Professional Use and Teaching of UML: Perceptions of Brasília-DF Region Software Practitioners and Higher Education Instructors

Edna Dias Canedo, Edson OliveiraJr, Heloise Acco Tives

[1]University of Brasília (UnB), Department of Computer Science, Brasília, DF, Brazil
State University of Maringá, Informatics Department, Maringá, PR, Brazil
Federal Institute of Paraná (IFPR), Palmas, PR, Brazil
e-mail: ednacanedo@unb.br, edson@din.uem.br, heloise.acco@ifpr.edu.br

Abstract. *Context: UML remains a widely recognized modeling language in software engineering education and industry. However, its adoption and teaching effectiveness face challenges related to complexity, time constraints, and alignment with industry practices. Understanding how UML is currently taught and used in professional settings is important for improving its role in software development. Goal: This study investigates how UML is taught in higher education and used in industry in Brasília-DF, identifying the most emphasized diagrams, key motivations for its adoption, and the challenges faced by educators and practitioners. Methods: We conducted a survey with 21 instructors and 51 software practitioners to analyze which UML diagrams are most frequently taught and used, how UML is applied in software projects, and what factors influence its adoption or rejection. The study combined quantitative analysis of closed-ended questions and qualitative coding of open-ended responses to provide a comprehensive perspective. Results: The findings indicate that class, use case, and sequence diagrams are the most commonly taught and emphasized in academia. Instructors value UML for its role as a communication standard and learning tool, but complexity, lack of time, and tool availability are major barriers to effective teaching. In industry, UML is primarily used for problem modeling and system flow representation, but semantic complexity, difficulty in maintaining updated documentation, and lack of training limit its widespread adoption. Many companies prefer simpler alternatives, such as flowcharts and data flow diagrams.*

1. Introduction

The Unified Modeling Language (UML) is a standardized visual modeling language widely used in software engineering to specify, visualize, construct, and document the artifacts of complex systems [Ciccozzi et al. 2019]. It provides a common notation that spans from software analysis to deployment, enabling developers to clearly and concisely represent both the structure and behavior of a system. It facilitates communication among team members and other stakeholders, promoting a shared understanding of the system under development [Koç et al. 2021]. Using UML, developers can visualize the system's structure, including classes, objects, relationships, and behaviors. This simplification aids in comprehending the system and enhances communication among team members and stakeholders [Metzner 2024].

Before writing a single line of code, developers can utilize UML to create class and component diagrams, effectively planning the system's architecture. This approach aids in defining the relationships between various system components, ensuring a robust foundation for development [Apostol et al. 2024]. Moreover, UML facilitates the identification and application of common design patterns within projects. By employing sequence and collaboration diagrams, developers can comprehend object interactions across different scenarios, thereby easing the integration of pertinent design patterns [Tavares et al. 2021].

UML diagrams serve as visual documentation of the system, providing a comprehensive overview of the software. This clarity aids developers in understanding the designed and implemented components, facilitating the integration of new team members by offering a visual representation of the code [Fernandes and Amaral 2022]. Modeling the system with UML can help identify design issues before coding begins. By simulating system behavior using sequence diagrams, for example, developers can anticipate potential logical problems and address them proactively, preventing them from becoming actual bugs [Kaur et al. 2023]. In the educational context, UML plays an important role in training future software practitioners. It is taught in various higher education institutions as a fundamental part of computer science curricula, preparing students to tackle the challenges of modeling and developing complex systems [Kuzniarz and Staron 2006, Apostol et al. 2024]. Understanding how UML is taught in higher education institutions and applied in the software development industry is important to align educational practices with the needs of the software industry, ensuring that graduates are equipped to utilize this tool in their professional environments effectively.

In this context, this study aims to investigate the perceptions of instructors from Higher Education Institutions (HEIs) in Brasília-DF and the region regarding how UML is being taught and practitioners' perceptions about its use in the local software development industry. By analyzing the pedagogical approaches adopted by instructors and the practices applied by industry practitioners, the study seeks to identify potential gaps between academic teaching and the demands of the software industry. The goal is to propose improvements to the curricula of UML-related courses, fostering a more relevant education that aligns with the actual needs of software development.

