hebig, Daniel Strüber, Anne Koziolek.
- Methodology: Weixing Zhang, Regina Hebig, Daniel Strüber, Anne Koziolek, Vincenzo Scotti.
- Project administration: Weixing Zhang, Anne Koziolek
- Software: Weixing Zhang, Bowen Jiang, Yuhong Fu, Vincenzo Scotti.
- Supervision: Weixing Zhang, Georg Grossmann, Markus Stumptner, Regina Hebig, Daniel Strüber, Anne Koziolek.
- Validation: Weixing Zhang, Bowen Jiang, Yuhong Fu, Haowei Cheng, Maximilian Hummel, Vincenzo Scotti, Nathan Hagel, Jialong Li, Regina Hebig, Daniel Strüber, Anne Koziolek.
- Visualization: Weixing Zhang, Bowen Jiang, Yuhong Fu, Vincenzo Scotti.
- Writing – original draft: Weixing Zhang, Bowen Jiang, Yuhong Fu, Haowei Cheng, Vincenzo Scotti, Daniel Strüber.

[1] https://credit.niso.org/

– Writing – review & editing: Weixing Zhang, Bowen Jiang, Yuhong Fu, Haowei Cheng, Maximilian Hummel, Vincenzo Scotti, Jialong Li, Georg Grossmann, Markus Stumptner, Regina Hebig, Daniel Strüber, Anne Koziolek.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023) Gpt-4 technical report. arXiv preprint arXiv:230308774
2. ACM SIGSOFT (2021) Empirical Standards for Software Engineering. https://www2.sigsoft.org/EmpiricalStandards/, accessed: 2025-09-19
3. Ahmed, K., Song, J., Chen, B., Wei, O., Zheng, B. (2025) Mcet: Behavioral model correctness evaluation using large language models. In: 2025 ACM/IEEE 28th International Conference on Model Driven Engineering Languages and Systems (MODELS), IEEE, pp 84–95
4. Alhanahnah, M., Rashedul Hasan, M., Xu, L., Bagheri, H. (2025) An empirical evaluation of pre-trained large language models for repairing declarative formal specifications. Empirical Software Engineering 30(5):149, DOI 10.1007/s10664-025-10687-1, URL https://doi.org/10.1007/s10664-025-10687-1
5. Amalfitano, D., Metzger, A., Autili, M., Fulcini, T., Hey, T., Keim, J., Pelliccione, P., Scotti, V., Koziolek, A., Mirandola, R., et al. (2026) A research roadmap for augmenting software engineering processes and software products with generative ai. ACM Transactions on Software Engineering and Methodology
6. Anthropic (2023) Introducing Claude. URL https://www.anthropic.com/news/introducing-claude
7. Atkinson, C., Kühne, T. (2001) The essence of multilevel metamodeling. In: International Conference on the Unified Modeling Language, Springer, pp 19–33
8. Baltes, S., Angermeir, F., Arora, C., Muñoz Barón, M., Chen, C., Böhme, L., Calefato, F., Ernst, N., Falessi, D., Fitzgerald, B., et al. (2025) Guidelines for empirical studies in software engineering involving large language models. arXiv e-prints pp arXiv–2508
9. Bamouh, L., Béziers La Fosse, T., Tisi, M. (2025) Towards diagram-based data model generation with llms. In: International Conference on Conceptual Modeling, Springer, pp 157–175
10. 'Alvarez Bornstein, B., Montesi, M. (2020) Funding acknowledgements in scientific publications: A literature review. Research Evaluation 29(4):469–488, DOI 10.1093/reseval/rvaa038
11. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D. et al. (2020) Language models are few-shot learners. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds) Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual, URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html
12. Brunello, N. (2026) 6 - trustworthiness of large language models: hallucinations. In: Pillai, A.S., Tedesco, R., Scotti, V. (eds) Challenges and Applications of Generative Large Language Models, Morgan Kaufmann, pp 107–126, DOI https://doi.org/10.1016/B978-0-443-33592-1.00007-3, URL https://www.sciencedirect.com/science/article/pii/B9780443335921000073
13. Burgueño, L., Di Ruscio, D., Sahraoui, H., Wimmer, M. (2025) Automation in model-driven engineering: A look back, and ahead. ACM Transactions on Software Engineering and Methodology 34(5):1–25
14. Cabot, J., Gogolla, M. (2012) Object constraint language (ocl): a definitive guide. In: International school on formal methods for the design of computer, communication and software systems, Springer, pp 58–90
