Transactions on Engineering and Computing Sciences - Vol. 11, No. 1
Publication Date: January 25, 2023
DOI:10.14738/tmlai.111.13820.
Almasabe, A., Ludi, S., & Elfaki, A. O. (2023). Scrutinizing UML Teaching and Learning Modeling Tools. Transactions on Engineering and Computing Sciences, 11(1). 01-21.

Scrutinizing UML Teaching and Learning Modeling Tools

Abdulqader Almasabe
University of North Texas TX, USA

Stephanie Ludi
University of North Texas TX, USA

Abdelrahman Osman Elfaki
University of Tabuk, Tabuk, KSA

ABSTRACT

The purpose of this paper was to identify, synthesize, and analyze the modeling tools, especially those that support uml modeling and interactive learning methods for software engineering and information systems design courses. The goals were to guide both professors and students to choose the proper modeling tools to support software engineering courses objectives. The research identified many successful modeling tools that can help students use what is suitable for learning uml modeling and contribute to engaging students in modeling. Moreover, this research used the meta-ethnography method for synthesizing qualitative results in the area of software engineering, especially modeling. Group of modelling tools have been chosen for this paper based on their usages popularity and successfulness in producing high quality design models. The contribution of this paper is highlighting and defining the strengths, weakness, and limitations of each studied tool.

Keywords: System Development; Modelling; UML

INTRODUCTION

Software engineering requires practical application of knowledge, and its main activity is to perform software development. It is not enough to teach software engineering through traditional theory-based lectures, because the course should be interactive and a collocative activity [1], [2]. Software modeling is the core topic in teaching software engineering, since modeling plays a major role in understanding its fundamental concepts and developing high-quality software [3]. We use models all the time to think about problems; solve these problems; construct mechanisms; analyze, design, and develop software; and teach. Besides, models can improve our understanding of a system's behavior and engage learners and students in modeling because models help students and educators learn more robustly [4], [5].

Unified modeling language (UML) is a modeling system that can simplify communication between software engineers and help developers deal with difficult problems by developing the issues, solving the problems, and picturing the design of a system [6]. UML has become a de facto standard language of the software development process, and software developers should be prepared to use UML and modeling effectively in their work. In addition, UML uses diagrams

in software development without providing instructions for using these diagrams, so it is the only language that supports diagrams notation.

However, using modeling tools is important for teaching students how to model, turn their models into real executable systems, and get useful feedback about their models [7]. Students who learn to model by documentation or simple drawing tools are disadvantaged when they enter the job market [8]. Therefore, knowing the strengths and weaknesses of modeling tools (especially those supported by UML) can help professors select tools that teach their students in an efficient way. Professors should choose modeling tools that are free and open source, with considerations of their use, complexity, support for UML, the feedback they give, and their installation (availability). If students are allowed to select their modeling tools, it would be useful for them to know the strengths and weaknesses of each tool so that they can share that knowledge with their peers.

The purpose of this research was to identify, synthesize, and discuss the strengths and weaknesses of modeling tools, especially those that support UML modeling and interactive learning methods for software engineering and courses in information system design. These are interactive learning courses that also use UML modeling. Knowing the benefits and drawbacks of modeling tools enables professors and students to choose the proper one. These modeling tools can help students learn UML modeling, and that will help to engage students in modeling. Besides, this research used meta-ethnography methods for qualitative methodology and for synthesizing its findings.

RESEARCH QUESTIONS

In this section, the boundaries and scope of this paper have been denoted by putting the research questions as follows:

- What are the strengths (benefits) that motivate professors and students use modeling tools for teaching and learning UML modeling?
- What are the weaknesses (drawbacks) that discourage professors and students from using modeling tools for learning UML modeling?
- What are the proper modeling tools for teaching and learning UML modeling?
- How can modeling tools support an interactive learning method?

LITERATURE REVIEW

In this section, we focus on the research that has studied the features of UML tools and some research that used modeling tools as part of interactive learning. The findings of the literature review are used in the methodology section, where meta-ethnography is the qualitative methodology applied to the literature findings. Therefore, we identify and synthesize the disadvantages and advantages of modeling tools for students to learn and professors to teach based on their strengths and weaknesses, and whether they can be used to support interactive learning.