The main findings of this research were as follows. Use Case Diagrams, Class Diagrams, and Sequence Diagrams are the most commonly covered by instructors in the classroom. Most of them use a CASE tool for UML modeling to support learning. The motivation for teaching UML is to enable students to understand software features. In contrast, the main limitation in teaching UML is the difficulty in integrating it with programming courses and the challenges students face in understanding the diagrams. According to the surveyed instructors, UML is widely used in the software development industry. The Class Diagram, Use Case Diagram and Sequence Diagram are the most used by practitioners who learned to use a CASE tool during their UML teaching in undergraduate courses. From the practitioners' perspective, UML modeling teaching should be improved with industry examples. The motivation for software companies to use UML is that modeling allows for a better understanding of the project as a whole. However, practitioners believe that UML has semantic complexity and that it is difficult to fully understand and master them.

This paper is organized as follows: Section 2 presents related work; Section 3 provides our study settings; Section 4 presents results of this study; Section 5 discusses threats to validity; and Section 6 concludes this paper.

2. Related Work

[Apostol et al. 2024] investigated the differences between specific groups of students, future software engineers, who worked with different UML diagrams, and another group that performed the activities without using UML diagrams. The authors observed the time, code correctness, and understanding of the project from the design phase to the development phase of the product. The authors concluded that UML diagrams played a vital role in the design, analysis, and documentation of software systems, with class diagrams standing out as a key tool for representing the architecture and behavior of the software.

[Ciccozzi et al. 2019] conducted a systematic literature review to identify trends, technical characteristics, evidence, and limitations of current solutions for executing UML models in both research and industrial practice. The authors identified 63 studies and 19 tools and concluded that there is a growing scientific interest in executing UML models; solutions providing translational execution clearly outperform interpretative solutions; model-level debugging is supported in very few cases; only a few studies provided evidence regarding the use of UML in the software industry; and the most common limitation is dealing with the coverage of UML diagrams.

[Huber and Hagel 2022] conducted a systematic literature review and identified various tools for teaching UML diagrams in higher education for software engineering. These tools offer distinct functionalities: 1) Visualization tools allow students to interact with UML diagrams and observe dynamic changes during code execution, as seen in tools like JaguarCode and JavalinaCode. and 2) Comparison tools automate the assessment of UML diagrams by comparing them with reference solutions provided by educators. Some, such as ModBud and UMLGrade, offer automatic feedback, while others, such as DUDE, leverage neural networks for analysis. Additionally, the authors identified five major challenges in teaching UML: 1) Difficulty in learning and applying UML – Most studies acknowledge that object-oriented design is challenging for students to grasp and implement correctly. 2) Existing tools are inadequate – Nearly half of the publications highlight that available tools are either insufficient or overly complex for educational purposes, often lacking essential functionalities for UML instruction; 3) Grading exercises are time-consuming for educators – Evaluating UML diagrams requires significant effort from instructors. 4) Lack of individualized and direct feedback – The high number of students makes it difficult for educators to offer personalized feedback. 5) Multiple correct solutions complicate assessment – Since UML problems often allow for multiple valid solutions, grading diagrams manually becomes a challenge.

[Farias et al. 2018] identified the state of practice in the Brazilian software industry regarding using UML in real-world scenarios. The authors conducted a survey with 222 practitioners from 140 different Information Technology companies in Brazil to investigate their experiences with UML, as well as the difficulties in adopting UML and what should be done to increase its adoption in practice. The authors found that 28% of participants used UML in their daily work, while 73% did not. Additionally, 55% of survey participants stated that UML is the official language for software modeling; 61% reported that the automatic generation of UML diagrams to represent an overview of the system under development would be useful for driving UML usage. Furthermore, the authors' findings revealed that practitioners do not frequently use UML.