15. Cámara, J., Troya, J., Burgueño, L., Vallecillo, A. (2023) On the assessment of generative ai in modeling tasks: an experience report with chatgpt and uml: Jj. cámara et al. Software and Systems Modeling 22(3):781–793

16. Chen, B., Chen, K., Hassani, S., Yang, Y., Amyot, D., Lessard, L., Mussbacher, G., Sabetzadeh, M., Varró, D. (2023) On the use of gpt-4 for creating goal models: an exploratory study. In: 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW), IEEE, pp 262–271
17. Chen, L., Guo, Q., Jia, H., Zeng, Z., Wang, X., Xu, Y., Wu, J., Wang, Y., Gao, Q., Wang, J., et al. (2024) A survey on evaluating large language models in code generation tasks. arXiv preprint arXiv:240816498
18. Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H.P.D.O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. (2021) Evaluating large language models trained on code. arXiv preprint arXiv:210703374
19. Chen, M., Tworek, J., Jun, H., Yuan, Q., et al. (2021) Evaluating large language models trained on code. arXiv preprint arXiv:210703374
20. Cheng, H., Husen, J.H., Lu, Y., Racharak, T., Yoshioka, N., Ubayashi, N., Washizaki, H. (2026) Generative ai for requirements engineering: A systematic literature review. Software: Practice and Experience 56(2):141–170
21. Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S.S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Narang, S., Mishra, G., Yu, A., andf Yanping Huang, V.Y.Z., Dai, A.M., Yu, H., Petrov, S., Chi, E.H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q.V., Wei, J. (2022) Scaling instruction-finetuned language models. CoRR abs/2210.11416, DOI 10.48550/ARXIV.2210.11416, URL https://doi.org/10.48550/arXiv.2210.11416, 2210.11416
22. Clarivate (2025) Journal citation reports. https://clarivate.com/academia-government/scientific-and-academic-research/research-funding-analytics/journal-citation-reports/
23. Cohen, J. (1960) A coefficient of agreement for nominal scales. Educational and psychological measurement 20(1):37–46
24. Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., Ram, O., Zhang, D., Rosen, E., Marris, L., Petulla, S., Gaffney, C., Aharoni, A., Lintz, N., Pais, T.C., Jacobsson, H., Szpektor, I., Jiang, N.J., Haridasan, K., Omran, A., Saunshi, N., Bahri, D., Mishra, G., Chu, E., Boyd, T., Hekman, B., Parisi, A., Zhang, C., Kawintiranon, K., Bedrax-Weiss, T. et al. (2025) Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. URL https://arxiv.org/abs/2507.06261, 2507.06261
25. Computing Research and Education Association of Australasia (2026) Icore2026 conference rankings via the icore conference portal. https://portal.core.edu.au/conf-ranks/
26. Czarnecki, K., Helsen, S. (2006) Feature-based survey of model transformation approaches. IBM systems journal 45(3):621–645
27. Devlin, J., Chang, M., Lee, K., Toutanova, K. (2019) BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), Association for Computational Linguistics, pp 4171–4186, DOI 10.18653/V1/N19-1423, URL https://doi.org/10.18653/v1/n19-1423
28. Di Rocco, J., Iovino, L., Pierantonio, A. (2012) Bridging state-based differencing and co-evolution. In: Proceedings of the 6th International Workshop on Models and Evolution, pp 15–20
29. Di Rocco, J., Di Ruscio, D., Di Sipio, C., Nguyen, P.T., Rubei, R. (2025) On the use of large language models in model-driven engineering: J. di rocco et al. Software and Systems Modeling 24(3):923–948
30. Donthu, N., Kumar, S., Mukherjee, D., Pandey, N., Lim, W.M. (2021) How to conduct a bibliometric analysis: An overview and guidelines. Journal of business research 133:285–296
31. Dvivedi, S.S., Vijay, V., Pujari, S.L.R., Lodh, S., Kumar, D. (2024) A comparative analysis of large language models for code documentation generation. In: Proceedings of the 1st ACM international conference on AI-powered software, pp 65–73
32. EAST-ADL Association (2021) East-adl. URL https://www.east-adl.info/, Accessed February, 2023
33. Ehrig, K., Küster, J.M., Taentzer, G. (2009) Generating instance models from meta models. Software & Systems Modeling 8(4):479–500

