Accepted for EMMSAD'2026 and publication in Springer LNBIP.
This is the camera-ready author version of the paper, the final version will be made accessible via Springer.

Uncovering LLM's Capabilities in Model-based Question Answering for UML Class Diagrams

Manuel Mischak[1], Charlotte Verbruggen[1][0000−0003−0418−2633], Philip Langer[2], and Dominik Bork[1][0000−0001−8259−2297]

[1] Business Informatics Group, TU Wien, Austria manuel.mischak1@gmail.com, charlotte.verbruggen@tuwien.ac.at, dominik.bork@tuwien.ac.at
[2] EclipseSource Services GmbH, Vienna, Austria planger@eclipsesource.com

Abstract. Conceptual models, such as UML diagrams, play a central role in abstracting and communicating the complexity of information systems. However, their interpretation remains challenging — particularly for non-experts and users with accessibility needs. This work investigates whether large language models (LLMs) can effectively assist users by answering comprehension questions. We analyze existing approaches to model comprehension and conversational assistance, and identify their strengths and limitations in handling diagrammatic information. Building on these insights, we design a prototype that integrates an LLM-based conversational interaction in the GLSP-based BIGUML editor. Using this prototype and a set of reported UML model comprehension questions, we uncover the influence of representation format, context size, and LLM provider on the extent to which LLMs can accurately answer questions about UML class diagrams. Our study offers unique insights into LLM capabilities for model-based question answering in UML class diagrams, a prerequisite for advancing accessible, AI-assisted modeling.

Keywords: Conceptual modeling · Model comprehension · LLM · Accessibility.

1 Introduction

Conceptual models effectively convey complex information. They are used in many forms and sizes for countless applications. However, people with visual impairments or those who struggle to understand large models at first glance face significant barriers in accessing and understanding this information [41]. Individuals who are not regularly working with large models or encounter them for the first time need significant time to understand the connections and clusters within them, which creates a need for an effective solution [17].

The relevance of this problem is underlined by the growing emphasis on inclusive design and accessibility in technology in general and conceptual modeling [19, 34], software engineering, and model-driven engineering [3, 25] in particular. By improving the accessibility of models, crucial information becomes

available to a broader audience [34], ultimately leading to more accessible software through increased diversity (i.e., inclusion) among stakeholders. In addition, improving the accessibility of model content supports model comprehension for all individuals who want to quickly understand a large model.

This research advances the accessibility of conceptual models by exploring the feasibility of integrating conversational functionality powered by Large Language Models (LLMs) into modeling tools, with the goal of making conceptual modeling more inclusive and accessible [3]. The particular task we investigate in this paper is model-based question answering. Improving the accessibility of such tools should be achieved without compromising on the accuracy of the answers. Therefore, this research aims to respond to the following research questions: **RQ1:** What is the state of the art in model comprehension and conversational agent techniques in conceptual modeling, and what are the agents' strengths and limitations? **RQ2:** To what extent do LLMs accurately answer comprehension questions on conceptual models?

In the remainder of this paper, Section 2 provides a comprehensive review of model summarization and conversational agents in conceptual modeling. Section 3 subsequently reports on the design and implementation of a prototype modeling tool enhanced with a conversational agent that enables model-based answering of comprehension questions. This tool supports an experiment evaluating LLM capabilities in model-based question answering. The experimental design and the results are presented in Section 4. Finally, we discuss the key findings and implications of our work in Section 5 before concluding in Section 6.

2 Related Work: Model Summarization and Conversational Agents

This section provides a summary of the comprehensive literature review on model summarization and conversational agents; all details are provided in the online supplementary material to this paper[1][28]. **Search and Screening Process.** To enable comprehensive literature coverage, three interrelated queries were formulated, each targeting model-related research in one of the following domains: Q_1: Model Accessibility, Q_2: Model Comprehension and Simplification, and Q_3: Conversational Chat Bots and Interfaces. The search queries were executed in February 2025 at the scientific databases Scopus, IEEE Xplore, and ACM Digital Library. To ensure homogeneity and quality in the results, papers were excluded if they were *i*) written before 2010; *ii*) less than four pages long; *iii*) not written in English; *iv*) not related to our research scope; *v*) not accessible as full text; or *vi*) non-scientific papers (e.g., tutorials, editorials).

Fig. 1 shows the screening process followed. Of the publications that passed the abstract-level screening, 99 proceeded to a full-text evaluation, where their content was critically assessed to confirm methodological soundness and thematic suitability, resulting in a final selection of 51 papers.

[1] `https://github.com/Charlotte-Verbruggen/EMMSAD26_LLMquestionAnswering`

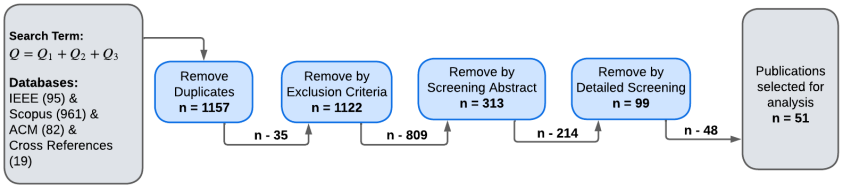

Fig. 1: Publications search process and filtering steps

Results. The remaining 51 publications were organized into relevant thematic categories : **Accessibility:** Studies on inclusive design and assistive technologies aimed at making models accessible to diverse users, including impaired individuals. **Summarization and comprehension:** Research focusing on means of condensing and abstracting, and comprehending complex models, thereby easing the interpretation and navigation of their contents. **Conversational:** Works exploring conversational agents or voice interfaces as intuitive, user-friendly channels for interacting with models. In the following, we will summarize the core contributions in each of these categories.