[Júnior et al. 2021] also investigated how UML is used in practice within the

Brazilian software industry. The authors conducted a survey with 314 practitioners from 180 IT companies to explore the factors that affect the use, difficulty, and frequency of use, perceived benefits, and contextual factors that hinder UML diagram adoption. In addition, the authors conducted 20 semi-structured interviews with software industry practitioners. The results showed that participants recognize the utility of UML models, such as their ability to improve understanding of the integration between corporate systems. However, 74% of participants do not use UML diagrams due to factors such as continuous delivery practices, time constraints, lack of knowledge about modeling, company culture, and the ongoing difficulty of keeping models up to date and synchronized.

[OliveiraJr et al. 2021] investigated the teaching, learning, and professional use of UML in Maringá and its surrounding region through two surveys: one with 23 instructors examining how UML is taught in higher education institutions. The findings indicate that UML is widely taught in these institutions, primarily as part of Software Engineering courses, with a strong emphasis on class, use case, and sequence diagrams. However, UML education remains more aligned with traditional software development processes than agile methodologies, which dominate the industry. Many professionals learned UML during their undergraduate studies, but its practical application is limited, primarily used for documentation and project communication. The main challenges include a lack of in-depth knowledge, low industry appreciation for modeling, and insufficient time for practical implementation. The authors suggest stronger collaboration between academia and industry, incorporating more real-world examples in education and focusing on the UML diagrams most relevant to professional practice.

[Choma Neto et al. 2021] investigated whether the teaching of UML in educational institutions in São Carlos, SP, meets the needs of IT companies in the region. The authors conducted a survey with IT professionals from 23 companies, focusing on the types of diagrams used, frequency of use, and challenges faced when using UML diagrams. The results revealed a mismatch between what is taught and market practices, with companies using only a few UML diagrams while others prefer agile approaches. The study suggests updating IT curricula to align UML teaching with current industry demands. Similarly, [Guelman et al. 2023] also investigated the teaching, learning, and professional use of UML in Belo Horizonte and its surrounding area by replicating the study by [OliveiraJr et al. 2021]. The authors conducted two surveys, one with 13 instructors and another with 33 IT professionals. The research analyzed how UML is taught, learned, and applied in professional practice. The results indicate discrepancies between academic teaching and industry practices, suggesting curricular adaptations in local IT programs, similar to the findings of the previous study.

3. Study Settings

Our objective was to investigate instructors' perceptions of how UML is being taught in undergraduate courses and practitioners' views on the pedagogical approaches adopted by instructors, as well as how these approaches are applied in the software industry. We refined this objective into the following research questions (RQs):

RQ.1: What teaching practices do instructors at higher education institutions in Brasília adopt to teach UML?
This question aims to identify how instructors approach UML diagrams and whether they use any CASE tools to support modeling practice. Additionally,

it explores their motivations and the challenges they face regarding using UML diagrams in teaching.

RQ.2: How did practitioners learn UML, and how do they use it in the software development industry in Brasília?
With this question, we seek to better understand how practitioners learned UML during their education and how they apply UML diagrams in their daily work in the software industry.

Our study focuses on instructors who teach modeling courses in computer science programs and practitioners working in the software development industry in Brasília. To ensure the accuracy of our target audience, we included two control questions in the questionnaire: one to confirm participants' profiles and another to gather information on their experience—either their years of teaching in higher education (for instructors) or their years of experience in software development (for practitioners). The purpose of these questions was to ensure that we collected responses only from individuals who belong to our intended study group.

We provided participants with clear information about the study, including its objectives, methodology, and how their data would be handled. Additionally, we emphasized the conditions and stipulations governing participation. Participants were informed that their participation was entirely voluntary and that they were free to decline participation or withdraw their consent at any time without facing any penalties. They were also assured of confidentiality and the researchers' responsibility to uphold these conditions and stipulations. Furthermore, we provided the researchers' contact information for any inquiries. This information was included in the Informed Consent Form (ICF), which was presented at the beginning of the questionnaire. All procedures adhered to ethical privacy standards following the General Data Protection Law (Lei nº 13.709/2018) [Brasil 2018], ensuring the confidentiality of participants' information. No personally identifiable information was disclosed or shared. All responses were handled with the utmost care and used exclusively for research purposes. All survey questions and their respective response options are available at Zenodo.