34. Eysholdt, M., Behrens, H. (2010) Xtext: Implement your language faster than the quick and dirty way. In: Proceedings of the ACM International Conference Companion on Object Oriented Programming Systems Languages and Applications, ACM, OOPSLA '10, pp 307–309, DOI 10.1145/1869542.1869625
35. Fan, A., Gokkaya, B., Harman, M., Lyubarskiy, M., Sengupta, S., Yoo, S., Zhang, J.M. (2023) Large language models for software engineering: Survey and open problems. In: 2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE), IEEE, pp 31–53
36. Ferrari, A., Spoletini, P. (2025) Formal requirements engineering and large language models: A two-way roadmap. Information and Software Technology 181:107697, DOI https://doi.org/10.1016/j.infsof.2025.107697, URL https://www.sciencedirect.com/science/article/pii/S0950584925000369
37. Fleurey, F., Steel, J., Baudry, B. (2004) Validation in model-driven engineering: testing model transformations. In: Proceedings. 2004 First International Workshop on Model, Design and Validation, 2004., IEEE, pp 29–40
38. Fowler, M. (2010) Domain-Specific Languages, Portable Documents. Pearson Education
39. Fu, Y., Grossmann, G., Kaur, K., Selway, M., Stumptner, M. (2022) Multi-level risk modelling for interoperability of risk information. In: IN4PL, pp 242–249
40. Fu, Y., Grossmann, G., Kaur, K., Selway, M., Stumptner, M. (2023) Towards the integration of multi-level and multi-view modelling for interoperability. In: 2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), IEEE, pp 679–688
41. Fu, Y., Selway, M., Grossmann, G., Kaur, K., Stumptner, M. (2024) Modelling a warehouse with slicer: A contribution to the multi warehouse challenge. In: Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, pp 828–837
42. Fu, Y., Grossmann, G., Kaur, K., Selway, M., Stumptner, M. (2025) Conflict management for multi-level models in collaborative modelling environments. In: 2025 ACM/IEEE 28th International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), IEEE, pp 502–511
43. Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Guo, Q., Wang, M., Wang, H. (2023) Retrieval-augmented generation for large language models: A survey. CoRR abs/2312.10997, DOI 10.48550/ARXIV.2312.10997, URL https://doi.org/10.48550/arXiv.2312.10997, 2312.10997
44. Giraldo, F.D., Espana, S., Pastor, O., Giraldo, W.J. (2018) Considerations about quality in model-driven engineering: Current state and challenges. Software Quality Journal 26(2):685–750
45. GitHub (2021) Introducing github copilot: your ai pair programmer. URL https://github.blog/news-insights/product-news/introducing-github-copilot-ai-pair-programmer/
46. Görmez, M.K., Yılmaz, M., Clarke, P.M. (2024) Large language models for software engineering: A systematic mapping study. In: Yilmaz, M., Clarke, P., Riel, A., Messnarz, R., Greiner, C., Peisl, T. (eds) Systems, Software and Services Process Improvement, Springer Nature Switzerland, Cham, pp 64–79
47. Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., Rao, A., Zhang, A., Rodriguez, A., Gregerson, A., Spataru, A., Rozière, B., Biron, B., Tang, B., Chern, B., Caucheteux, C., Nayak, C., Bi, C., Marra, C. et al. (2024) The llama 3 herd of models. CoRR abs/2407.21783, DOI 10.48550/ARXIV.2407.21783, URL https://doi.org/10.48550/arXiv.2407.21783, 2407.21783
48. Guo, D., Zhu, Q., Yang, D., Xie, Z., Dong, K., Zhang, W., Chen, G., Bi, X., Wu, Y., Li, Y.K., et al. (2024) DeepSeek-Coder: When the large language model meets programming – the rise of code intelligence. arXiv preprint arXiv:240114196
49. Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z.F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., Xue, B., Wang, B., Wu, B., Feng, B., Lu, C., Zhao, C., Deng, C., Ruan, C., Dai, D., Chen, D., Ji, D., Li, E. et al. (2025) Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. Nature 645(8081):633–638, DOI 10.1038/s41586-025-09422-z, URL https://doi.org/10.1038/s41586-025-09422-z

50. Hachm, Z., Le Calvar, T., Bruneliere, H., Tisi, M. (2025) Towards llm agents for model-based engineering: A case in transformation selection. In: 2025 ACM/IEEE 28th International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), IEEE, pp 421–431