Research Used a Modeling Tool to Support Interactive Learning

Krusche, et al. [9] conducted research using an online editor that helped more than 1000 students in their software engineering modeling course for undergraduate students. With the

increase in the number of students in software engineering introductory courses, such as UML modeling, it is difficult for instructors to teach the creative aspects of modeling and get feedback from each student, especially for exams and correcting exercises. Besides, it is not possible to evaluate output models solutions immediately in a software engineering course, especially UML, because there could be multiple acceptable solutions for each question. Researchers have developed an interactive learning method for modeling that is based on the usability of online editors, such as Artemis. Artemis is an open-source and exercise system that gives individual feedback. It is used in interactive learning courses by many universities, and it integrates with Apollon. Apollon is an open-source and online modeling editor that provides seven UML diagrams. Artemis and Apollon support students and instructors in learning and teaching modeling throughout the entire software engineering lifecycle. Based on an online survey, data analysis, and quasi-experiment, an interactive learning method for modeling that used an easy online editor improved the learning and outcomes of students by up to 87%, and it increased their motivation for using modeling. The benefits of Apollon are that it is an open-source, lightweight, free web application. It is easy to use, and it supports the most important UML diagrams (use case, class, activity, object, communication, component, and deployment). However, Apollon is not fully diagramed, which is a weakness [9], [10].

The work in [11] presented a study at the Kaunas University of Technology for a course on information system design course that covered topics on UML. Because students were losing their motivation and engagement in the learning process, the researchers proposed a course that implemented gamified UML and a course to teach system design using a Moodle platform to overcome the problem. Moodle is an e-learning management system or e-learning tool that is free and open source. The course in information system design taught UML and principles of the Relational Unified Process (RUP), and it covered a wide selection of topics. They used plug-ins to extend the Moodle functionality and customize the course materials. The course was divided into 10 levels according to UML diagrams and RUP engineering disciplines to give more organization and interest to the course. The 10 levels were divided into two categories: syntax for UML diagrams and semantic levels for RUP. Based on a survey and their analysis, the interactive learning method that used gamified UML and taught the design course on the Moodle management system had a positive effect on students' grades. In addition, the gamified course succeeded in teaching the basics of UML, and students planned to continue using it because they enjoyed using it. However, this course had a small number of students, so the research should be extended to include more students. The benefits of Moodle platforms are that they are widely available, customizable, and open source. However, they were developed to deal with large projects (small and medium schools), and some students had difficulties (such as shutting down or blocking) when they took tests or accessed materials [11].

Students' and Professors' Experience with Modeling Tools

The work in [8] conducted a survey on students' experience with 31 UML tools for software modeling. The 117 participants who took the survey had used these modeling tools and studied software engineering courses in seven different countries: the United States, Brazil, the United Kingdom, Canada, Spain, China, and Denmark. The research focused on the features, weaknesses, and strengths of the UML tools, so professors and students could select tools that mattered the most to students. Then, the researchers focused on the nine UML tools that were

most heavily used, and they examined the tools' strengths and weaknesses in terms of feedback, use, installation, learning, cost, and the ability to draw diagrams. The researchers found that students chose the modeling tools based on being free, easy to install, easy to learn, whether they generated code, and whether they supported important notations. They also noted the students' complaints about a lack of feedback, diagrams that were hard to draw, and whether the programs were slow to use [8].

The following 31 modeling tools (were used in seven countries and provided 149 detailed responses from 117 students) have been selected carefully based on their popularity and successfulness in producing correct models that meet its targeted design. These tools are: ArgoUML, StarUML, MagicDraw UML, draw.io, Eclipse Modeling Tools, IBM Rational Software Modeler, jUCMNav, OmniGraffle, OSATE, PlantUML, Umple, Violet, Visio, Visual Paradigm, yED USA 15 Acceleo, Cacoo, draw.io, OSATE, Astah, Dia, LucidChart, Papyrus, Simulink, USE, yUML, Gliffy, and Edraw.

The following are the modeling tools that were most heavily used by students, with their benefits and drawbacks:

1. **StarUML:**
   a. Feature: provides full UML, ERD, DFD, cross-platform, actively maintained, model analysis, some code generation, and open source.
   b. Benefits: wide and easy to use.
   c. Drawbacks: the least ability to draw diagrams, only version 2010 is free and open source, and it needs improvement.

