ICIC Express Letters
Part B: Applications
Volume **16**, Number **6**, June **2025**

ICIC International ©2025 ISSN 2185-2766
pp. **619–627**

PERCEIVED CHALLENGES OF USING UML BY STUDENTS IN MULTICULTURAL GLOBAL SOFTWARE ENGINEERING TEAMS

Daniel Moritz Marutschke[1], Patricia Brockmann[2]
Tri Astoto Kurniawan[3], Mate Kovacs[4] and Victor Kryssanov[4]

[1]College of Global Liberal Arts
[4]College of Information Science and Engineering
Ritsumeikan University
2-150 Iwakura-cho, Ibaraki, Osaka 567-8570, Japan
{ moritz; kovacsm }@fc.ritsumei.ac.jp; kvvictor@is.ritsumei.ac.jp

[2]Faculty of Computer Science
Nuremberg Institute of Technology
Keßlerplatz 12, Nuremberg 90489, Germany
patricia.brockmann@th-nuernberg.de

[3]Faculty of Computer Science
Universitas Brawijaya
Jalan Veteran, Malang 65145, Indonesia
triak@ub.ac.id

Received September 2024; accepted November 2024

Abstract. *Working in multicultural distributed teams to develop software demands a combination of high technical skills and soft skills. With an increasingly globalized work environment, these stills are necessary for future engineers. Implementing a graduate-level university course – jointly taught at universities in two or more countries – poses several challenges, ranging from organizational and budgetary constraints to an increased effort from all involved instructors with different academic settings. One of the challenges – and the focus of this paper – is the use of UML within distributed teams as a communication and collaboration tool. A Global Software Engineering (GSE) Course was synchronously taught at three universities in Japan, Germany, and Indonesia. Students were surveyed before (ex-ante) and after (ex-post) the semester about their perceived ease of use and usefulness of UML, and collaboration benefit of UML. Challenges that students faced were collected after the completion of the projects. The results show disconnect between the perceived usefulness and intent to use UML, which is in accordance with previous research. Especially collaborative projects can benefit from UML usage. Germany with a low power-distance showed the least impact, international students in Japan the most.*

Keywords: Global Software Engineering education, Distributed teams, UML, Multicultural analysis

1. **Introduction and Background.** In the contemporary landscape of software engineering, the adoption of distributed teams across different geographical and cultural boundaries has become increasingly prevalent. This shift towards international distributed teams is driven by the need to leverage global talent, reduce costs, and enhance productivity through round-the-clock development cycles. However, this globalization introduces a range of challenges, particularly in the realm of multicultural communication. Effective communication is crucial for the success of these distributed teams, yet it is often impeded by cultural differences, language barriers, and varying communication styles. These obstacles underscore the recent and growing importance of intercultural understanding as a foundational element in forming and maintaining effective distributed teams.

DOI: 10.24507/icicelb.16.06.619

To underline the two backgrounds of this research, this introduction is structured in three parts: Global Software Engineering education, UML education, and Project-Based Learning (PBL). The research questions are given at the end of this section.

1.1. **Teaching Global Software Engineering.** With software engineering tasks becoming increasingly complex, companies and software development teams often employ skilled personnel all around the globe [1]. The need for simulating situations where teams have to operate geographically distributed has long been known. Due to increased burden both institutionally and to instructors as well as the lack of teaching frameworks, implementations of Global Software Engineering (GSE) classes remain rare.

Bosnić et al. describe their experiences in managing distributed, project-based courses [2]. They introduced a model identifying three parts in GSE courses – 1) Institutional, where participating universities have different regulations, grading systems and work-load (credit points); 2) Teaching, where course instructors have different levels of experience and teaching loads; 3) Project, where imbalances in the numbers of students at each participating university, their previous knowledge and cultural differences.

The benefit of empirical studies that emphasize the pedagogical challenges and values are formulated by Carver et al. in cost and benefit analyses with strategies to fully utilize these studies [3].