51. Hou, X., Zhao, Y., Liu, Y., Yang, Z., Wang, K., Li, L., Luo, X., Lo, D., Grundy, J., Wang, H. (2024) Large language models for software engineering: A systematic literature review. ACM Transactions on Software Engineering and Methodology 33(8):1–79
52. Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., et al. (2025) A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. ACM Transactions on Information Systems 43(2):1–55
53. Hummel, P., Grimstad, J., Xia, Y., Morozov, A. (2024) Generation of matlab simscape models using large language models for training reliability evaluation methods. In: ASME International Mechanical Engineering Congress and Exposition, American Society of Mechanical Engineers, vol 88698, p V011T14A010
54. Hutchinson, J., Whittle, J., Rouncefield, M., Kristoffersen, S. (2011) Empirical assessment of mde in industry. In: Proceedings of the 33rd international conference on software engineering, pp 471–480
55. ISO (2024) ISO/TR 8344:2024(en) Information and documentation — Issues and considerations for managing records in structured data environments. https://www.iso.org/obp/ui/es/#iso:std:iso:tr:8344:ed-1:v1:en, accessed: 2026-03-25
56. Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., de Las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L.R., Lachaux, M., Stock, P., Scao, T.L., Lavril, T., Wang, T., Lacroix, T., Sayed, W.E. (2023) Mistral 7b. CoRR abs/2310.06825, DOI 10.48550/ARXIV.2310.06825, URL https://doi.org/10.48550/arXiv.2310.06825, 2310.06825
57. Jouault, F., Allilaire, F., Bézivin, J., Kurtev, I. (2008) ATL: A model transformation tool. Science of Computer Programming 72(1–2):31–39, DOI 10.1016/j.scico.2007.08.002
58. Kitchenham, B., Brereton, P. (2013) A systematic review of systematic review process research in software engineering. Information and software technology 55(12):2049–2075
59. Kitchenham, B.A. (1996) Evaluating software engineering methods and tool part 1: The evaluation context and evaluation methods. ACM SIGSOFT Software Engineering Notes 21(1):11–14
60. Kojima, T., Gu, S.S., Reid, M., Matsuo, Y., Iwasawa, Y. (2022) Large language models are zero-shot reasoners. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds) Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022, URL http://papers.nips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html
61. Kong, Y., Zhang, N., Duan, Z., Yu, B. (2025) Collaboration with generative ai to improve requirements change. Computer Standards & Interfaces 94:104013
62. Lamas, V., Garcia-Gonzalez, D., Sala, L., Luaces, M.R. (2025) Dsl-xpert 2.0: Enhancing llm-driven code generation for domain-specific languages. Information and Software Technology p 107954
63. Van der Lee, C., Gatt, A., Van Miltenburg, E., Krahmer, E. (2021) Human evaluation of automatically generated text: Current trends and best practice guidelines. Computer Speech & Language 67:101151
64. Li, J., Zhang, M., Li, N., Weyns, D., Jin, Z., Tei, K. (2024) Generative ai for self-adaptive systems: State of the art and research roadmap. ACM Trans Auton Adapt Syst 19(3), DOI 10.1145/3686803, URL https://doi.org/10.1145/3686803
65. Li, R., Ben Allal, L., Zi, Y., Muennighoff, N., et al. (2023) StarCoder: may the source be with you! Transactions on Machine Learning Research
66. Liu, Y., Zhou, H., Guo, Z., Shareghi, E., Vulić, I., Korhonen, A., Collier, N. (2024) Aligning with human judgement: The role of pairwise preference in large language model evaluators. arXiv preprint arXiv:240316950
67. Macedo, N., Jorge, T., Cunha, A. (2016) A feature-based classification of model repair approaches. IEEE Transactions on Software Engineering 43(7):615–640
68. McHugh, M.L. (2012) Interrater reliability: the kappa statistic. Biochemia medica 22(3):276–282
69. Mesnard, T., Hardin, C., Dadashi, R., Bhupatiraju, S., Pathak, S., Sifre, L., Rivière, M., Kale, M.S., Love, J., Tafti, P., Hussenot, L., Chowdhery, A., Roberts, A., Barua, A., Botev, A., Castro-Ros, A., Slone, A., Héliou, A., Tacchetti, A., Bulanova, A., Paterson, A., Tsai, B., Shahriari, B., Lan, C.L., Choquette-Choo, C.A., Crepy, C., Cer, D., Ippolito, D., Reid, D., Buchatskaya, E., Ni, E. et al.

