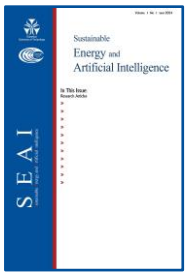
Hamedan University of Technology
Iranian Association of Electrical and Electronics Engineers

Sustainable Energy and Artificial Intelligence

Online ISSN: 3060-8015

Journal Homepage: www.enai.hut.ac.ir



Large Language Models as an Assistant to Interpret UML Models in Model-Based Engineering: An Exploratory Study

Hassan Bashiri[1*], Alireza Khalilipour[2], Parsa Bakhtiari[1], Moharram Challenger[2]

[1] *Department of Computer Engineering, Hamedan University of Technology, Hamedan, Iran*
[2] *Department of Computer Science, University of Antwerp, and Flanders Make Strategic Research Center, Antwerp, Belgium*
* *Corresponding Author: bashiri@hut.ac.ir*

Article Info

Article type:
Original Article

Article history:
Received 2024-09-25;
Revised 2024-11-01;
Accepted 2024-11-17.

How to cite this article:
Bashiri, H., Khalilipour, A., Bakhtiari, P. and Challenger, M. (2025). Large Language Models as an Assistant to Interpret UML Models in Model-Based Engineering: An Exploratory Study. *Sustainable Energy and Artificial Intelligence*, *1*(1), 45-50.
DOI: 10.61186/seai.2409-1009

Abstract

Creating a formal common language, beyond the ambiguities of natural languages, between different stakeholders, from analysts to test engineers, is one of the key goals of software modeling. Although notations are standard in software modeling languages such as UML, junior engineers' interpretation of models varies. Model interpretation in the presence of experienced people increases the learning rate for junior engineers. One of the potentials of large language models is the ability to interpret images and models. This research aims to use large language models as an assistant to interpret UML models to increase junior engineers' learning rate and understanding of the software models. We conducted an evaluation study to examine how helpful an LLM can be to help interpret the software models. Although large language models are still not very accurate in interpreting UML models, the experiment's results showed that students' learning rates increased by LLMs as model interpretation assistants. In other words, the large language model worked well as a teaching assistant. The detailed results of this exploratory study are reported in this paper.

***Keywords*:** Large Language Model; Model-Based Engineering; Software Modeling Language; UML; Model Interpretation Assistant.

1. Introduction

Software modeling is one of the key pillars of the software development process. It creates a common language to understand the requirements, expands design thinking, and examines different solutions in terms of time and cost [1,2]. Good models are essential for communicating between stakeholders and project teams. As the system becomes bigger and the system under study becomes more complex, the importance of modeling becomes clearer [3].

This research focuses on software modeling activity. We hypothesize that junior engineers must first acquire the ability to interpret models to become experts in software modeling. The more complete and comprehensive examples they examine, the stronger their model-driven thinking abilities.

There are two challenges in teaching software modeling [4]:

1. Teachers do not have comprehensive access to software development models, especially models close to reality. These models can be

used as an educational resource. The more diverse the models, the more efficient the training. In other words, accessing a set of modeled diagrams corresponding to a problem is a valuable educational resource for mentors and modeling instructors.

2. It takes a lot of practice to become an expert in modeling. The practical step is that junior engineers draw models for a problem, and mentors or experienced people check their models, and possible problems are reflected to them for learning. However, this method of training, in addition to being time-consuming, also involves the challenge of accessing experienced people and mentors. Furthermore, as part of this practice, the trainees should do model interpretation, which helps them understand enough details of the model and the notation.

This research examines the role of large language models (LLMs) in addressing the above-mentioned challenges. We evaluate LLMs' roles as mentors and teaching assistants in interpreting UML models.

LLMs are used in various sectors, and their scope is increasing daily. However, the issue of using large language models in interpreting models has not been addressed much. This research empirically evaluates the idea of using LLMs as a teaching assistant for a better and more comprehensive interpretation of UML diagrams.

The rest of this paper is organized as follows: Section 2 elaborates on the literature, reporting related work. Section 3 discusses the main concepts of LLMs. Section 4 describes the proposed approach to evaluating the usefulness of LLM in interpreting software models. Finally, section 5, presents the results and concludes the paper.

2. Related Work

Since the current study deals with generated interpreted text via large language models, in this section, we will consider related studies. In this sense, how LLMs generate text, the generation of model interpretation text, and the utilization of LLMs in educational venues are the most desired keys.