Accessibility in Modeling. Accessibility in model comprehension is crucial to enabling people with disabilities to fully participate in education and professional settings in conceptual modeling, model-driven engineering, and agile development [27]. Traditional modeling tools often rely on visual components and pointing-device-based interaction, which create significant barriers for visually impaired users to understand models and fully engage in modeling processes [20]. Despite some progress in software and web accessibility, conceptual modeling remains largely inaccessible for those with disabilities beyond visual impairments, including physical, auditory, or cognitive challenges [34, 26].

Empirical studies have confirmed that task completion rates and user satisfaction levels improve when interfaces are adapted based on modalities suited to the user's abilities [48]. The literature further shows that speech-based interactions are preferred in most cases [14, 15, 39, 43], especially when providing a high-level summary first, followed by potential follow-up questions from the user [7]. Several techniques to improve accessibility have been proposed, including blended modeling with a combination of textual and graphical model representations [20], tactile and haptic user interactions [20, 47], sonification of models [15, 43, 4], and the use of conversational AI for modeling assistance [36, 1, 31, 35].

Model Summarization/Comprehension. As the model size grows, its visual representations become increasingly dense and complicated. This complexity can hinder the understanding, communication, maintenance, and further development of the modeled system [13]. Model summarization techniques are vital for reducing the complexity of large models while maintaining their usability and interpretability. By providing higher-level abstractions and focusing on essential elements, summarization techniques make vast amounts of data more accessible to various stakeholders [46, 33].

To realize summarization techniques for conceptual models, these models first need to be transformed into an LLM-friendly input format, rather than using images or plain XML [40, 10]. Van der Zee et al. [46] focus on (simulation) model simplification, providing a review of simplification strategies, including abstraction and modularization. The clarity and operationalization of the construct "model understandability" are critical for effective model summarization and comprehension. Houy et al. [12] offer a comprehensive reconstruction of experimental research on conceptual model understandability, identifying multiple ways that influence how models are interpreted.

In the technological landscape of model summarization, several works leverage LLMs to implement context-aware summarization [33, 8, 11]. Other works propose a generic model decomposition technique that facilitates model management by decomposing complex models into smaller, comprehensible sub-models that conform to the original metamodel [21, 5].

Remaining challenges include carefully balancing the trade-off between simplicity and accuracy in summarization, as well as handling LLM hallucinations and systematically validating the summaries [8, 42].

Conversational Agents. Conversational LLMs have been leveraged to conceptual models in formats such as XML, JSON, and image-based representations [30, 22]. Research indicates that JSON and simplified XML formats offer the best trade-off between token efficiency and comprehension quality compared to regular XML and PNG, making them ideal for AI-driven model analysis [16]. Furthermore, structured prompt engineering techniques such as chain-of-thought and non-technical abstraction have been shown to significantly improve response accuracy and user engagement, e.g., in AI-powered BPMN interpretation tools [45] and domain modeling [38].

Answer to RQ1 Current summarization and conversational agent tools in conceptual modeling aim to simplify complex models and enhance user interaction through natural language, thereby improving accessibility for a diverse range of users. Despite many advances, significant challenges remain. The process of simplification must carefully balance reducing complexity with retaining critical causal relationships. Over-simplification can lead to the loss of key dependencies, whereas insufficient abstraction may leave users overwhelmed. Additionally, outputs generated by LLMs might suffer from issues such as hallucinations and factual inconsistencies, particularly in complex modeling scenarios, which undermines reliability.

Synopsis. The preceding analysis reveals significant remaining challenges, two of them we aim to address in the following: 1) **Representation format.** Existing research has utilized various representation formats to engage with LLMs for summarizing and comprehending conceptual models. These findings suggest that the representation format significantly affects the accuracy of the LLM's response. However, systematic research is needed to determine how best to represent conceptual models for LLMs. 2) **Systematic evaluation.** Existing research

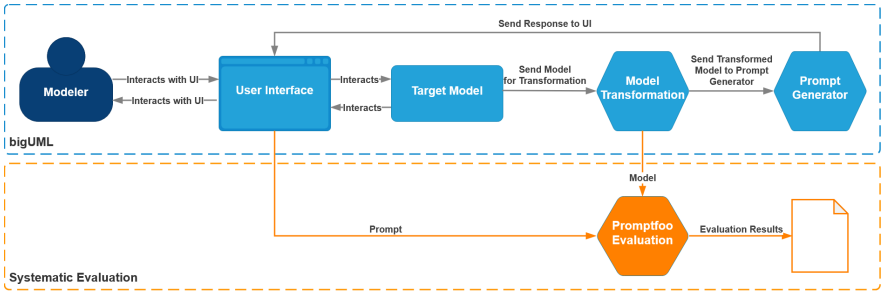

Fig. 2: High-Level System Architecture

has shown a lack of systematic evaluation of factors such as representation format and context size (also related to scalability) and their influence on the accuracy of LLMs' model-based question answering capabilities.

3 Prototype Design and Implementation

To evaluate RQ2, we designed and implemented a prototype of a model conversational agent. The prototype extends the BIGUML modeling tool and supports the evaluation of LLMs on model-based question answering (see Section 4).

3.1 Conceptual Design

The system architecture is designed to be flexible, scalable, and maintainable, integrating independent yet interrelated modules to work seamlessly together. Fig. 2 shows the high-level architecture using a C4 model [44]. The main modules needed to provide LLM support for model-based question answering are:

- **User Interface (UI):** The user interacts with the target model via the UI.
- **Model Transformation Module:** Before the model can be processed by the LLM, the Model Transformation Module restructures the original model into an intermediate representation that the LLM can interpret more easily.
- **Prompt Generator:** This module leverages an LLM to generate prompts to obtain clear and concise textual explanations of the target model.
- **Systematic Evaluation:** The systematic evaluation with the Promptfoo framework is further explained in Section 4.