Before conducting the research, we conducted a pilot test [Kitchenham and Pfleeger 2008] with three ICT instructors and three practitioners who were not involved in the study to review the questions and response options. Since no improvements were suggested, the questions were validated. The pilot participants took an average of 10 minutes to complete the survey, and this estimated completion time was communicated to the survey participants when the survey was made publicly available. The pilot responses were not included in the data analysis. We hosted the survey on Google Forms and shared it through cards and text on social media. We used posts and direct messages on Twitter, LinkedIn, Facebook, Instagram, and WhatsApp.

We adopted a mixed-methods approach, combining qualitative research with quantitative analysis through visualizations and correlations. For the qualitative analysis, we applied Grounded Theory procedures, specifically open and axial coding [Corbin and Strauss 2008], to gain a deeper understanding of the data. For the coding process, all authors contributed to open coding, with each focusing on at least one question from the survey instrument. Following this, we conducted axial coding with all authors involved in this phase. This step provided a more in-depth analysis of the initial categories, helping to establish relationships between the codes and create a more coherent and integrated structure of the identified categories and subcategories. Finally, each open-ended question was reviewed by at least two authors. Figure 1 illustrates our coding process. The example shows how the response from participant #R12 was analyzed. Specifically, this participant commented on what motivates them to use UML diagrams for teaching modeling in their course (Q23). Figure 1 shows the category and subcategory created from the raw data of #R12's response. The complete coding of all open-ended questions is available on Zenodo.

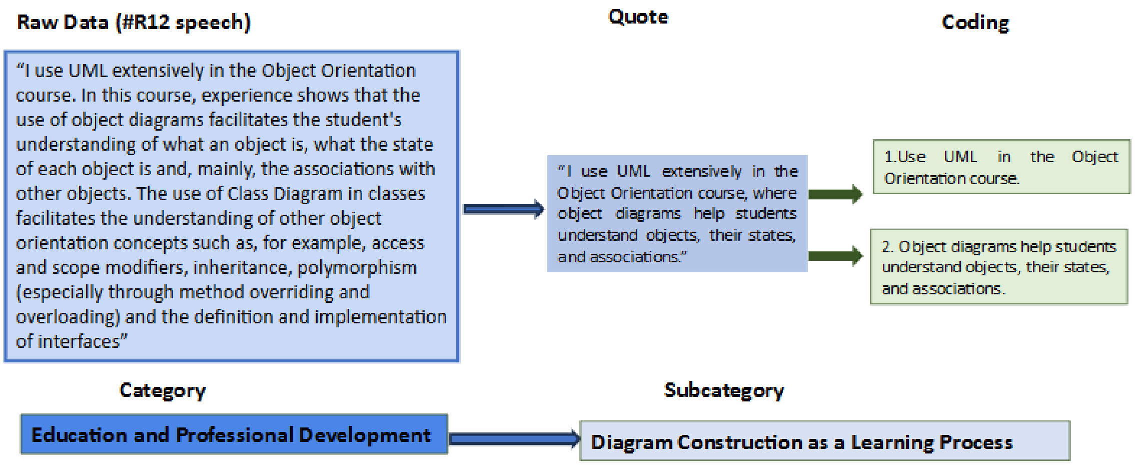
Raw Data (#R12 speech)
Quote
Coding
"I use UML extensively in the Object Orientation course. In this course, experience shows that the use of object diagrams facilitates the student's understanding of what an object is, what the state of each object is and, mainly, the associations with other objects. The use of Class Diagram in classes facilitates the understanding of other object orientation concepts such as, for example, access and scope modifiers, inheritance, polymorphism (especially through method overriding and overloading) and the definition and implementation of interfaces"
"I use UML extensively in the Object Orientation course, where object diagrams help students understand objects, their states, and associations."
1.Use UML in the Object Orientation course.
2. Object diagrams help students understand objects, their states, and associations.
Category
Subcategory
Education and Professional Development
Diagram Construction as a Learning Process