2. **Umple:**
   a. Features: supports full UML2 subset (class state), ERD, textual, diagrammatic modeling, cross-platform, model analysis, actively maintained code generation, free, and open source.
   b. Benefits: good feedback, very easy to use.
   c. Drawbacks: most buggy.

3. **MagicDraw:**
   a. Features: support full UML2, cross-platform, model analysis, actively maintained code generation.
   b. Benefits: wide to use.
   c. Drawbacks: not open source, not free, has a speed problem or slow to use, and needs improvement.

4. **ArgoUML:**
   a. Features: supports UML1.xsubset, cross-platform, model analysis, limited code generation, free, and open source but old: from 2014.
   b. Benefits: very easy to use.
   c. Drawbacks: Lack of feedback, least ability to draw diagrams, and needs improvement.

5. **Astah:**
    a. Features: supports UMLs, cross-platform, some model analysis, limited code generation.
    b. Benefit: very easy to use.
    c. Drawbacks: not free and not open source.

6. **USE:**
    a. Features: Textual modeling and more focus on class diagrams, OCL constraints, cross-platform, actively maintained, and model analysis.
    b. Benefits: Good feedback, free, and open source.
    c. Drawbacks: Least able to draw diagrams and least easy to use.

7. **Eclipse Modeling Tool:**
    a. Features: Supports full UML2, cross-platform, actively maintained, and model analysis.
    b. Benefits: perceived benefit in code generation.
    c. Drawbacks: Most complex among the nine tools, slow to use or has a speed problem, and the least easy to use.

8. **Papyrus:**
    a. Features: Supports full UML2, flagship Eclipse project, cross-platform, model analysis, actively maintained, code generation.
    b. Benefits: wide to use, free, open-source, and good feedback.
    c. Drawbacks: difficult to use, most buggy, and needs improvement.

9. **Visual Paradigm:**
    a. Features: Supports full UML2, cross-platform, code generation, model analysis, and is actively maintained.
    b. Benefit: wide to use.
    c. Drawbacks: not free, not open source, difficult to use, and needs improvement.

As we see in the details above regarding the strengths and weaknesses of each modeling tool, the most important for this research is that all the tools support UML modeling. There are some tools that are difficult to use, are not good at drawing diagrams, need improvements or updates, are not free, are not open source, and have speed problems. This is especially true of Eclipse modeling, the most complex tool among the nine listed. Moreover, ArgoUML and Astah do not provide feedback according to the students' complaints, and ArgoUML, StarUML, and USE provide the least ability to draw diagrams for students. Besides, MagicDraw and Eclipse have speed problems (they are slow to use), and Umple and Papyrus are the buggiest tools among the nine.

Conversely, some tools are easy to use, provide good feedback, are wide to use, free, and open-source. The tools perceived as the easiest to use were Umple, ArgoUML, Astah, and USE, while Eclipse modeling and Papyrus were perceived as the least easy to use. In addition, Payrus, Umple, and USE gave good feedback. All the factors of the modeling tools are important, such

as being easy to draw diagrams, giving feedback, ease of use, free, open-source, and speed. However, drawing diagrams and feedback are the most important factors in the modeling tools based on my research because UML modeling must be an interactive learning method that can help professors, students, and educators interact and learn from each other.

In 2016, Agner and Lethbridge [12] obtained research data about the advantages and disadvantages of modeling tools from 125 professors in 30 countries. All the professors had taught undergraduate software modeling for 5 years. The professors had used many tools, and the tool most used was ArgoUML, at 36.6%. Five other tools had been used by more than 20% of the professors: StarUML, Visual Paradigm, Papyrus, MagicDraw, and Astah. The advantages and benefits that motivated professors to use these tools were that they were easy to learn, free, and easy to install, and they supported the most significant notations. The easiest-to-use tools were Umple, PlantUML, and Visio, and the least complex tools to use were Visual Paradigm and Astah [12].

On the other hand, the most difficult to use were IBM Rational Rhapsody, BoUML, Papyrus, and The Acceleo, and the tools that received the most complaints were MagicDraw and Papyrus because they are more complex, according to the professors.