The primary function of the **Model Transformation and Prompt Generator** module is to convert structured model data into a coherent, structured textual representation that is more appropriate for processing by an LLM. This intermediate format must preserve structural elements, relationships, and metadata, while removing unnecessary data like layout information. During the transformation, the original model is first parsed. In this step, all relevant elements, relationships, and metadata are identified and parsed. In the second step, the information is restructured into a unified textual format, ensuring that the result is accurate, concise, and reflective of the model's intrinsic structure. The transformed model data is passed to the LLM module (cf. Fig. 2).

In the **Prompt Generator** module, LLM processing is enhanced by carefully designing prompts—often incorporating role instructions, chain-of-thought reasoning, and contextual cues—that guide the LLM to produce more accurate and comprehensive responses. The LLM's natural language understanding capabilities are used to analyze the intermediate representation. Its tasks include: *i*) Understanding the hierarchy and relationships inherent in the transformed model; *ii*) Using its large context window to ensure that even models with extensive detail are processed without critical information being truncated; *iii*) Producing a concise and human-readable response to the prompted question.

3.2 Implementation

The conceptual architecture (Fig. 2) is implemented into the open source, GLSP-based [24] BIGUML [23] tool. BIGUML is extended by two central elements: *i*) a user interface extension for the conversational LLM interaction [35], and *ii*) the Prompt Generator Module itself, responsible for preparing, sending, and processing LLM prompts based on the UML model.

The interaction begins when the user selects their preferred options in the newly developed BIGUML AI agent interface, which the frontend forwards to the server. On the backend, the relevant model data is retrieved from the GLSP server, combined with user options, and passed to the Prompt Generator. The final prompt is sent to the LLM via the Google Generative AI Client. The generated explanation is returned to the backend and then passed back to the frontend UI for rendering in the AI Agent output area.

Modeling Environment (UI and Target Model). In addition to using the JSON-based model structure, the prototype also provides a second method for supplying model information via *PlantUML input*. This option can be configured directly in the UI. When enabled, the UI displays a text field that lets users enter a file path to an external PlantUML file. This part provides the prototype with a further perspective on comprehending UML models and enables us to evaluate the influence of model representation on the quality of LLM responses.

Prompt Generator. This module converts user-configurable settings and UML model information into structured prompts that an LLM processes to generate meaningful responses to comprehension questions. Once the prompt has been constructed, the module sends it to the LLM and returns the response to the UI. It represents the execution phase and directly links the UML model data with an automated explanation produced by the LLM.

The system instruction defines how the LLM should behave when processing the prompt. It is provided separately from the user query and acts as a higher-level guiding framework that the model must follow throughout the explanation process. The system instructs the model to act as a "UML expert" and explicitly defines rules for using a user-selected language style, carefully reading diagram data, avoiding hallucinations, and keeping responses concise. This approach is supported by [8, 9], indicating that minimal expert-focused system instructions

tend to work best for technical reasoning tasks. This design ensures that the prototype operates within controlled boundaries, supports evaluation of comparable outputs, and focuses on providing technically accurate explanations rather than interpretative speculation, which, in the case of UML model question answering, may lead to inaccurate statements or incorrectly referenced model elements.

After the prompt has been built, an LLM request is made using the Google Generative AI Client in a single-step execution without conversational history. This ensures deterministic processing and prevents interference from previous interactions. A **temperature** value of 0.0 was chosen to minimize randomness and encourage consistent responses across identical configurations. The maximum **token limit** of 1000 ensures that outputs are sufficiently detailed while preventing overly long or repetitive explanations. A comprehensive evaluation of the tool involving stakeholders is left for future work.

4 Experiments

This section focuses on the use of our BIGUML conversational extension to systematically evaluate LLMs' capabilities for UML model question-answering. We thus evaluate the accuracy of the LLM responses, which we identified as a key requirement of accessible systems. Evaluating the accessibility of the prototype in a user study is left for future work.

4.1 Research Questions & Evaluation Method

This section drills down into the second major research question (RQ2: To what extent do LLMs accurately answer comprehension questions on conceptual models?) by posing sub-questions aimed at uncovering a deeper understanding of the extent to which contextual factors influence the capabilities of LLMs in answering comprehension questions about UML models.

RQ2.1 To what extent does the *representation format* influence the ability of LLMs to accurately answer comprehension questions on UML models?
This RQ aims to investigate whether the representation of the UML model in *JSON or plantUML format* influences the LLM's ability to accurately respond to UML model comprehension questions.

RQ2.2 To what extent does the *context length* influence the ability of LLMs to accurately answer comprehension questions on UML models?
This RQ aims to investigate whether *the size of a UML model*, and consequently the context size, influences the LLM's ability to accurately respond to UML model comprehension questions.

RQ2.3 To what extent does *the training data* influence the ability of LLMs to accurately answer comprehension questions on UML models?
This RQ aims to investigate whether *reusing comprehension questions* already reported in the past, and, therefore, part of the LLM training data, influences the LLM's capabilities of accurately responding to UML model comprehension questions.

RQ2.4 To what extent does *the LLM provider* influence the ability of LLMs to accurately answer comprehension questions on UML models?
This RQ aims to investigate whether LLMs of *OpenAI (GPT4o-mini and 5) and Google (Gemini-2.5-flash-lite and Gemini-2.5-pro)* differ with respect to responding accurately to UML comprehension questions.

4.2 Experimental Setup

In this experiment, we investigate the ratio of passing responses (independent variable) given a certain accuracy threshold (dependent variable) under different formats, model sizes, and LLM providers (factors).

Evaluation Module (Promptfoo) Promptfoo[2] is an open-source framework for automated evaluation of LLM applications. It offers a practical way to test prompt interactions and supports reproducible benchmarking by running predefined prompts and test scenarios, checking model responses, and applying scoring rules to evaluate their accuracy. Promptfoo uses (YAML) configuration files, where evaluation scenarios are defined. These include the prompt to use, the test input (e.g., the model representation), the LLM provider, and the criteria responses must meet. This setup allows for a clear, structured definition of test cases without requiring complex programming.