Figure 1. Coding Process Overview

4. Study Results

4.1. RQ.1

Twenty-one instructors participated in the survey. Most of them have over 10 years of experience in higher education (43%), and 53% reported having more than 5 years of experience teaching UML. Among the participants, 18 were men, and only three were women instructors. 48% of the participants hold a Ph.D., with 11 working at public institutions and 10 at private institutions in Brasília. 81% of the instructors teach in in-person courses, while 19% teach in distance education courses. 52% of the instructors include UML as part of the syllabus in multiple courses. The average annual workload dedicated to teaching UML is 60 hours.

The diagrams most taught in the courses by the instructors are use case diagrams, class diagrams, sequence diagrams, activity diagrams, object diagrams, state diagrams, and component diagrams. The instructors stated that the three diagrams they emphasize the most during their classes are Class diagrams, Use case diagrams, and Sequence diagrams, as shown in Figures 2 and 3. The instructors also stated that they use and/or teach the use of CASE tools for UML modeling in their courses, such as Jude/Astah and Draw.io. In addition, 61% of the instructors teach UML in the context of traditional software development (waterfall or similar) and agile (Scrum), and 90.5% of them consider it important to teach how to develop a software architecture using UML as the modeling language.

Regarding the motivations for using UML diagrams in teaching modeling in their courses, we identified the categories Education and Professional Development, and the

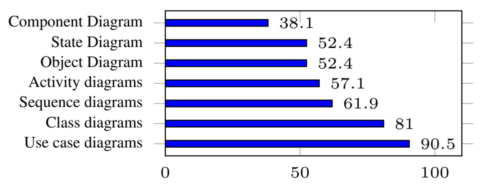

Figure 2. Diagrams taught by the instructors

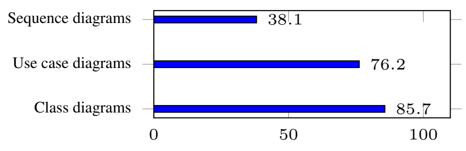

Figure 3. The three diagrams that instructors emphasize most in their lessons.

most mentioned subcategories by the participants were Communication Standard in Software Development (9 mentions), Diagram Construction as a Learning Process (5 mentions), and Market-Driven Learning (3 mentions).

As for the limitations of using UML diagrams in teaching modeling in their courses, we identified the categories of Complexity of UML, Lack of Time, and Tools. 12 participants mentioned the Complexity of the UML category. Regarding the instructors' evaluation of UML's use in the software development industry today, we identified two categories: Limited or Rare Use and Widespread Use, and five subcategories: Limited Industry Adoption (11 mentions), Use of Other Approaches (2 mentions), Lack of Updates to the Diagrams (1 mention), Importance of UML in Systems Modeling (5 mentions), and Documenting the Software Architecture (2 mentions). The complete codebook for the open-ended questions can be accessed at Zenodo.

RQ.1 Summary: The most taught UML diagrams were use case, class, sequence, activity, object, state, and component diagrams, emphasizing class, use case, and sequence. 61% of the instructors teach UML in both traditional and agile software development contexts and 90.5% value teaching software architecture with UML. Key motivations for using UML include communication standards, learning through diagram construction, and market demand, while the main limitations are UML complexity, time constraints, and tool availability.

4.2. RQ.2

Over 66% of the participants held a bachelor's degree, 11.8% had a master's degree, and 21.6% were specialists. Most participants work as software developers, with 43% having more than five years of professional experience in the field. Regarding development methodologies, 19.6% work in traditional software development (waterfall or similar), 51% in Agile (Scrum), and 29.4% in both traditional and Agile models. The majority learned class diagrams, use case diagrams, object diagrams, sequence diagrams, and activity diagrams during their undergraduate courses, with the three diagrams they were most exposed to being class diagrams, use case diagrams, and sequence diagrams.