Instructors of Global Software Engineering need to balance teaching technical, soft, and intercultural skills. Marinho et al. highlight the significance of intercultural skills in GSE, emphasizing the need to identify, communicate, and manage cultural differences within international software development teams [4]. They recommend establishing a cultural knowledge base about the team members' diverse backgrounds to foster awareness and understanding of these differences. Additionally, teams should proactively plan responses to mitigate potential cultural misunderstandings.

How knowledge and experience are managed in GSE education requires careful considerations. Learning goals and learning environment have to match for a successful outcome [5, 6].

Students have to understand key problems in distributed software system development. They have to handle tools for distributed collaboration, such as cloud platforms, video conferencing software, and agile tools. Technical knowledge must be experienced, such as UML diagrams, good programming practices, and modern practices of software engineering. On an intercultural level, communication with project members from different countries must be faced. All have to be operated in an ethical scheme to foster team communication, information exchange, and respect.

Colomo-Palacios et al. describe several technical skills essential for students of GSE education, including requirements engineering, engineering design (e.g., UML, and state diagrams), software construction, testing, maintenance, configuration management, quality management, tools and methods, and software engineering process models such as phase-based or agile development models [7]. Joseph et al. argue that, although technical skills are crucial for a career in software engineering, they are insufficient for success in an IT career on their own [8]. Therefore, even with comprehensive technical training, the needs of software engineering, particularly Global Software Engineering, are not fully met. A balanced approach integrating both technical and non-technical skills is essential.

The identification of non-technical skills varies depending on the cultures involved and the project environment. Several studies have explored the necessary skills for students in Global Software Engineering. Beecham et al. and Clear et al. emphasize the importance of managing geographical distance, fostering collaboration and teamwork in distributed settings, and coordinating distributed software development processes [6, 9]. As the working landscape becomes increasingly globalized, a shift from merely acquiring knowledge to

mastering skills may be required for sustainable development. Damasevicius et al. support this perspective from a long-term learning standpoint [10].

1.2. **UML education.** Unified Modeling Language (UML) serves as a standard visual language for specifying, visualizing, constructing, and documenting the artifacts of software systems [11, 12, 13]. The tools available for creating UML diagrams are continuously evolving and research is active in tracking usability and conformity [14]. The common framework can help bridge differences in cultural and educational background, especially in multicultural teams. While cultural difference may be reflected in UML usage, language barriers and other cultural discrepancies post a higher disruption in international, distributed software development projects.

One of the primary challenges in using UML within multicultural distributed teams is ensuring that all team members have a consistent and accurate understanding of the models. This requires not only a standardized approach to UML but also an awareness and accommodation of cultural differences in interpretation and communication. However, for the need for often asynchronous communication and meetings, UML can mitigate misunderstandings and improve team cohesion.

GSE education brings unique opportunities to leverage practical projects in a safe, academic environment for students to gather experience before entering the global workforce. Students are expected to have some degree of format UML education by this point, but have varying degree of experience using it in practical projects.

To address these challenges, it is essential to integrate cultural competency training into the workflow of distributed software engineering teams and into educational curricula. Authors of this paper regularly advise students about common difficulties in multicultural distributed teams and help reduce friction between team members by giving examples from previous GSE projects and how difficulties were alleviated. One key strategy is the shift from synchronous team meetings, where a natural hurdle is English as a common language, to asynchronous exchange using collaborative modeling using UML diagrams.

1.3. **Project-Based Learning.** Project-Based Learning (PBL) is essential in GSE education because it closely mirrors real-world industry scenarios, emphasizing customer-oriented development and enhancing collaboration and communication skills in distributed teams. PBL nurtures critical thinking and problem-solving abilities, fosters adaptability, and integrates technical, managerial, and intercultural skills, providing a comprehensive learning experience. It increases student engagement and motivation through tangible projects and allows for continuous feedback and improvement, preparing students effectively for the challenges of professional software engineering careers. This approach has been reported as highly effective in teaching Global Software Engineering [5, 6, 15, 16, 17, 18].

The success of Project-Based Learning in GSE education depends on well-implemented guidelines by instructors. It requires disciplined planning by instructors from all participating sides and a high degree of guidance and oversight throughout the project [19].