(2024) Gemma: Open models based on gemini research and technology. CoRR abs/2403.08295, DOI 10.48550/ARXIV.2403.08295, URL https://doi.org/10.48550/arXiv.2403.08295, 2403.08295
70. Moezkarimi, Z., Eriksson, K., Johansson, A.A., Bucaioni, A., Sirjani, M. (2025) Harnessing chatgpt for model transformation in software architecture: From uml state diagrams to rebeca models for formal verification. In: 2025 IEEE 22nd International Conference on Software Architecture Companion (ICSA-C), IEEE, pp 387–396
71. Mohagheghi, P., Fernandez, M.A., Martell, J.A., Fritzsche, M., Gilani, W. (2008) Mde adoption in industry: challenges and success criteria. In: International Conference on Model Driven Engineering Languages and Systems, Springer, pp 54–59
72. Norheim, J.J., Rebentisch, E., Xiao, D., Draeger, L., Kerbrat, A., de Weck, O.L. (2024) Challenges in applying large language models to requirements engineering tasks. Design Science 10:e16, DOI 10.1017/dsj.2024.8
73. Object Management Group (2019) OMG Systems Modeling Language (SysML), Version 1.6. Tech. Rep. formal/2019-11-01, Object Management Group, URL https://www.omg.org/spec/SysML/1.6/
74. OECD, Eurostat (2018) Oslo Manual 2018: Guidelines for Collecting, Reporting and Using Data on Innovation, 4th Edition. The Measurement of Scientific, Technological and Innovation Activities 2018, DOI 10.1787/9789264304604-en
75. OpenAI (2022) Introducing ChatGPT. URL https://openai.com/index/chatgpt/
76. OpenAI (2026) Openai GPT-5 system card. CoRR abs/2601.03267, DOI 10.48550/ARXIV.2601.03267, URL https://doi.org/10.48550/arXiv.2601.03267, 2601.03267
77. Paige, R.F., Brooke, P.J., Ostroff, J.S. (2007) Metamodel-based model conformance and multiview consistency checking. ACM Transactions on Software Engineering and Methodology (TOSEM) 16(3):11–es
78. Panach, J.I., Dieste, O., Marín, B., España, S., Vegas, S., Pastor, O., Juristo, N. (2018) Evaluating model-driven development claims with respect to quality: A family of experiments. IEEE Transactions on Software Engineering 47(1):130–145
79. Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M. (2008) Systematic mapping studies in software engineering. In: 12th international conference on evaluation and assessment in software engineering (EASE), BCS Learning & Development
80. Petersen, K., Vakkalanka, S., Kuzniarz, L. (2015) Guidelines for conducting systematic mapping studies in software engineering: An update. Information and software technology 64:1–18
81. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I. (2018) Improving language understanding by generative pre-training. URL https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf
82. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J. (2020) Exploring the limits of transfer learning with a unified text-to-text transformer. J Mach Learn Res 21:140:1–140:67, URL https://jmlr.org/papers/v21/20-074.html
83. Ratiu, D., Voelter, M. (2016) Automated testing of dsl implementations: experiences from building mbeddr. In: Proceedings of the 11th International Workshop on Automation of Software Test, pp 15–21
84. Reineke, J., Tripakis, S. (2014) Basic problems in multi-view modeling. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Springer, pp 217–232
85. Ren, S., Guo, D., Lu, S., Zhou, L., Liu, S., Tang, D., Sundaresan, N., Zhou, M., Blanco, A., Ma, S. (2020) Codebleu: a method for automatic evaluation of code synthesis. arXiv preprint arXiv:200910297
86. Rizzi, S., Francia, M., Gallinucci, E., Golfarelli, M. (2025) Conceptual design of multidimensional cubes with llms: An investigation. Data & Knowledge Engineering 159:102452
87. Rose, L.M., Paige, R.F., Kolovos, D.S., Polack, F.A. (2009) An analysis of approaches to model migration. In: Proc. Joint MoDSE-MCCM Workshop, pp 6–15