To respond to RQ2.4, we executed experiments with four current LLMs, namely Gemini-2.5-flash-lite, Gemini-2.5-pro, GPT 4o-mini, and GPT-5-mini. PromptFoo supports various evaluation methods. In this case, the Rubric-based assertion is used, in which responses are assessed against predefined quality criteria. Rubric-based assertions allow nuanced analysis of responses and provide great flexibility in assessing accuracy, completeness, and clarity. This is especially useful in the context of model-based question answering, where semantically correct responses may vary in wording.

A key characteristic of Promptfoo's rubric-based evaluation is that scoring is conducted by a separate LLM explicitly defined as an assertion model. This model evaluates the response generated by the primary model under test based on the configured rubric. Applying an "LLM as a judge" approach, the assertion model interprets the evaluation instructions and determines for itself whether the response meets the defined quality criteria. This mechanism allows fully automated assessment and reduces human bias during evaluation [32]. In these experiments, we conducted five runs with GPT-5-mini as assessor (between 21/11/2025 and 26/11/2026), and five runs with gemini-2.5-flash-lite as assessor (between 21/02/2026 and 24/02/2026). To mitigate the risk of hallucinations by the assessor LLM, we verify the scoring of a sample of 72 responses, providing human validation of the LLM's assessment across model-question pairs, answering LLMs and assessor LLMs.

[2] Promptfoo homepage [online]: `https://www.promptfoo.dev/`

Selection of UML Models and Comprehension Questions. The selection of models and comprehension questions was based on the availability of UML class diagrams with accompanying comprehension questions. A total of four models from the empirical work by Sharif and Maletic [37] were selected to provide realistic, academically grounded evaluation material (Set A). To assess the effects of increased contextual complexity on model comprehension, an upscaling strategy was implemented. This involved extending the original models with additional classes and relationships to enlarge the structural context without altering the semantics of the original model. These injected elements were fully independent of the core classes, ensuring that they expanded the contextual load without interfering with existing elements.

For Set A, the comprehension questions used in this evaluation are taken directly from [37] to avoid introducing unintended biases through paraphrasing. These questions focus on core model understanding, including structural relations such as inheritance or associations, identification of key model elements, and interpretation of semantic roles. The expected answers for Set A are created manually by the authors. This ensures that the interpretation of the questions is scientifically grounded and correct.

In addition to the medium-sized models derived from [37], a second set (Set B in the following) of three models was included to extend the evaluation to larger-scale system representations. These models were sourced from the ModelSet dataset [18]. An overview of the number of elements in the models of set A and set B can be found in the online supplementary material.The corresponding comprehension questions and expected answers for set B are manually constructed by the authors. The selected models contain structures that are substantially more complex, thereby enabling the evaluation to assess LLM question-answering not only on academic examples but also on more realistic, industrial-scale models. This enables a comparative analysis of LLM question answering performance across different scales. Finally, since the comprehension questions sourced from [37] have been published, they may be included in the training data for the selected LLM providers, potentially leading to data leakage. Set B is therefore accompanied by new, manually defined comprehension questions to support RQ 2.3.

Assertion Setup. Each Promptfoo test case includes a rubric specification that instructs the evaluator LLM on how to judge the quality of the primary model's output based on predefined criteria. Listing 1.1 demonstrates how the Promptfoo evaluation works using an example test case. First (lines 1-3), the variables representing the model, the comprehension question, and the provided output are injected. The second block (lines 4-20) then defines the assertions (including which LLM provider provides the answer). In Promptfoo, each test case can specify one or more assertions under the assert key. An assertion describes how the model output should be evaluated and always includes a type field that selects the evaluation mechanism.

The rubric description consists of two parts. The first part details the evaluation criteria (lines 8-12) based on the expected outcome. These criteria check the correctness of the answer and the justification provided, allowing for variation in wording as long as the conceptual interpretation is correct. The second component is the accuracy grading scale (lines 13-20). Finally, line 21 defines the required accuracy score to pass the evaluation (in this example: 0.5).

This scoring system distinguishes between answers that are completely, mostly, or partially correct and those that are entirely incorrect. It also supports more precise analysis, as it captures the extent to which the model can justify its reasoning. Because the result of each test run is interpreted as binary (pass or fail), the evaluation is repeated with four threshold configurations to produce the full results matrix, reflecting the grading scale by varying the passing threshold (line 21). Moreover, to ensure stable results, we run the experiments 5 times and average the results across all runs.

Listing 1.1: Example Test Case

```
1  vars:
2    uml: file://workspace/model-test/model-A.txt
3    question: <Original comprehension question from source material>
4  assert:
5    - type: llm-rubric
6      provider: openai:gpt-5-mini
7      value: |
8          Evaluate the correctness and justification of the answer based
9          on the following:
10         - The response must identify the expected design conclusion.
11         - It must provide the correct rationale.
12         - Different wording is acceptable if conceptually accurate.
13
14         Grading scale:
15         - 1.0: Fully correct and justified.
16         - 0.75: Mostly correct with minor omissions.
17         - 0.5: Partially correct.
18         - 0.25: Vague or weak reasoning.
19         - 0.0: Incorrect or irrelevant.
20
21         A score of 0.5 or higher should be considered a pass.
```

Based on the format and the model upscaling, we defined six model format variants for Set A (PlantUML, PlantUMLx2, PlantUMLx10, JSON, JSONx2, and JSONx10). In total, 2.160 evaluation runs were conducted for Set A (9 model-question combinations × 6 formats × 4 answering LLMs × 2 assessor LLMs × 5 reps). For Set B, 480 runs were conducted (6 model-question combinations × 2 formats × 4 answering LLMs × 2 assessor LLMs × 5 reps). For each threshold, a cumulative score was calculated by relating the number of PASSING test cases to the total number of evaluated cases, using the following formula:

$$\text{Score}_\tau = \frac{\text{Number of Passed Test Cases}_\tau}{\text{Total Number of Test Cases}} \quad \text{with } \tau \in \{25\%, 50\%, 75\%, 100\%\}$$