Kuzniarz and Martins [13] presented research in 2016 on techniques, methods, and tools used in teaching a course on model-driven software development (MDSD). Seven professors from five countries—Spain, Canada, Brazil, Poland, and Sweden—had taught MDSD courses and participated in this pilot study in which MDSD analyzed subjects such as UML modeling, requirements modeling, and model transformation. The findings identified 10 tools used to teach modeling in MDSD: ArgoUML, Acceleo, Astah, Eclipse Modeling, Papyrus, Visual Paradigm, OCL Editor, ATL, Enterprise Architect, and Rational Software. Visual Paradigm was used by three professors, and it was the tool most used of the 10 tools. Moreover, students worked on group projects to help them understand UML modeling concepts [13].

A survey study was presented by Reggio, Leotta, and Ricca in 2014 [14] on the use of UML diagram types. This research aimed to discover which UML diagram tools were known and used by the participants. Most survey participants were academic professors and Ph.D. students who complained about the UML complex [14]. Another study discussed practices by Paige, Polack, Kolovos, Rose, and Matragkas for teaching modeling. It showed that the tools did not provide feedback, lacked resources, and were complex to use, based on the researcher's experiences about the complaints that constrained the use of tools to teach modeling [15]. The literature showed that the benefits and drawbacks of all the modeling tools supported UML tools, which can be used for interactive learning and teaching by professors, students, and educators.

RESEARCH METHODOLOGY

This literature review was then analyzed by meta-ethnography, a method published in 1988 by Noblit and Hare [16]. Most approaches to qualitative synthesis are based on meta-ethnography. They are called systematic qualitative reviews, which draw the findings from individual research together. This method is an attempt to develop knowledge synthesis in an inductive and interpretive form as an educational synthesis that uses aggregation. The thematic approach

abstracted data and isolated elements in each study, and it failed schools to desegregate. Because abstraction minimizes the uniqueness of each site, the common findings in the context became a confounding factor in the study instead of providing an explanation of these findings as a contribution. Therefore, the synthesis did not give researchers and policy-makers an understanding of what went wrong and what could be done. To address these limitations, Noblit and Hare developed a distinct method for synthesizing qualitative studies that is interpretive rather than aggregative, and it was informed by Turner's theory of social explanation [16].

However, the interpretive explanation required developing an understanding of multiple cases, narratives, accounts, or studies. This is a long-term intensive study that includes interviews, observation, and analysis of documents, which concerned Noblit and Hare. To construct an approach to reducing these long-term intensive studies, Noblit and Hare developed an approach called meta-ethnography. This approach is used in most qualitative research because it provides a strict procedure for deriving substantive interpretations of any set of ethnographic or interpretive research. Consequently, Noblit and Hare identified seven phases that are compatible with other methods of synthesis but differ in their procedures and assumptions. Meta-ethnography phases may occur in parallel and overlap when they are observed in practice. The first difference is that the general approach of qualitative methods seeks to achieve an interpretive explanation by choosing the sample for the research to be related to the topic of interest rather than being exhaustive. The second difference seeks to reduce the account while keeping a sense of the account by using the explanations and interpretations in the original studies as data based on the selection and analysis of key metaphors [16].

The seven phases of Noblit and Hare's meta-ethnography are as follows:

1. **Getting started:** Identifying the focus of interest of research in a qualitative study.
2. **Deciding what is relevant to the initial interest:** Using relevant conferences, papers, and journal articles for literature and not being exhaustive.
3. **Reading the studies:** Reading the studies repeatedly to find the metaphors.
4. **Determining how the studies are related:** Listing the concepts or metaphors in the studies and finding their relations to each other.
5. **Translating studies into one another:** Comparing the concepts or metaphors.
6. **Synthesis translating**: Deciding whether there are common types of translation or if some translations or concepts can encompass others.
7. **Expressing the synthesis:** Proposing a name for the proposed synthesis, findings, or result.

Thus, this approach uses translation processes with the concepts, metaphors, and their interrelationships and compares one account with another account. The objective of the translation process is to maintain the original meaning and contextualization because it is idiomatic and concentrates on translating the meaning of the text instead of a literal translation. This is the third difference. Thus, there are three potential types of relationships defined by Noblit and Hare to lead translation and subsequent synthesis:

a. **Reciprocal translation:** This assumption applies when the concepts (accounts) of the studies are directly similar and comparable [16] [17].
b. **Refutational translation:** This assumption applies when the concepts (accounts) may conflict or are in relative conflict with each other [17].
c. **Line of argument:** This assumption applies when the concepts or accounts are not directly comparable and similar or do not conflict with each other. This line of synthesizing the argument is based on interference [17].