88. Rozière, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X.E., Adi, Y., Liu, J., Remez, T., Rapin, J., Kozhevnikov, A., Evtimov, I., Bitton, J., Bhatt, M., Canton-Ferrer, C., Grattafiori, A., Xiong, W., Défossez, A., Copet, J., Azhar, F., Touvron, H., Martin, L., Usunier, N., Scialom, T., Synnaeve, G. (2023) Code llama: Open foundation models for code. CoRR abs/2308.12950, DOI 10.48550/ARXIV.2308.12950, URL https://doi.org/10.48550/arXiv.2308.12950, 2308.12950

89. Sallou, J., Durieux, T., Panichella, A. (2024) Breaking the silence: the threats of using llms in software engineering. In: Proceedings of the 2024 ACM/IEEE 44th International Conference on Software Engineering: New Ideas and Emerging Results, pp 102–106
90. Sami, A.M., Rasheed, Z., Waseem, M., Zhang, Z., Tomas, H., Abrahamsson, P. (2024) A tool for test case scenarios generation using large language models. arXiv preprint arXiv:240607021
91. Sanh, V., Webson, A., Raffel, C., Bach, S.H., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Raja, A., Dey, M., Bari, M.S., Xu, C., Thakker, U., Sharma, S.S., Szczechla, E., Kim, T., Chhablani, G., Nayak, N.V., Datta, D., Chang, J., Jiang, M.T., Wang, H., Manica, M., Shen, S., Yong, Z.X., Pandey, H., Bawden, R., Wang, T., Neeraj, T., Rozen, J., Sharma, A. et al. (2022) Multitask prompted training enables zero-shot task generalization. In: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022, OpenReview.net, URL https://openreview.net/forum?id=9Vrb9D0WI4
92. Santiago, I., Jiménez, A., Vara, J.M., De Castro, V., Bollati, V.A., Marcos, E. (2012) Model-driven engineering as a new landscape for traceability management: A systematic literature review. Information and Software Technology 54(12):1340–1356
93. Sasaki, Y., Washizaki, H., Li, J., Sander, D., Yoshioka, N., Fukazawa, Y. (2024) Systematic literature review of prompt engineering patterns in software engineering. In: 2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC), IEEE, pp 670–675
94. Schmid, L., Hey, T., Armbruster, M., Corallo, S., Fuchß, D., Keim, J., Liu, H., Koziolek, A. (2025) Software architecture meets llms: A systematic literature review. URL https://arxiv.org/abs/2505.16697, 2505.16697
95. Schmidt, D.C., et al. (2006) Model-driven engineering. Computer-IEEE Computer Society- 39(2):25
96. Schulhoff, S., Ilie, M., Balepur, N., Kahadze, K., Liu, A., Si, C., Li, Y., Gupta, A., Han, H., Schulhoff, S., et al. (2024) The prompt report: A systematic survey of prompt engineering techniques. arXiv preprint arXiv:240606608
97. Scotti, V., Sbattella, L., Tedesco, R. (2024) A primer on seq2seq models for generative chatbots. ACM Comput Surv 56(3):75:1–75:58, DOI 10.1145/3604281, URL https://doi.org/10.1145/3604281
98. Scotti, V., Keim, J., Hey, T., Metzger, A., Koziolek, A., Mirandola, R. (2025) A roadmap for tamed interactions with large language models. arXiv preprint arXiv:251024819
99. Shaw, M. (2003) Writing good software engineering research papers. In: 25th International Conference on Software Engineering, 2003. Proceedings., IEEE, pp 726–736
100. Siddiq, M.L., Islam-Gomes, A., Sekerak, N., Santos, J.C.S. (2025) Large language models for software engineering: A reproducibility crisis. URL https://arxiv.org/abs/2512.00651, 2512.00651
101. Silva, J., Ma, Q., Cabot, J., Kelsen, P., Proper, H.A. (2025) Towards human-in-the-loop llm-enabled domain modeling. In: International Conference on Conceptual Modeling, Springer, pp 127–145
102. Steinberg, D., Budinsky, F., Paternostro, M., Merks, E. (2008) EMF: Eclipse Modeling Framework, 2nd edn. Addison-Wesley Professional
103. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023) Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:230709288
104. Umre, J., Parihar, A.S., Gupta, A. (2026) Csllm: Code-specific large language models—a survey. Expert Systems with Applications 308:130991, DOI https://doi.org/10.1016/j.eswa.2025.130991, URL https://www.sciencedirect.com/science/article/pii/S0957417425046068