4.3 Results

The quantitative outcomes summarized in the tables below provide an overview of how comprehension accuracy responds to changes in input structure, contex-

Table 1: Averaged comprehension accuracy for Set A, assessed by GPT-5-mini

Score	PlantUML	PlantUMLx2	PlantUMLx10	JSON	JSONx2	JSONx10
25 %	0,99 ± 0,01	0,97 ± 0,01	0,92 ± 0,02	0,96 ± 0,02	0,87 ± 0,01	0,86 ± 0,04
50 %	0,98 ± 0,03	0,96 ± 0,02	0,89 ± 0,02	0,94 ± 0,02	0,85 ± 0,03	0,80 ± 0,03
75 %	0,96 ± 0,02	0,92 ± 0,04	0,87 ± 0,03	0,89 ± 0,04	0,80 ± 0,04	0,76 ± 0,02
100 %	0,93 ± 0,02	0,92 ± 0,03	0,86 ± 0,03	0,87 ± 0,03	0,77 ± 0,02	0,74 ± 0,00

Table 2: Averaged comprehension accuracy for Set A, assessed by Gemini-2.5-flash-lite

Score	PlantUML	PlantUMLx2	PlantUMLx10	JSON	JSONx2	JSONx10
25 %	0,99 ± 0,02	0,97 ± 0,00	0,95 ± 0,01	0,93 ± 0,02	0,86 ± 0,02	0,83 ± 0,02
50 %	0,94 ± 0,02	0,96 ± 0,02	0,92 ± 0,03	0,93 ± 0,02	0,83 ± 0,02	0,80 ± 0,03
75 %	0,95 ± 0,02	0,92 ± 0,04	0,87 ± 0,02	0,90 ± 0,02	0,78 ± 0,04	0,76 ± 0,03
100 %	0,94 ± 0,02	0,89 ± 0,04	0,84 ± 0,03	0,87 ± 0,02	0,76 ± 0,02	0,73 ± 0,03

tual size, and model provider. Table 1 and Table 2 present the results for Set A with GPT-5-mini and Gemini-2.5-flash-lite as assessors, respectively.

For each model representation variant, the cumulative pass rate averaged over five runs is listed across the four thresholds. For PlantUML and JSON, the baseline model representations result in the highest comprehension accuracy. The PlantUML representation consistently outperforms the JSON representation. Increasing context complexity had a negative impact on performance for both formats, indicating that increased contextual load reduces the ability of LLMs to prioritize structurally relevant elements, likely due to context dilution and growing token-based complexity. A direct comparison between the two representation formats reveals clear differences in accuracy as the contextual size grows. When scaling to x2 and x10, JSON experiences a significantly greater performance degradation. Overall, Gemini's scores as an assessor are slightly lower but still show a similar pattern.

Answer to RQ2.1 Our results show that the PlantUML representation consistently leads to better comprehension than the JSON representation.

To answer RQ2.2, model size and token counts are used as metrics for the context size. For both formats, we see a consistent degradation of the accuracy as the model size increases in Table 1 and Table 2. JSON uses notably more tokens than PlantUML in every test setting. In the baseline version, the JSON format (480,891 tokens) requires 8.5 times as many tokens as the PlantUML format (56,488 tokens). Similar factors are found with increasing model size. This shows that JSON introduces much more structural information, thereby increasing the LLM's processing effort. Given that the PlantUML representation delivers higher comprehension accuracy (see answer to RQ2.1) while requiring significantly fewer tokens, our results correspond to the findings in [16], where more streamlined formats generally resulted in a better trade-off between response quality and token consumption.

Answer to RQ2.2 Our results show that the comprehension accuracy drops with increasing context length. The comprehension degradation is stronger for JSON than for PlantUML.

Table 3: Averaged comprehension accuracy for Set B, assessed by GPT-5-mini

Score	PlantUML	JSON
25 %	0,98 ± 0,02	0,95 ± 0,02
50 %	0,88 ± 0,00	0,85 ± 0,05
75 %	0,61 ± 0,04	0,45 ± 0,02
100 %	0,46 ± 0,00	0,27 ± 0,02

Table 4: Averaged comprehension accuracy for Set B, assessed by Gemini-2.5-flash-lite

Score	PlantUML	JSON
25 %	0,96 ± 0,00	0,86 ± 0,06
50 %	0,86 ± 0,02	0,79 ± 0,06
75 %	0,68 ± 0,05	0,56 ± 0,04
100 %	0,53 ± 0,02	0,40 ± 0,02

Tables 3 and 4 summarize the results of the evaluation runs of Set B, confirming that PlantUML yields higher accuracy. However, the accuracy for both formats is much lower in Set B than in Set A, as fewer cases achieve 75% or 100%. Apart from the models' complexity, this difference may also be due to the fact that the comprehension questions for Set B were not previously published, whereas those for Set A have been reported before. This effect is stronger when using GPT-5-mini as an assessor than when using Gemini-2.5-flash-lite.

Answer to RQ2.3 Reusing previously published data (in our case, comprehension questions) leads to inflated accuracy results.

A comparison of the evaluated LLM providers (see Table 5 and Table 6) confirms general expectations. The most recent models, GPT-5-mini and Gemini-2.5-pro, outperform the other models across all test configurations. These models perform consistently well for all PlantUML representations in Set A, regardless of their size. Note that GPT-5-mini tends to outperform Gemini-2.5-pro regardless of the LLM used as assessor.

Answer to RQ2.4 Newer models outperform older ones, and GPT-5-mini performs best among the four tested models in our experiments.

While using the 'LLM as assessor' eliminates potential biases of a human evaluator, the assessor LLM could still produce hallucinations. Therefore, we conduct a human validation on a sample of the results, consisting of 72 answers: 9 model-question pairs (Set A) × 1 format (PlantUML) × 4 answering LLMs × 2 assessor LLMs × 1 run. This resulted in an agreement percentage of 93,1% (67/72). Of the five differing evaluations, four were partially correct answers that the LLM labeled correct (at the passing threshold of 100%) while only one instance in this sample was incorrectly labeled as FAIL by the LLM assessor.