Students were typically assigned to work on a concrete project for a real-world customer. The four projects described in this paper include a customer loyalty system for an Irish pub, an e-voting system prototype, a university laboratory scheduling and management system, and an elevator scheduling and optimization system. While the frameworks for each project were provided by the instructors, students were responsible for identifying, defining, and analyzing the specifications and requirements.

The research questions addressed in this research are

RQ1 How do graduate students perceive usefulness, ease of use and their intent of using UML before and after a distributed software development project?

RQ2 What are best practices that can be derived from a three-university GSE course?

RQ3 Can UML questionnaires inform on GSE class improvements?

The rest of the paper is structured as follows. Section 2 covers the research method used to analyze the course data, including a course description and data collection. Results are discussed in Section 3. The paper concludes in Section 4.

2. **Research Method.** Graduate school students participating in a GSE class co-taught at three universities in Japan, Germany, and Indonesia were surveyed using ex-ante and ex-post questionnaires. Questions are asked about demographic aspects, belonging to underrepresented groups, English proficiency, and free-form text comments. Two sets of questions were also asked: 1) Cultural dimensions and factors impactful to GSE were modeled after Hofstede et al. [20] and Marutschke et al. [21]; 2) Perceived easy of use, usefulness, and intent to use UML were formulated by applying the Technology Acceptance Model (TAM) framework, which has also a long history in educational research [22, 23, 24]. Due to the low number of participants, findings were visualized by violin plots and qualitatively analyzed.

The following questions were asked (label as given in Figures 1 and 2). Students were asked to rate on a five-point Likert scale, the first from "No confidence (1)" to "High confidence (5)" and the rest from "Strongly Disagree (1)" to "Strongly Agree (5)". How would you rate your confidence in using UML diagrams to represent and communicate software analysis and design artifacts? (*UML Confidence*).

My interaction (modeling and software development) with UML diagrams is clear and understandable. (*Clear UML interaction*) – Interacting (modeling and software development) with UML diagrams does not require a lot of my mental effort. (*UML mental effort*) – Creating UML diagrams (for modeling and software development) requires a lot of time and effort. (*Creation time and effort*) – I find it easy to get UML diagrams to do what I want it to do. (*UML control*) – Using UML diagrams improves communication and understanding between my team and professors/instructors. (*Improved communication*) – Using UML diagrams improves my productivity in software development. (*Improved productivity in SED*) – Using UML diagrams enhances collaboration in software development (amongst team members). (*Enhanced collaboration*) – I consider UML diagrams useful for software design. (*Useful for software design*) – I intend to use UML diagrams in future software design and development. (*Use intent*) – For collaborative software development, I predict that I would use UML diagrams in the future. (*Use intent for collaboration*).

2.1. **Course description.** This case study examines a master's level course in Global Software Engineering, taught during the fall semester of 2023/2024. Instructors and students from three universities Ritsumeikan University, Nuremberg Institute of Technology, and Brawijaya University in Indonesia, Japan, and Germany participated. The Japanese and German instructors had prior experience conducting distributed courses in Global Software Engineering [25], while the Indonesian university joined for the first time in the fall semester of 2023/2024. This and previous courses were in principle taught in the same educational framework. The addition of a third university was intended to increase the cultural diversity of the participants, and the Indonesian partner facilitated the inclusion of a real-world customer from a rapidly growing economy.

The course aimed to simulate a multi-national software engineering project using a Project-Based Learning (PBL) format, which has been shown to enhance student engagement in software engineering courses [26] (see also Section 1.3). Students were assigned to cross-site teams composed of members from Indonesia, Japan, and Germany. Each team worked with the local city government in (name of city omitted for peer review), Indonesia, which acted as the customer. The teams were tasked with performing requirements engineering and developing a prototype for smart tourism applications.

Institutional differences were evident before the course began. Each university had a different semester schedule: classes in Indonesia started mid-August, in Japan at the end of September, and in Germany at the beginning of October. This resulted in only three months of overlap – October, November, and December – when all three universities were in session. Additionally, grading requirements varied: Indonesian students had to demonstrate UML proficiency for requirements engineering, Japanese students had to give midterm and final presentations, and German students were required to present a final project and submit a written report.