33.3% of them stated that they learned UML in a dedicated course, 29.4% as part of a Software Engineering course, and 37.3% through content spread across multiple courses. Additionally, the average UML teaching workload in their education was 60 hours. The practitioners learned to use the tools Jude/Astah and Draw.io. This result aligns with the findings from the survey with instructors. It validates their responses, as practitioners working in Brasília reported learning the same tools and diagrams mentioned by the instructors.

45% of the participants stated that they have basic experience with UML notation and model software only at the level of the most common UML elements, such as classes and inheritance. Additionally, the most commonly used diagrams in their organizations are use case diagrams, class diagrams, activity diagrams, and sequence diagrams.

Regarding the aspects that need improvement in UML modeling education, participants identified areas where UML teaching could be enhanced to make learning more effective and aligned with industry needs. Four main categories emerged: 1) Industry Examples (16 mentions) – This was the most frequently mentioned category, emphasizing the importance of using practical and applied teaching methods that allow students to see the real-world relevance of UML; 2) Adapting Teaching to Industry Trends (6 mentions) – Participants suggested a greater focus on modeling approaches aligned with agile methodologies, reflecting the increasing adoption of frameworks like Scrum and Kanban in the software industry; 3) Interconnections Between Diagrams (1 mention) – Some participants highlighted the need for a deeper understanding of how different UML diagrams relate to each other; and 4) Time Allocation (1 mention) – Participants also suggested allocating more instructional hours to ensure a more solid understanding of UML modeling. These findings highlight the need to make UML education more practical and aligned with real-world software industry practices, as presented in Table 1.

The factors that drive companies to adopt UML are primarily related to professionals' prior knowledge and the benefits UML offers for modeling and project planning. We identified the following categories: 1) Ease of Understanding System Flow (9 mentions) – UML is seen as a valuable tool for visually representing system logic, facilitating communication between teams; 2) Problem Modeling (9 mentions) – Another key motivation cited by participants is UML's ability to represent challenges and solutions in a structured way; 3) Widespread Knowledge Among Professionals (3 mentions) – UML's status as an industry-standard notation encourages its adoption; 4) Development Time Estimation (2 mentions) – Some companies use UML to assist in planning and estimating the time required for project implementation; 5) Updated Documentation (1 mention) – Practitioners recognize UML as a means of maintaining clear and accessible documentation; and 6) Availability of Supporting Tools (1 mention) – Some practitioners mentioned the existence of UML support tools as a motivating factor for its use. These results reveal that clarity in representing system flows and problem modeling are the primary drivers for UML adoption in companies, as shown in Table 1.

In addition to UML, companies also use other languages for software modeling, with flowcharts being the most popular alternative: 1) Flowchart (11 mentions) – The most frequently cited option, used to represent processes and decisions in a simplified manner; 2) Data Flow Diagram (2 mentions) – Used to illustrate the flow of data within a system; and 3) Block Diagram (2 mentions) – Applied to represent system components and their interactions. In summary, flowcharts are widely used as a simpler alternative to UML, as presented in Table 1.

Challenges Faced by Practitioners in Using UML The challenges reported by practitioners relate to aspects such as complexity, documentation maintenance, and lack of knowledge. We identified the following categories: 1) Semantic Complexity (12 mentions) – The most frequently reported difficulty was the complexity of understanding UML diagrams, which can hinder their adoption and effective use; 2) Documentation Maintenance (5 mentions) – Professionals highlighted the challenge of keeping UML documentation up to date, which can become an obstacle in agile development; and 3) Lack of Knowledge (4 mentions) – A lack of adequate training was cited as a limiting factor for the effective use of UML in companies. These findings indicate that the semantic complexity of UML diagrams is the most significant challenge faced by practitioners, followed by difficulties in maintaining updated documentation, as presented in Table 1.

Table 1. Coding of open-ended questions regarding the Benefits and Challenges of UML in Software Development.