5 Discussion of Findings

Implications for Research and Practice. Our results show that while JSON and PlantUML are lightweight compared to formats like XML, there is a clear difference in model-based question-answering performance between the two. Therefore, a systematic assessment is needed of how different model representations

Table 5: Pass rates per LLM provider across model formats, averaged over 5 runs, assessed with GPT-5-mini

Score	Gemini-2.5-flash-lite	Gemini-2.5-pro	GPT 4o-mini	GPT-5-mini
Set A: PlantUML	0.93 ± 0.02	0.98 ± 0.02	0.96 ± 0.07	1 ± 0.00
Set A: PlantUML 2x	0.86 ± 0.04	0.98 ± 0.02	0.94 ± 0.08	1 ± 0.00
Set A: PlantUML 10x	0.81 ± 0.00	0.99 ± 0.02	0.79 ± 0.08	0.99 ± 0.02
Set A: JSON	0.91 ± 0.05	0.96 ± 0.04	0.82 ± 0.04	0.99 ± 0.02
Set A: JSON 2x	0.79 ± 0.02	0.84 ± 0.02	0.84 ± 0.05	0.88 ± 0.02
Set A: JSON 10x	0.78 ± 0.01	0.82 ± 0.02	limit	0.79 ± 0.02
Set B: PlantUML	0.67 ± 0.04	0.87 ± 0.02	0.54 ± 0.05	0.92 ± 0.00
Set B: JSON	0.38 ± 0.00	0.75 ± 0.03	0.55 ± 0.02	0.85 ± 0.04

Table 6: Pass rates per LLM provider across model formats, averaged over 5 runs, assessed with Gemini-2.5-flash-lite

Score	Gemini-2.5-flash-lite	Gemini-2.5-pro	GPT 4o-mini	GPT-5-mini
Set A: PlantUML	0.92 ± 0.00	1 ± 0.00	0.94 ± 0.05	1 ± 0.00
Set A: PlantUML 2x	0.85 ± 0.02	0.99 ± 0.01	0.94 ± 0.04	1 ± 0.00
Set A: PlantUML 10x	0.81 ± 0.01	0.97 ± 0.03	0.83 ± 0.04	0.97 ± 0.02
Set A: JSON	0.87 ± 0.02	0.92 ± 0.05	0.84 ± 0.02	1 ± 0.00
Set A: JSON 2x	0.79 ± 0.02	0.84 ± 0.02	0.74 ± 0.04	0.87 ± 0.02
Set A: JSON 10x	0.78 ± 0.00	0.83 ± 0.04	limit	0.76 ± 0.04
Set B: PlantUML	0.67 ± 0.05	0.87 ± 0.02	0.66 ± 0.07	0.89 ± 0.06
Set B: JSON	0.50 ± 0.03	0.73 ± 0.02	0.58 ± 0.04	0.87 ± 0.02

affect model-based question answering and other LLM-assisted modeling tasks. Representation formats should be developed specifically for model exchange with LLMs, ideally a generic one that can be used across multiple modeling languages. A standardized representation could be leveraged to develop a standardized integration of modeling tools, their interactions, and model exchange with LLMs.

Since our results show that context size matters, it would be beneficial to investigate how existing modularization techniques [2, 6, 29] can be applied before model exchange with LLMs. Moreover, further research could investigate new, specific semantic modularization techniques to help determine which parts of a model are most relevant to send to an LLM for a specific type of question or conversation (cf. [6]). This could contribute to the accessibility of modeling tools, e.g., by supporting visually impaired users in navigating models.

The results of RQ2.2 and RQ2.3 demonstrate that larger model sizes and previously unpublished questions reduce the accuracy of LLM responses. As such, the reliability of LLMs for model-based question answering should be improved before they can be used effectively to improve the accessibility of modeling tools.

Finally, future research would benefit from a clear overview of previously reported models and model-question pairs, to gain insight in potential data leakage in their LLM model-based question answering experiments.

Limitations. Although the evaluation provides valuable insights, several limitations must be acknowledged. First, the study focuses on a predetermined set of UML models from two sources. While this may not fully reflect the variety and complexity of models used in industry, it helps maintain comparability.

Second, only PlantUML and JSON model formats were tested, chosen for their compatibility with the prototype and widespread use. Other formats or

hybrid approaches may yield different results. Additionally, the model upscaling technique artificially increased context complexity by introducing independent structures without changing the core semantics. Although this approach supports controlled testing, it may not perfectly mirror real-world large models.

Third, the evaluation focuses solely on UML class diagrams, which represent only one type of structural model. Other diagram types, such as sequence diagrams, state machines, or activity diagrams, incorporate different semantic and behavioral aspects. These might interact differently with LLMs, leading to varied comprehension outcomes. Thus, the findings can not be generalized directly to other UML diagram types or modeling languages without further investigation.

Fourth, alternative prompt engineering strategies or more flexible response conditions could enhance accuracy scores, especially in higher-complexity scenarios. Furthermore, performance was assessed using an automated scoring method with 2 other LLMs serving as evaluator. While this promotes consistency, it may introduce bias inherent to the evaluation models themselves. However, a human validation of a sample of the results reaching an agreement percentage of 93.1%.

Fifth, LLM performance varies across model providers and versions, and the results reflect only the models available at the time of testing. Therefore, the findings should be interpreted within the context of current technology and not assumed to generalize to future LLM developments without further validation.

Finally, this study is based primarily on descriptive statistics rather than inferential statistical testing. While we have a total of 2.640 observations, the rich set of variables used in this experimental set-up (model-question pairs, formats, sizes, answering LLMs, assessing LLMs) means that the number of observations per configuration is very small. Therefore, any inferential analysis on the effect of a single variable would have low statistical power and must therefore be interpreted with caution. Nevertheless, inferential methods may still provide some supplementary insight; for transparency, these analyses are reported in the online supplementary material[1][28].