The advent of large language models (LLMs) such as GPT-2 and Bert12 currently involves plenty of disciplines, e.g., text generation. Wei et al. [5] have categorized these languages based on diverse characteristics, including their abilities in text generation. According to this study, an effective decoding technique should generate meaningful continuation given a context. As a result, LLMs are worthy of consideration for generating text that interprets circumstances, i.e., UML models.

In overview research, Naveed et al. [6] examined the development of language processing from the earliest approaches to natural language processing to the generation of text adopting the most recent large language models. The study specifically tackles the issues of integrating large models with education. In this regard, some studies examined LLMs as an educational assistance.

How language models can be used as teaching aids by students and teachers has been covered in studies, [7,8] and [9]. By assessing their performance, preferences, and learning styles, LLMs may assist students in creating individualized learning experiences by generating practice questions and study materials exclusively for them [7]. Lesson plan creation, assignment grading, and the creation of inclusive and diverse educational content can all be facilitated by LLMs for teachers, allowing up a considerable amount of time for instruction and student interaction [8,9]. As advanced conversational partners in language learning, LLMs may reproduce conversations in various languages, improve vocabulary, fix grammar, and help with pronunciation to satisfy practice requirements for fluency [10]. Such studies determine LLMs are slightly becoming a considerable teaching assistance, hence they would be a basement of the associated research.

In the context of interpreting conceptual models such as UML models, Conrardy and Cabot have introduced a mechanism to generate image-based UML class diagrams using LLMs [11]. They determined and assessed UML models from images by applying LLMs GPT-4V, Gemini Pro, Gemini Ultra, and CogVLM, in different level details of prompts. Although this study is a turn signal for conceptual model generation Conversely, our approach has employed LLMs to interpret the models.

Finally, regarding the dataset, Fatma et al. provide a benchmark dataset of functional requirements [12]. They have employed an LLM engine (GPT 3.5) to automate the generation of UML use case diagrams associated with the requirements. Moreover, ModelSet, a dataset of Ecore metamodels and UML-labeled models, has been provided in [13]. Although we have not used this kind of data-which will be explained later—it is evident that these datasets can be used as input for large language models during the learning phase.

3. Large Language Model Concepts

Large language models represent a significant advance in natural language processing. The way machines understand and produce human language has completely changed. LLMs are complicated systems composed of several key components, each of which plays a critical role in the performance and efficiency of the model. Understanding these components helps one understand language models and their capabilities in various applications. The main components of a large language model are:

- **Architecture:** At the core of any LLM is its architectural design, which determines how information flows in the model and how input data is processed.
- **Attention Mechanism:** The attention mechanism is one of the basic components of the transformer architecture that allows the model to focus on specific parts of the input sequence in producing the output. The attention mechanism allows the model to display the importance of different words or tokens in the input with different weights and improve the ability to understand and produce text.
- **Embeddings:** Embeddings represent words or tokens in a continuous vector space. These embeddings understand the semantic connections between words and allow the model to infer meaning and context from the input data. Techniques such as Word2Vec, GolVe, or various text embeddings such as BERT embeddings are commonly used.
- **Training data:** The quality and quantity of training data significantly affect the performance of a large language model. These models are trained on huge textual data from diverse sources such as books, articles, websites, and social media platforms.
- **Parameters:** A large language model is characterized by many parameters that determine the complexity and flexibility of the model. These parameters are learned during training through techniques such as stochastic gradient descent or variational versions such as Adam's optimization. The size of the parameter space directly affects the model's capacity to learn complex patterns and nuances in the language.
- **Fine Tuning / Reconfiguration:** LLMs are typically pre-trained on large datasets; they can be fine-tuned on task-specific data to adapt them to task-specific applications. Fine-tuning involves updating model parameters using smaller, purpose-specific datasets, allowing the model to specialize in text classification, language translation, sentiment, or knowledge analysis in specific domains.

4. Evaluation Approach

In the problem statement section, we mentioned two major problems in software modeling education. As a possible solution for the first challenge, we propose producing models from the text using LLM, which can be used both as a draft version of the software model in the modeling process and for the teaching process; and the second is to use LLM's potential as a mentor and teaching assistant in the interpretation of UML models.

Under the current conditions of LLMs, they cannot model from text in a language such as UML. Therefore, it is necessary to equip large language models with the ability to draw the model in addition to fine-tuning it in modeling.

However, for the second problem, LLMs can become teaching assistants, meaning we should use them to interpret models and increase the quality of software modeling education for students and junior engineers. In this research, we have focused on the second challenge and second solution. We employed the students of a software engineering class to test the capability of the large language model in interpreting the software models. Fifteen students who have attended this course are involved in the modeling training section. We used the LLMs as teaching assistants to strengthen the students' interpretation of the model by the students as shown in Fig. 1.