What should change/improve in teaching UML modeling?		
Category	**Sub-category**	**Cited#**
1.Industry Examples	Practical and applied learning methods	16
2. Adapting Teaching to Industry Trends	Agile-centric software education	6
3. Interconnections Between Diagrams	Understanding the interconnection between diagrams	1
4. Time Allocation	More hours for learning	1
What motivates your company to use UML?		
Category	**Sub-category**	**Cited#**
1. Ease of Understanding System Flow	Ease of Understanding System Flow	9
2. Problem Modeling	Problem Modeling	9
3. Widespread Knowledge Among Professionals	Widespread Knowledge Among Professionals	3
4. Development Time Estimation	Project planning	2
5. Updated Documentation	Documentation up to date	1
6. Availability of Supporting Tools	Tools available	1
What other Software Modeling Language do you use in your company		
1. Flowchart	Flowchart	11
2. Data flow diagram	Data flow diagram	2
3. Block diagram	Block diagram	2
What difficulties do you, as a professional in the field, observe in using UML?		
1. Semantic Complexity	Understanding UML Diagrams	12
2. Documentation Maintenance	Keep information up to date	5
3. Lack of Knowledge	Lack of knowledge	4

RQ.2 Summary: Most participants have a bachelor's degree and over five years of experience, working mainly with Agile development. UML was primarily learned in Software Engineering courses, with exposure to class, use case, and sequence diagrams. The main challenges reported are semantic complexity, documentation maintenance, and lack of knowledge. Companies adopt UML to facilitate problem modeling and communication and use flowcharts as a simplified alternative.

5. Threats to Validity

Construct validity concerns whether the survey accurately measures the intended concepts. To mitigate this threat, we designed the questionnaire based on a literature review and piloted it with domain experts before distribution. However, there is a risk that some participants interpreted questions differently, leading to response variations. Additionally, while we aimed to cover key aspects of UML adoption and challenges, some relevant factors may not have been explicitly addressed.

Internal validity relates to whether confounding factors may have influenced the results. Since participation was voluntary, there is a possibility of self-selection bias, meaning that respondents with stronger opinions or experiences regarding UML were more likely to participate. Additionally, differences in teaching methodologies and industry practices across institutions and organizations could have influenced the participants' perspectives, potentially affecting the consistency of the results.

External validity addresses the generalizability of the findings. Our study included responses from 21 instructors and 51 practitioners, primarily from Brasília, which may limit how much the results apply to other regions or countries. Moreover, the sample size, while providing valuable insights, may not be fully representative of the broader population of software engineering educators and professionals. Future studies with larger and more geographically diverse samples could strengthen the generalizability of our findings.

Conclusion validity concerns the reliability of the relationships identified in the data. We used a mixed-methods approach to analyze the responses, combining quantitative analysis with qualitative coding of open-ended questions to ensure robustness. However, the relatively small sample size may limit the statistical power of some findings. To improve reliability, future research could incorporate additional validation methods, such as follow-up interviews or case studies, to triangulate the survey results.

6. Conclusion

This study explored the teaching and adoption of UML in academia and industry, analyzing the perspectives of instructors and practitioners. The findings indicate that use case, class, and sequence diagrams are the most commonly taught and emphasized in software engineering courses. Despite UML's role as a standardized communication tool, its complexity, time constraints in teaching, and tool availability remain key challenges for educators.

From an industry perspective, UML is valued for problem modeling, system flow representation, and team communication. However, its adoption is limited by semantic complexity, difficulties in maintaining updated documentation, and a lack of sufficient training. Many companies also use flowcharts and data flow diagrams as simpler alternatives.

To address the challenges identified, future research should focus on a) enhancing UML education by incorporating more industry-relevant examples and aligning teaching with Agile methodologies; b) developing practical guidelines for UML use in software projects, helping practitioners overcome adoption barriers; and investigating alternative modeling approaches that balance expressiveness and ease of use, making UML more adaptable to modern software development practices.

Data Availability

The materials produced during the research, including the survey form and the file containing all the survey responses, are available at https://zenodo.org/records/14814718.

Acknowledgments

Edson OliveiraJr thanks CNPq/Brazil grant #311503/2022-5.