So, both translation and synthesis involve constant comparison to determine how studies relate to each other by analyzing the text until a comprehensive understanding of the concept is achieved. Then, the synthesis is done. How translations are synthesized, and the product of this process depends on how studies relate to each other. Both translation and synthesis involve continuous comparative. analysis of texts until a comprehensive understanding of the phenomena is realized and the synthesis is then complete.

RESULTS AND DISCUSSION

Getting Started

This research's main topic is to identify and synthesize modeling tools and determine if they support interactive methods for teaching and learning UML modeling tools. The information comes from various related studies in different conference papers and journal articles. The findings of these studies help professors with teaching and students with learning UML modeling in an efficient way.

Deciding What is Relevant to the Initial Interest

The studies that are relevant to this research are related only to tools used for teaching and learning UML modeling. Thus, all the text articles in the literature review came from several conference papers and journal articles associated with UML modeling. We used the following keywords for searching: UML modeling, tools used for UML modeling, and interactive learning methods for teaching UML modeling. As a result, many articles were found in the search process. Then, based on the literature review, the standards must be related to modeling tools. The next step used the results that came from the literature screening.

Reading the Studies

Many articles related to UML modeling tools and how they support interactive learning methods have been studied and read repeatedly and traced precisely. Consequently, the interpretative metaphors in the literature are in the form of concepts based on the modeling tools used for teaching and learning UML modeling. Thus, the explanation of each study concept (metaphor) can be used as a success factor or tool for teaching and learning UML modeling.

Determining how the Studies are Related

To identify the relationships among the studies and how they are to be synthesized, the studies must be put together [16]. To understand the relationship of each study by the explanation that has been done in the literature, the comparisons were done on the concept (tools) across several studies. Therefore, the relationship is reciprocal because there are comparable features and similarities between the studies.

Translating Studies into one Another and Synthesis Translating

Both the translating and synthesizing were done concurrently using meta-ethnography [16]. The translation step provides the concepts from one study and compares them to another study that has the same or similar concepts, even if they have different names, as mentioned in this process. The synthesis not only implies individual parts, but makes a whole into a common concept that includes other identified concepts. However, the studies in this paper included more than 11 synthesized success modeling tools found in many types of research and the synthesizing process defined above.

Expressing the Synthesis

This research is an expression of synthesis in which each modeling tool in Table 1 is suggested by the concepts in the studies. All modeling tools shown in Table 1 can be used for teaching and learning UML modeling. Some are used to support interactive learning, and they have strengths and weaknesses. There is no ranking for those tools; no one is less or more important than another.

Table 1 shows the summary of extracted tools.

No	Modeling Tool
1	StarUML
2	Umple
3	MagicDraw
4	ArgoUML
5	Astah
6	Eclipse Modeling Tool
7	USE
8	Papyrus
9	Visual Paradigm
10	Apollon
11	IBM Rational Rhapsody
12	BoUML
13	Acceleo
14	Moodle Platform

The findings in this research have used the methodology of meta-ethnography, which identifies and synthesizes the weaknesses and strengths of modeling tools for students to learn and professors to teach, based on the modeling tools used to support interactive learning.

First, SartUML provides full UML, ERD, DFD, open-source, cross-platform, wide, easy-to-use, and code generation, but it has the least ability to draw diagrams and needs improvement because only the 2010 version is open-source and free. This tool is good for professors and students who want a tool to support full UML modeling, free, open-source, and easy to use. It is not for those who want an updated tool with more ability to draw diagrams. Second, Umple supports full UML2, ERD, textual, cross-platform, open source, free, actively maintained code generation, and good feedback, and it is one of the easiest modeling tools to use. However, it is also the buggiest tool, which is not good for those who do not like buggy tools.

Third, MagicDraw provides full UML2, cross-platform, model analysis, actively maintained code generation, and wide and easy to use, but it is not open source, not free, slow to use, and needs improvement. Thus, it has several drawbacks that are not good for professors, students, and educators who are looking for these features.

Fourth, ArgoUML supports UML1, cross-platform, model analysis, limited code generation, and free and open-source, and it is one of the easiest modeling tools to use. But it is an old version, from 2014, lacks feedback, is not very good at drawing diagrams, and needs improvement. It is not good for professors who want to use it in an interactive learning method because of the lack of feedback and the issue of drawing diagrams. These are important features that support the learning method.