The steps to do the evaluation are as follows:

- A class diagram, use case diagram, and component diagram with UML were presented to the students.
- They were asked to write their interpretation of these diagrams in natural language.
- We gave the image of each diagram separately to the LLMs that can interpret the image (ChatGPT4 and Gemini) and asked the LLM to write the image interpretation (software model) for us.
- The students read the commentary produced by the LLM.
- In the end, each student identified how the LLM helped them better understand the model, compared with their initial understanding of the software model.

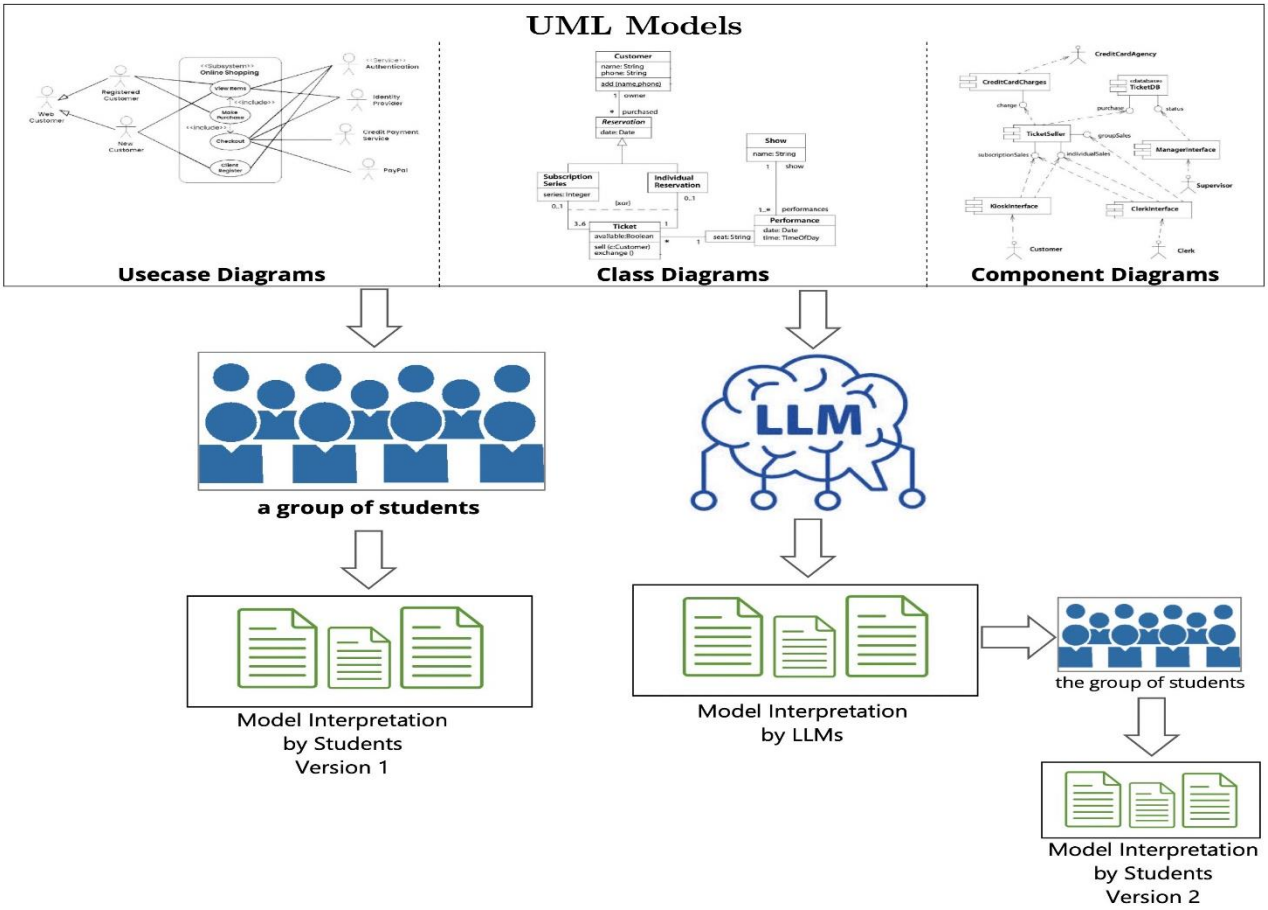


Fig. 1. The proposed approach for model evaluation

For the evaluation, some of the input UML models were selected from the reference book for interpretation by LLM [14], and some other models were part of the student's homework.