2.2. **Data collection.** Anonymous online questionnaires were distributed to students at the beginning and the end of the semester. As each university followed a different schedule – the university in Indonesia started first in August, Japan followed in September, and Germany in October – the surveys were conducted accordingly.

A total of 29 students started taking the course with 22 finishing their projects. Their demographic details are given in Table 1.

Table 1. Demographic statistics of the dataset

	In	Ja	Ge	$\overline{\text{Age}}$
Beginning	5	13	11	24.0 ($\sigma = 2.7$)
End	5	8	9	24.4 ($\sigma = 3.5$)

Online surveys have shown to be helpful in data collection, due to their relative speed and low cost [27]. Issuing questionnaires anonymously online may also increase the number of willing participants, who prefer filling forms out on a computer and where a face-to-face environment may pose a hurdle [28]. Due to differing power distances inherent between students and instructors from Japan (international students), Germany, and Indonesia [20], online surveys were implemented to allow students to answer questions anonymously.

Table 2. Nationalities and native languages of students as self reported by students, empty entries not counted, some indicating multiple native languages

Nationalities	Before	After	Language	Before	After
Chinese	6	0	Chinese	4	0
Indonesian	7	7	Indonesian	7	8
German	11	7	German	11	8
Other	5	5	Other	6	6

As can be seen from Table 2, main nationalities and spoken languages were German and Indonesian. Some students did not fill out this part, as it was left optional, others indicated multiple native languages. Most self reported intermediate to high English proficiency, as listed by Table 3. Students were instructed to indicate their level based on TOEFL iBT scores or estimate their level, if they do not know their score.

Questions to ask students about their UML usage and understanding was modeled after the Technology Acceptance Model, including perceived ease of use, perceived usefulness, attitude, and intent [22, 24] to facilitate comparison with other studies and possible future analyses. Other questions asked about the use of specific UML models and struggles students faced using UML diagrams communicating amongst team members. The following Section 3 details the results gathered from the ex-ante and ex-post questionnaires.

TABLE 3. English proficiency as self reported by students

Proficiency level (TOEFL iBT score)	Number of students
Highly Advanced or Native (100-120)	3
Advanced (80-99)	13
Intermediate (61-79)	11
Basic (45-60)	2
Low (20-44)	0
None (0-19)	0

3. **Results and Discussion.** This section describes the findings from comparing the TAM-modeled questions from ex-ante to ex-post. A visual analysis of violin plots, mapped as probability density functions, is presented. The following three subsections 3.1 to 3.3 discuss viewpoints and observations in each university with suggested best practices and improvement strategies for upcoming classes.

Out of the three universities, students from Brawijaya University had the most exposure to UML and (software) engineering education. Both Ritsumeikan University and Nuremberg Institute of Technology are computer science colleges. This is reflected in Figures 1 and 2, where students from Indonesia were consistently rating the effort and usefulness of UML highly (with the exception of the mental effort necessary to create UML diagrams).

After completion of the software engineering projects, a general tendency towards higher appreciation for UML and increased intent to use in professional and collaborative projects could be observed. The most uniform change can be seen for students in Japan, whereas