Fifth, Astah is used for UMLs, cross-platform, some model analysis, and limited code generation, and it is one of the easiest to use. However, it is not good for those looking for a free and open-source modeling tool.

Sixth, the Eclipse Modeling tool supports full UML2, cross-platform, perceived benefit in code generation, and model analysis. On the other hand, it is one of the most complex among the modeling tools. It has speed problems, and it is one of the least easy to use. It is not good for professors and students who are searching for these features.

Seventh, the USE modeling tool is one of the least able to draw diagrams and the least easy to use, but it has good features, such as textual modeling and more focus on class diagrams, OCL constraints, cross-platform, actively maintained, provides model analysis and good feedback. It is free and open source, which makes it a good choice for professors and students. It can support interactive learning because of its good features, especially feedback, free, and open source.

Eighth, Papyrus provides full UML2, cross-platform, model analysis, actively maintained, code generation, wide-to-use, free, open-source, and good feedback. This makes it very good to use, and it supports interactive learning. On the flip side, this tool is difficult to use, needs improvement, and is one of the buggiest of the modeling tools.

Ninth, the Visual Paradigm tool is difficult to use, not free, not open source, and needs improvement, but it provides full UML2, is wide to use, cross-platform, code generation, model analysis, and is actively maintained.

Tenth, the Apollon tool is an open-source, lightweight, free web application. It is easy to use, and it supports the most important UML diagrams (use case, class, activity, object, communication, component, deployment diagrams). This tool is a very good modeling tool that is used for interactive learning in software engineering, which has improved the learning and outcomes of students by up to 87% and increased their motivation for using modeling. But Apollon tool is not fully diagramed, which is a weakness.

Eleventh, the Moodle platform is free, available, customizable, and open source, but it is developed to deal with big projects (small and medium schools), and it has some troubles (shut

down or blocking) when students take tests or access materials. The Moodle platform was used for a gamifying course that presented an implementation of gamified UML and an approach for teaching information systems design. The survey results and the analysis show that it is an interactive learning method that uses gamified UML. Teaching the design course with the Moodle management system had a positive effect on student grades, and students planned to continue using it in the future because they enjoyed using it. Last, IBM Rational Rhapsody, BoUML, and the Acceleo modeling tools are the ones most difficult to use, according to the professors. On the other hand, they are wide to use. and students worked on group projects to help them understand UML modeling concepts.

All the above modeling tools support UML modeling, which helps professors in teaching and students to learn UML modeling. These modeling tools are used by students to work on group projects to understand UML modeling concepts. Some of them are used in interactive learning in software engineering and courses in information systems design, and they have improved students' learning and outcomes. As we discussed, the modeling tools' weaknesses and strengths can help professors, students, and educators choose the right modeling tool based on their needs. Drawing diagrams, providing feedback, being easy to use, and being free are the most important factors in modeling tools because UML modeling must use interactive learning, so professors, students, and educators can interact and learn from each other.

CONCLUSION

Modeling is the fundamental process in software engineering in the corner stone for developing computerized systems [19-22].

The purpose of this study was to identify, synthesize, and discuss the strengths and weaknesses of modeling tools, especially those that support UML modeling and interactive learning methods in courses that use modeling tools, such as "Software Engineering and Information Systems Design." The objectives were to guide both professors and students in choosing the proper modeling tools for their software engineering courses and other courses that use modeling as an interactive learning method. As a result of this research, we have identified many successful modeling tools. These tools can help students use what is suitable for learning UML modeling and contribute to engaging students in modeling. Besides, this research used the meta-ethnography method for synthesizing qualitative results in the software engineering area, especially modeling.

References

1. Bruegge, B., Krusche, S., & Alperowitz, L. (2015). Software engineering project courses with industrial clients. *ACM Transactions on Computing Education (TOCE)*, *15*(4), 1-31.

2. Whitehead, J. (2007, May). Collaboration in software engineering: A roadmap. In *Future of Software Engineering (FOSE'07)* (pp. 214-225). IEEE.

3. R. F. Paige, F. A. Polack, D.S. Kolovos, L. M. Rose, N. Matragkas, and J. R. Williams, "Bad modeling teaching practices." in *Proceedings of the ACM/IEEE 17th International Conference on Model Driven Engineering Languages and Systems—Educators Symposium* (Educators' Symposium@MODELS) (2014)

4. B. Crawford, and M. Cullin, "Supporting prospective teachers' conceptions of modelling in science," *International Journal of Science Education, 26*(11), pp. 1379–1401, 2004.