5. Results and Conclusion

As seen in Fig. 2, in general, 57.1% of the evaluator students expressed that the LLM helped them a lot in understanding and interpreting the model and rewriting it more accurately. Also, 28.6% of the evaluators stated that the LLM helped them a little to understand the model better. However, 14.3% of the participants declared that the LLM did not help them at all.

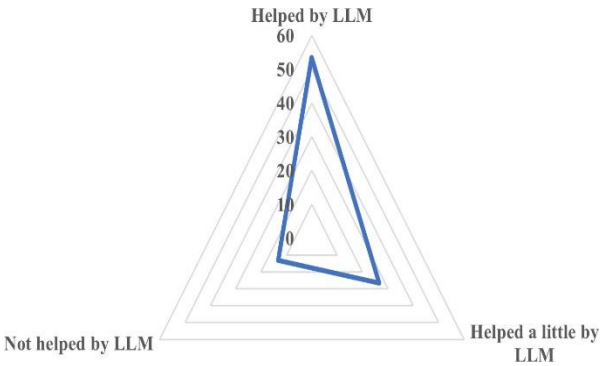


Fig. 2. Experiment results

The noteworthy point in examining UML diagrams is that the more difficult the UML model, the better and more the LLM helps to understand the interpretation of the diagrams.

Some of the most important points that the evaluator students mentioned after using the LLM as an advantage are:

- The LLM first gives a general view of the problem and then interprets the different parts in a categorized way. This method of interpretation teaches the student to examine the model from different perspectives in addition to interpreting its components. While the details of modeling are critical, understanding the whole image of the model and the larger angle of view can help them understand the models better and ultimately improve their modeling skills.
- Also, another advantage is the possibility of continuing the conversation to disambiguate the parts that the student did not understand and need further explanation.
- Asking the LLM to provide suggestions for completing and modifying the model was another thing that several students mentioned as an advantage of the role given to the LLM. Although the objective of the research is not to measure the assistance of LLMs to junior

engineers in UML modeling, this advantage can greatly help them in drawing complete and comprehensive diagrams. The initial diagrams drawn by the student or junior engineer are given to the LLM, and he/she is consulted about the weaknesses or shortcomings of the diagram in the role of a mentor.

In this experiment, only one person had access to ChatGPT4 and rated the quality of the commentary as much better than that produced by Gemini. To control consistency, the data of this user dropped out of the analysis.

Finally, as a threat to the validity of this research, the number of students in this experiment is not large. Still, the results obtained can be re-evaluated in a wider experiment with more participants. In conclusion, this research and the small experiment designed for it show the great potential of using large language models to advance model-based engineering. Four out of five evaluators claimed that LLM could help him/her interpret the software models. Because the roles of software system analysts and designers cannot be easily replaced, the importance of accurate and comprehensive training for junior engineers can be dramatically increased using LLMs as interpreters of software models. Finally, the results show that the experiment had a positive effect on participants' perception of LLMs as teaching assistants. In other words, simply put, junior engineers accept LLMs as their teaching assistants.

In future work, we would like to focus on the first part of the problem: producing models from the text using LLM, which can be used both as a draft version of the software model in the modeling and teaching processes.

References

[1] Brambilla, M., Cabot, J., & Wimmer, M. (2017). *Model-driven software engineering in practice*. Morgan & Claypool Publishers.

[2] Broy, M., Kirstan, S., Krcmar, H., & Schätz, B. (2012). What is the benefit of a model-based design of embedded software systems in the car industry?. In *Emerging technologies for the evolution and maintenance of software models* (pp. 343–369). IGI Global.

[3] Bruneliere, H., Eramo, R., Gomez, A., Besnard, V., Bruel, J. M., Gogolla, M., ... & Rutle, A. (2018). Model-Driven Engineering for Design-Runtime Interaction in Complex Systems: Scientific Challenges and Roadmap: Report on the MDE@ DeRun 2018 Workshop. In *Software Technologies: Applications and Foundations: STAF 2018 Collocated Workshops, Toulouse, France, June 25-29, 2018, Revised Selected Papers* (pp. 536-543). Springer International Publishing.

[4] Di Rocco, J., Di Ruscio, D., Iovino, L., Laemmel, R., & Pierantonio, A. (2017). MDE adoption—a three-legged chair. In *Proceedings of the Workshop on Grand Challenges in Modeling at STAF*.