6 Conclusion

We investigated the capabilities of LLMs for model-based question answering over UML models, as this is a prerequisite for effectively using conversational agents to enhance modeling accessibility. An analysis of the state of the art revealed advances in summarization techniques and conversational agent tools for this purpose. However, the balancing of retaining key model elements and meaningful abstraction in model summarization, as well as handling hallucinations and factual inconsistencies, are still open challenges. We developed a prototype of a conversational agent for model-based question answering and investigated the effect of model representation, model size, novelty of the questions, and the LLM provider on the accuracy of the answers, showing that more compact formats, such as PlantUML, yield better results than more verbose formats, such as JSON. While both formats perform similarly in the baseline models, the difference becomes clear as model size increases. In addition, the accuracy of the

results decreases for questions that are not yet part of the LLMs' training data. GPT-5-mini performed the best among the four tested models. These findings pave the way for realizing model accessibility through conversational LLM-based user interactions by highlighting key factors that affect LLM performance.

Acknowledgments. This study was partially funded by the FFG-funded project SmartGLSP – Facilitating Large Language Modelsfor Smart GLSP-based Modeling (Grant number: FO999925707).

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

References

1. Alam, M.Z.I., Islam, S., Hoque, E.: Seechart: enabling accessible visualizations through interactive natural language interface for people with visual impairments. In: 28th International Conference on Intelligent User Interfaces. pp. 46–64 (2023)
2. Bork, D., Garmendia, A., Wimmer, M.: Towards a multi-objective modularization approach for entity-relationship models. In: Michael, J., Torres, V. (eds.) ER Forum, Demo and Posters 2020. pp. 45–58 (2020)
3. Bork, D., Klikovits, S., Michael, J., Netz, L., Rumpe, B.: Inclusive model-driven engineering for accessible software. In: MODELS 2025 NIER. pp. 253–259 (2025)
4. de Carvalho, J.F., Amaral, V.: Towards a modelling workbench with flexible interaction models for model editors operating through voice and gestures. In: COMPSAC 2021. pp. 1026–1031. IEEE (2021)
5. Corradini, F., Ferrari, A., Fornari, F., Gnesi, S., Polini, A., Re, B., Spagnolo, G.O.: A guidelines framework for understandable bpmn models. Data & Knowledge Engineering **113**, 129–154 (2018)
6. d'Aquin, M., Schlicht, A., Stuckenschmidt, H., Sabou, M.: Ontology modularization for knowledge selection: Experiments and evaluations. In: DEXA 2007. pp. 874–883 (2007)
7. Demir, S., Oliver, D., Schwartz, E., Elzer, S., Carberry, S., McCoy, K.F.: Interactive sight into information graphics. In: Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A). pp. 1–10 (2010)
8. Fahland, D., Fournier, F., Limonad, L., Skarbovsky, I., Swevels, A.J.: How well can large language models explain business processes? arXiv:2401.12846 (2024)
9. Fill, H.G., Fettke, P., Köpke, J.: Conceptual modeling and large language models: impressions from first experiments with chatgpt. Enterprise Modelling and Information Systems Architectures (EMISAJ) **18**, 1–15 (2023)
10. Gambe, B., Thomas, A.: Automated translation of uml class diagrams. In: 2024 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD). pp. 1–6. IEEE (2024)
11. Gavric, A., Bork, D., Proper, H.: How does uml look and sound? using ai to interpret uml diagrams through multimodal evidence. In: Advances in Conceptual Modeling, ER 2024 Workshops. pp. 187–197. Springer (2024)
12. Houy, C., Fettke, P., Loos, P.: Understanding understandability of conceptual models–what are we actually talking about? In: ER 2012. pp. 64–77 (2012)