105. Van Der Straeten, R., Mens, T., Van Baelen, S. (2008) Challenges in model-driven software engineering. In: International conference on model driven engineering languages and systems, Springer, pp 35–47
106. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I. (2017) Attention is all you need. In: Guyon, I., von Luxburg, U., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R. (eds) Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pp 5998–6008, URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
107. Viyović, V., Maksimović, M., Perisić, B. (2014) Sirius: A rapid development of DSM graphical editor. In: IEEE 18th International Conference on Intelligent Engineering Systems (INES), pp 233–238, DOI 10.1109/INES.2014.6909375

108. Wagner, S., Barón, M.M., Falessi, D., Baltes, S. (2025) Towards evaluation guidelines for empirical studies involving llms. In: 2025 IEEE/ACM International Workshop on Methodological Issues with Empirical Studies in Software Engineering (WSESE), IEEE, pp 24–27
109. Wang, Y., Wang, W., Joty, S., Hoi, S.C.H. (2021) CodeT5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp 8696–8708, DOI 10.18653/v1/2021.emnlp-main.685
110. Washizaki, H. (2024) Guide to the software engineering body of knowledge v4.0a. IEEE Computer Society
111. Weber, T., Brandmaier, M., Schmidt, A., Mayer, S. (2024) Significant productivity gains through programming with large language models. Proceedings of the ACM on Human-Computer Interaction 8(EICS):1–29
112. Weixing Zhang, Bowen Jiang, Yuhong Fu, Haowei Cheng, Maximiliian Hummel, Vincenzo Scotti, Nathan Josias Hagel, Jialong Li, Georg Grossmann, Markus Stumptner, Regina Hebig, Daniel Strüber, Anne Koziolek (2026) Replication Package. `https://osf.io/g5by9/overview?view_only=5c10c1e56be3480d8d25e017b4276f7a`, accessed: 2026-04-10
113. Wohlin, C. (2014) 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, pp 1–10
114. Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L. et al. (2025) Qwen3 technical report. URL `https://arxiv.org/abs/2505.09388`, `2505.09388`
115. Zhang, Q., Fang, C., Xie, Y., Zhang, Y., Yu, S., Sun, W., Yang, Y., Chen, Z. (2026) A survey on large language models for software engineering. Science China Information Sciences 69(4):141102, DOI 10.1007/s11432-025-4670-0, URL `https://doi.org/10.1007/s11432-025-4670-0`
116. Zhang, W., Holtmann, J. (2023) Technical report: Unresolved challenges and potential features in eatxt. arXiv preprint arXiv:231210250
117. Zhang, W., Hebig, R., Steghöfer, J.P., Holtmann, J. (2023) Creating python-style domain specific languages: A semi-automated approach and intermediate results. In: MODELSWARD, pp 210–217
118. Zhang, W., Hebig, R., Strüber, D., Steghöfer, J.P. (2023) Automated extraction of grammar optimization rule configurations for metamodel-grammar co-evolution. In: Proceedings of the 16th ACM SIGPLAN International Conference on Software Language Engineering, pp 84–96
119. Zhang, W., Holtmann, J., Strüber, D., Hebig, R., Steghöfer, J.P. (2024) Supporting meta-model-based language evolution and rapid prototyping with automated grammar transformation. Journal of Systems and Software 214:112069
120. Zhang, W., Hebig, R., Strüber, D. (2025) Leveraging llms to support co-evolution between definitions and instances of textual dsls. In: Proceedings of the First Large Language Models for Software Engineering Workshop (LLM4SE), co-located with STAF 2025, Koblenz, Germany
121. Zhang, W., Jiang, B., Fu, Y., Koziolek, A., Hebig, R., Strüber, D. (2026) Leveraging llms to support co-evolution between definitions and instances of textual dsls: A systematic evaluation. arXiv preprint arXiv:260211904
122. Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., et al. (2023) A survey of large language models. arXiv preprint arXiv:230318223 1(2):1–124