References

Apostol, D.-C., Bogdan, R., and Marcu, M. (2024). Uml diagrams in teaching software engineering classes. a case study in computer science class. In *2024 IEEE 22nd World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pages 000327–000332.

Brasil (2018). Lei nº 13.709, de 14 de agosto de 2018. Lei Geral de Proteção de Dados Pessoais (LGPD). *Diário Oficial da República Federativa do Brasil, http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm.*

Choma Neto, J., Bento, L. H. T. C., OliveiraJr, E., and Souza, S. d. R. S. d. (2021). Are we teaching uml according to what it companies need?: a survey on the são carlos-sp region. In *Sociedade Brasileira de Computação*, pages 34–43, https://sol.sbc.org.br/index.php/educomp/article/view/14469/1431.

Ciccozzi, F., Malavolta, I., and Selic, B. (2019). Execution of uml models: a systematic review of research and practice. volume 18, page 2313–2360, https://doi.org/10.1007/s10270-018-0675-4.

Corbin, J. and Strauss, A. (2008). Basics of qualitative research: Techniques and procedures for developing grounded theory. *Thousand Oaks*, 3:1–400.

Farias, K., Gonçales, L., Bischoff, V., Da Silval, B., Guimarães, E., and Nogle, J. (2018). On the uml use in the brazilian industry: A state of the practice survey. In *Proceedings - SEKE 2018*, Proceedings of the 30th International Conference on Software Engineering and Knowledge Engineering, SEKE, pages 372–375. Knowledge Systems Institute Graduate School.

Fernandes, M. A. and Amaral, V. (2022). A simulation framework for uml education. In *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 161–166.

Guelman, I., OliveiraJr, E., and Xavier, L. (2023). Ensino, aprendizagem e uso profissional da uml em belo horizonte e região. In *Simpósio Brasileiro de Educação em Computação (EDUCOMP)*, pages 49–58, https://doi.org/10.5753/educomp.2023.228173. SBC.

Huber, F. and Hagel, G. (2022). Tool-supported teaching of uml diagrams in software engineering education - a systematic literature review. In *2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*, pages 1404–1409.

Júnior, E., Farias, K., and Silva, B. (2021). A survey on the use of uml in the brazilian industry. In *Proceedings of the XXXV Brazilian Symposium on Software Engineering*, SBES '21, page 275–284, New York, NY, USA. Association for Computing Machinery.

Kaur, R., Malik, M. Z., and Singh, M. (2023). Improving delivery of uml class diagrams concepts in computer science education through collaborative learning. In *2023 IEEE Frontiers in Education Conference (FIE)*, pages 1–5.

Kitchenham, B. A. and Pfleeger, S. L. (2008). Personal opinion surveys. In Shull, F., Singer, J., and Sjøberg, D. I. K., editors, *Guide to Advanced Empirical Software Engineering*, pages 63–92. Springer.

Koç, H., Erdoğan, A. M., Barjakly, Y., and Peker, S. (2021). Uml diagrams in software engineering research: A systematic literature review. *Proceedings*, 74(1).

Kuzniarz, L. and Staron, M. (2006). Best practices for teaching uml based software development. In Bruel, J.-M., editor, *Satellite Events at the MoDELS 2005 Conference*, pages 320–332, Berlin, Heidelberg. Springer Berlin Heidelberg.

Metzner, A. (2024). Systematic teaching of uml and behavioral diagrams. In *2024 36th International Conference on Software Engineering Education and Training (CSEE&T)*, pages 1–5.

OliveiraJr, E., Colanzi, T., Amaral, A., Cordeiro, A., Neto, J. C., and Souza, S. (2021). Ensino, aprendizagem e uso profissional da uml em maringá e região. In *Anais do XXIX Workshop sobre Educação em Computação*, pages 328–337, 10.5753/wei.2021.15924. SBC, Porto Alegre, RS, Brasil.

Tavares, J. F., Costa, Y. M. G., and Colanzi, T. E. (2021). Classification of uml diagrams to support software engineering education. In *2021 36th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW)*, pages 102–107, 10.1109/ASEW52652.2021.00030.