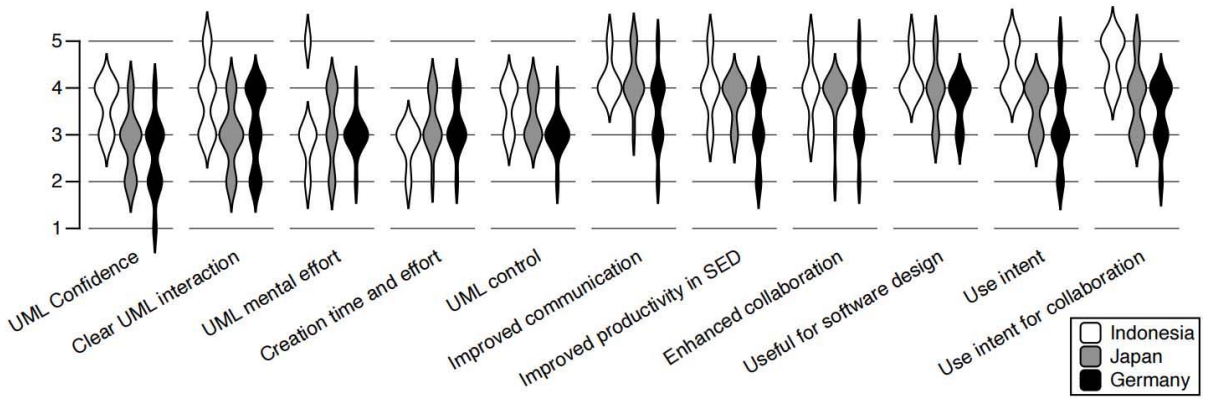

FIGURE 1. Perception of UML use as self-reported by students before the semester (ex-ante)

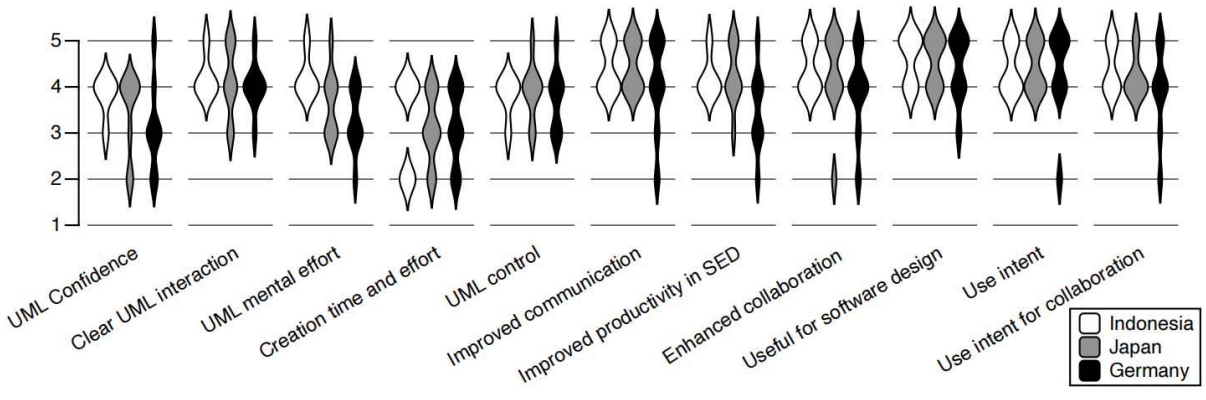

FIGURE 2. Perception of UML use as self-reported by students after the semester (ex-post)

the plots for students in Germany mostly retain a thin tail towards disagreeing with UML's usefulness and the intent to use it in the future.

Challenges most selected from multiple choice by students (in descending order) were creating a representative use case scenario, translating problem and solution domains into a UML diagram, collaboration using UML diagrams, choosing the appropriate diagram for a particular scenario, and understanding UML syntax and semantics.

3.1. **Course observations in Japan.** Japanese students encountered typical challenges throughout the semester, which they managed to overcome. Drawing from prior experience in GSE classes, common issues included varying communication styles within teams, reliance on software tools, and differing academic goals from team members in Germany and Indonesia.

Given the distinct organizational cultures and structures of the three universities – each with specific syllabus requirements and instructor expectations – perceptions of achievements varied. Students, influenced by their own academic goals, often experienced friction within distributed teams. To navigate these differences, students were advised to collaborate closely and received regular (often daily) feedback from instructors. Team members ultimately matured from the experience of presenting their team's working prototype successfully.

Despite these challenges, all teams exceeded instructors' expectations with their semester results. Particularly noteworthy were the practical aspects, where students developed diverse smart tourism applications, such as tourist attraction recommendations, dietary-specific restaurant recommendations, dynamic pricing for attractions, and specialized tour guides with Q&A chatbots. The instructors recognized the significant effort and coordination required to create these working prototypes. While the teams ultimately succeeded, there was room for improvement in team management and organization.

3.2. **Course observations in Germany.** German students were already familiar with the Project-Based Learning format of this course, thanks to prior experience in other courses. They greatly valued the opportunity to work on an international, real-world project in collaboration with the Malang City Government in Indonesia.