[5] Wei, C., Wang, Y. C., Wang, B., & Kuo, C. C. J. (2023). An overview on language models: Recent developments and outlook. *arXiv preprint arXiv:2303.05759*.

[6] Naveed, H., Khan, A. U., Qiu, S., Saqib, M., Anwar, S., Usman, M., ... & Mian, A. (2023). A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*.

[7] Dai, W., Lin, J., Jin, H., Li, T., Tsai, Y. S., Gašević, D., & Chen, G. (2023, July). Can large language models provide feedback to students? A case study on ChatGPT. In *2023 IEEE International Conference on Advanced Learning Technologies (ICALT)* (pp. 323-325). IEEE.

[8] Kasneci, E., Seßler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., ... & Kasneci, G. (2023). ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and individual differences*, *103*, 102274.

[9] Rane, N. (2023). Enhancing the quality of teaching and learning through ChatGPT and similar large language models: challenges. *Future prospects, and ethical considerations in education (September 15, 2023)*.

[10] Young, J. C., & Shishido, M. (2023). Investigating OpenAI's ChatGPT potentials in generating Chatbot's dialogue for English as a foreign language learning. *International journal of advanced computer science and applications*, *14*(6).

[11] Conrardy, A., & Cabot, J. (2024). From Image to UML: First Results of Image Based UML Diagram Generation Using LLMs. *arXiv preprint arXiv:2404.11376*.

[12] Bozyigit, F., Bardakci, T., Khalilipour, A., Challenger, M., Ramackers, G., Babur, Ö., & Chaudron, M. R. (2024). Generating domain models from natural language text using NLP: a benchmark dataset and experimental comparison of tools. *Software and Systems Modeling*, 1-19.

[13] López, J. A. H., Cánovas Izquierdo, J. L., & Cuadrado, J. S. (2022). ModelSet: a dataset for machine learning in model-driven engineering. *Software and Systems Modeling*, 1-20.

[14] Rumbaugh, J., Jacobson, I., and Booch, G. *The unified modeling language reference manual (2nd Edition)*. Pearson Education India, 2004.

Biography

Hassan Bashiri is a faculty member in the department of computer engineering at the Hamadan University of Technology. He does research in fields such as intelligent information retrieval and large language models, as well as agent-oriented modeling in complex systems. Mr. Bashiri also teaches courses such as software engineering and systems analysis and design at the university. Since 2024, he has started research on large language models as a research associate with MICSS Lab at the University of Antwerp. The team aims to develop large language models for domain-specific applications. He is also an Executive Board member of the International CoMSES Association, which is headquartered at the University of Arizona. The purpose of CoMSES is to develop, share and reuse computational models that support social research.

Alireza Khalilipour is a PhD researcher at the University of Antwerp, focusing on advancing Software Architecture, Model-Driven Software Engineering (MDSE), and Artificial Intelligence (AI) for complex systems. He is an active member of the Modelling Intelligent Complex Software and Systems (MICSS) Lab, where he explores intelligent model management techniques. His work in the MICSS Lab involves applying data science methods to extract valuable insights, patterns, and knowledge from both conceptual and geometrical models. This research has applications in improving model-driven practices and automating the analysis of complex software architectures, which is essential in fields such as AI-driven engineering and large-scale software development.

Parsa Bakhtiari is a master student in Artificial Intelligence from Hamedan University of Technology, specializing in natural language modeling and large language models (LLMs). His research focuses on enhancing human-computer interaction through advanced AI models and exploring the applications of LLMs in various industries. Parsa aims to contribute to developing more intuitive and intelligent systems that address the needs of modern industry.

Moharram Challenger received his PhD in IT from the International Computer Institute at Ege University in February 2016. From 2010 to 2013, he was a researcher and team leader of a bilateral project between Slovenia and Turkey (TUBITAK). From 2012 to 2016, he was the R&D director of UNIT IT Ltd., leading one national project funded by TUBITAK and two international software-intensive projects in Europe called ITEA Model Writer and ITEA Assume. During 2016, he did an external postdoc at Wageningen University of Research, in the Netherlands. In 2017 and 2018, he has been a member of the faculty as an assistant professor at Ege University. From 2019 to 2020, he was a postdoc researcher at the University of Antwerp, doing research in Flanders Make projects. He is currently a tenure-track assistant professor in the Department of Computer Science at the University of Antwerp. His research interests include domain-specific modelling languages, multi-agent systems, cyber-physical systems, the Internet of Things, digital twins, and intelligent systems.