5. C. Schwarz, B. J. Reiser, E. A. Davis, L. Kenyon, A. Achér, D. Fortus, et al. "Developing a learning progression for scientific modeling: Making scientific modeling accessible and meaningful for learners," *Journal of Research in Science Teaching, 46*(6), pp. 632–654, 2009. https://doi. org/10.1002/tea.20311

6. C. T. Nicolaou, and C. P. Constantinou, "Assessment of the modeling competence: A systematic review and synthesis of empirical research," *Educational Research Review, 13*(3), pp. 52–73, 2014.

7. S. Akayama, B. Demuth, T. C. Lethbridge, M. Scholz, P. Stevens, and D. R. Stikkolorum, "Tool use in software modelling education." in *Proceedings of the ACM/IEEE 16th International Conference on ModelDriven Engineering Languages and Systems—Educators Symposium*, Miami, USA, 2013.

8. Agner, L. T., Lethbridge, T. C., & Soares, I. W. (2019). Student experience with software modeling tools. *Software & Systems Modeling*, *18*(5), 3025-3047.

9. S. Krusche, N. von Frankenberg, L. M. Reimer, and B. Bruegge, "An interactive learning method to engage students in modeling," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training* (pp. 12–22), June 2020.

10. S. Krusche, N. von Frankenberg, and S. Afifi, "Experiences of a software engineering course based on interactive learning," in *SEUH*, pp. 32–40, 2017.

11. M. Jurgelaitis, V. Drungilas, and L. Čeponienė, "Gamified Moodle course for teaching UML," *Baltic Journal of Modern Computing*, *6*(2), pp. 119–127, 2018.

12. L. T. Agner, and T. C. Lethbridge, "A survey of tool use in modeling education," in *Proceedings of the ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pp. 303–311, 2017.

13. L. Kuzniarz, and L. E. G. Martins, "Teaching model-driven software development: A pilot study," in *Proceedings of the ITiCSE Working Group Reports,* New York: ACM, 2016, pp. 45–56.

14. Reggio, G., Leotta, M., & Ricca, F. (2014, September). Who knows/uses what of the UML: A personal opinion survey. In *International Conference on Model Driven Engineering Languages and Systems* (pp. 149-165). Springer, Cham.

15. R. F. Paige, F. A. Polack, D. S. Kolovos, L. M. Rose, N. Matragkas, and J. R. Williams, "Bad modelling teaching practices" in *ACM/IEEE 17th International Conference on Model Driven Engineering Languages and Systems—Educators Symposium*, Valencia, Spain, 2014.

16. G. W. Noblit, and R.D. Hare, *Meta-Ethnography: Synthesizing Qualitative Studies.* London: SAGE Publications, Inc., 1988.

17. M. Edwards, M. Davies, and A. Edwards, "What are the external influences on information exchange and shared decision-making in healthcare consultations: A meta-synthesis of the literature," *Patient Education Counsel, 75*, pp. 37–52, 2009.

18. P. J. Clarke, and A. Pierantonio, "Teaching modeling: A software perspective. *Computer Science Education*, *28*(1), pp. 1–4, 2018.

19. O. Elfaki, S. Phon-Amnuaisuk, , &C.K. Ho: Modeling variability in software product line using first order logic.

In *2009 Seventh ACIS International Conference on Software Engineering Research, Management and Applications* (pp. 227-233). IEEE, 2009.

20. Y Duan, C Cruz, AO Elfaki, Y Bai, W Du: Modeling value evaluation of semantics aided secondary language acquisition as model driven knowledge management." In *Computer and Information Science*, pp. 267-278. Springer, Heidelberg, 2013.

21. Y Duan, Q Duan, AO Elfaki, C Ren: Formalizing over design and under design of value engineering for context-aware cloud service development. In *2015 3rd International Conference on Future Internet of Things and Cloud* (pp. 72-79). IEEE.2015.

22. A Elfaki, A Aljaedi, Y Duan: Mapping ERD to knowledge graph. In *2019 IEEE World Congress on Services (SERVICES)* (Vol. 2642, pp. 110-114). IEEE.2019.