13. Jošt, G., Huber, J., Heričko, M., Polančič, G.: An empirical investigation of intuitive understandability of process diagrams. Comput. Stand. & Interfaces **48**, 90–111 (2016)
14. Kim, N.W., Ataguba, G., Joyner, S.C., Zhao, C., Im, H.: Beyond alternative text and tables: Comparative analysis of visualization tools and accessibility methods. In: Computer graphics forum. vol. 42, pp. 323–335. Wiley Online Library (2023)
15. Kim, N.W., Joyner, S.C., Riegelhuth, A., Kim, Y.: Accessible visualization: Design space, opportunities, and challenges. In: Computer graphics forum. vol. 40, pp. 173–188. Wiley Online Library (2021)
16. Kourani, H., Berti, A., Hennrich, J., Kratsch, W., Weidlich, R., Li, C.Y., Arslan, A., Schuster, D., van der Aalst, W.M.: Leveraging large language models for enhanced process model comprehension. arXiv preprint arXiv:2408.08892 (2024)
17. Larkin, J.H., Simon, H.A.: Why a diagram is (sometimes) worth ten thousand words. Cognitive science **11**(1), 65–100 (1987)
18. López, J.A.H., Izquierdo, J.L.C., Cuadrado, J.S.: Modelset: a dataset for machine learning in model-driven engineering. Softw. Syst. Model. **21**(3), 967–986 (2022)
19. Lukyanenko, R., Bork, D., Storey, V.C., Parsons, J., Pastor, O.: Inclusive conceptual modeling: Diversity, equity, involvement, and belonging in conceptual modeling. In: Companion Proceedings of ER 2023 (2023)
20. Luque, L., Veriscimo, E.d.S., Pereira, G.d.C., Filgueiras, L.: Can we work together? on the inclusion of blind people in uml model-based tasks. In: Inclusive Designing: Joining Usability, Accessibility, and Inclusion. pp. 223–233. Springer (2014)
21. Ma, Q., Kelsen, P., Glodt, C.: A generic model decomposition technique and its application to the eclipse modeling framework. Softw. Syst. Model. **14**, 921–952 (2015)
22. Martínez-Lasaca, F., Díez, P., Guerra, E., de Lara, J.: Lowcobot: Towards chatting with low-code platforms (2024)
23. Metin, H., Bork, D.: Introducing bigUML: A flexible open-source glsp-based web modeling tool for uml. In: MODELS-C 2023. pp. 40–44 (2023)
24. Metin, H., Bork, D.: On developing and operating glsp-based web modeling tools: Lessons learned from bigUML. In: MODELS 2023. pp. 129–139. IEEE (2023)
25. Michael, J., Netz, L., Rumpe, B., Müller, I., Grundy, J.C., Wickramathilaka, S., Khalajzadeh, H.: Addressing visual impairments with model-driven engineering: A systematic literature review. CoRR **abs/2510.06483** (2025)
26. Miñón, R., Arrue, M., Abascal, J.: Conceptual model for automatic generation of context-sensitive user-tailored interfaces. In: Proceedings of the XV International Conference on Human Computer Interaction. pp. 1–4 (2014)
27. Miranda, D., Araújo, J., Liebel, G.: A conceptual model for web accessibility requirements in agile development. In: 1st IEEE/ACM Workshop on Multidisciplinary, Open, and RElevant Requirements Engineering. pp. 15–21 (2024)
28. Mischak, M., Verbruggen, C., Langer, P., Bork, D.: Charlotte-verbruggen/EMMSAD26_LLMquestionAnswering: v1.0 (Apr 2026). `https://doi.org/10.5281/zenodo.19630170`
29. Moody, D.L.: Entity connectivity vs. hierarchical levelling as a basis for data model clustering: An experimental analysis. In: Database and Expert Systems Applications, 14th International Conference, DEXA. pp. 77–87. Springer (2003)
30. Pérez-Soler, S., Daniel, G., Cabot, J., Guerra, E., de Lara, J.: Towards automating the synthesis of chatbots for conversational model query. In: Enterprise, Business-Process and Information Systems Modeling 2020. pp. 257–265 (2020)

31. Pidó, S., Pinoli, P., Crovari, P., Ieva, F., Garzotto, F., Ceri, S.: Ask your data—supporting data science processes by combining automl and conversational interfaces. IEEE Access **11**, 45972–45988 (2023)
32. Promptfoo, I.: Llm rubric — model-graded expected outputs configuration — promptfoo documentation. `https://www.promptfoo.dev/docs/configuration/expected-outputs/model-graded/llm-rubric/` (2025), accessed: 2025-11-25
33. Ramprasad, A., Sivakumar, P.: Context-aware summarization for pdf documents using large language models. In: 2024 International Conference on Expert Clouds and Applications (ICOECA). pp. 186–191. IEEE (2024)
34. Sarioğlu, A., Metin, H., Bork, D.: Accessibility in conceptual modeling - A systematic literature review, a keyboard-only UML modeling tool, and a research roadmap. Data Knowl. Eng. **158**, 102423 (2025)
35. Schwantler, S., Klikovits, S., Metin, H., Langer, P., Bork, D.: Talk to me! toward speech-based uml modeling. In: PoEM 2025. p. 83–101 (2025)
36. Seo, J., Kamath, S.S., Zeidieh, A., Venkatesh, S., McCurry, S.: Maidr meets ai: Exploring multimodal llm-based data visualization interpretation by and with blind and low-vision users. In: ACM SIGACCESS 2024. pp. 1–31 (2024)
37. Sharif, B., Maletic, J.I.: An empirical study on the comprehension of stereotyped uml class diagram layouts. In: ICPC 2009. pp. 268–272. IEEE (2009)
38. Silva, J., Ma, Q., Cabot, J., Kelsen, P., Proper, H.A.: Towards human-in-the-loop llm-enabled domain modeling. In: ER 2025. pp. 127–145 (2025)
39. Siu, A., SH Kim, G., O'Modhrain, S., Follmer, S.: Supporting accessible data visualization through audio data narratives. In: CHI 2022. pp. 1–19 (2022)
40. Speth, S., Meiliner, N., Becker, S.: Chatgpt's aptitude in utilizing uml diagrams for software engineering exercise generation. In: CSEE&T 2024. pp. 1–5 (2024)
41. Sweller, J.: Cognitive load theory. In: Psychology of learning and motivation, vol. 55, pp. 37–76. Elsevier (2011)
42. Tikka, S., Helske, J., Karvanen, J.: Clustering and structural robustness in causal diagrams. Journal of Machine Learning Research **24**(195), 1–32 (2023)
43. Torres, M.J.R., Barwaldt, R., Pinho, P.C.R., de Topin, L.O.H., Otero, T.F.: An auditory interface to workspace awareness elements accessible for the blind in diagrams' collaborative modeling. In: Frontiers in Education. pp. 1–7 (2020)
44. Vázquez-Ingelmo, A., García-Holgado, A., García-Peñalvo, F.J.: C4 model in a software engineering subject to ease the comprehension of uml and the software. In: EDUCON 2020. pp. 919–924. IEEE (2020)
45. Villegas-Ch., W.E., Govea, J., Gutierrez, R.: Optimizing language model-based educational assistants using knowledge graphs: Integration with moodle LMS. IEEE Access **12**, 191994–192012 (2024)
46. van der Zee, D.J.: Approaches for simulation model simplification. In: 2017 Winter Simulation Conference (WSC). pp. 4197–4208. IEEE (2017)
47. Zhao, Y., Nacenta, M.A., Sukhai, M.A., Somanath, S.: Tada: Making node-link diagrams accessible to blind and low-vision people. In: Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems. pp. 1–20 (2024)
48. Zouhaier, L., BenDalyHlaoui, Y., Ayed, L.B.: Adaptive user interface based on accessibility context. Multimedia Tools and Applications **82**(23), 35621–35650 (2023)