German culture promotes a relatively low power distance between students and instructors [20], making German students proactive in questioning their instructors. This behavior may have seemed disrespectful to students from Japan or Indonesia. As is typical in a low-context culture [29], German students communicated directly and extensively, often criticizing team members in ways that might be perceived as impolite by their Japanese and Indonesian counterparts.

Initially, German students focused prematurely on technological aspects. Language barriers with cross-site team members led to misunderstandings of the problem domain, resulting in insufficient attention to the requirements specifications generated by students in Malang. German students learned the hard way that focusing solely on technical solutions without addressing the actual problem wasted valuable time and caused frustration.

As the semester progressed, German students adapted by relying more on written communication. This allowed team members in each country to adequately formulate their questions and responses in English, the common third language. German students also learned the importance of listening to subtle cues rather than interrupting and speaking over their project partners.

By the end of the semester, German students reflected on their lessons learned. They realized that in multinational projects, technical skills alone were insufficient. Intercultural communication skills were crucial for cross-site team coordination. Understanding project requirements before diving into technical solutions proved essential.

3.3. **Course observations in Indonesia.** Collaboration among Indonesian group members was exceptionally motivating and exceeded expectations, despite significant challenges in communicating in English. The integration of Japanese and German teams a few weeks into the already ongoing semester at Brawijaya University created some friction due to differing ideas about project requirements. Students required guidance and occasional intervention from instructors to understand the problem domain. As Indonesian students needed to discuss with the Malang City Government to elicit requirements, instructor assistance during these meetings was crucial.

The early discussions with government officials and Indonesian students, combined with the intercultural perspectives of Japanese and German team members, led to several innovative solutions to existing problems. Students needed frequent reminders to focus on solving actual problems rather than using modern technology for its own sake. The resulting prototypes demonstrated the effectiveness of the PBL and authentic teaching approach. One team, which showed the highest practical implications, was selected to present their solution to the Malang City Government.

4. **Conclusions.** Overall, the three-university GSE course positively impacted the academic community, as reflected in student performances. From a focus on UML use, several conclusions can be drawn.

First, while most students had formal training or education in UML, the survey results show a disconnect between the perceived usefulness and intent to use UML. In agreement with previous research, UML education for engineers has potential for improvement.

Second, there is potential to tap into the impact of collaborative projects using UML diagrams to share ideas and concepts. Encouraging students to collaborate on the basis of UML diagrams also increases the value of documentation created during the projects, which next generation students can benefit from.

Third, the challenges that students mentioned the most using UML diagrams can inform class design and help instructors prepare supporting materials and targeted feedback.

Due to the complexity of conducting such a GSE course, instructors plan to streamline educational materials and guide students more effectively in the future. Long-term projects, rather than one-semester projects, will be considered for future classes.

REFERENCES

[1] M. Humayun, M. Niazi, M. Assiri and M. Haoues, Secure global software development: A practitioners' perspective, *Applied Sciences*, vol.13, no.4, 2465, 2023.

[2] I. Bosnić, F. Ciccozzi, I. Crnkovic, I. Cavrak, E. N. R. Mirandola and M. Zagar, Managing diversity in distributed software development education – A longitudinal case study, *ACM Transactions on Computing Education*, vol.19, no.2, pp.1-23, 2019.

[3] J. Carver, L. Jaccheri, S. Morasca and F. Shull, Issues in using students in empirical studies in software engineering education, *Proc. of the 5th International Workshop on Enterprise Networking and Computing in Healthcare Industry (IEEE Cat. No. 03EX717)*, pp.239-249, 2003.

[4] M. Marinho, A. Luna and S. Beecham, Global software development: Practices for cultural differences, in *Product-Focused Software Process Improvement. PROFES 2018. Lecture Notes in Computer Science*, M. Kuhrmann et al. (eds.), Cham, Springer, 2018.

[5] S. Schneider, R. Torkar and T. Gorschek. Solutions in global software engineering: A systematic literature review, *International Journal of Information Management*, vol.33, no.1, pp.119-132, 2013.

[6] S. Beecham, T. Clear, J. Barr, M. Daniels, M. Oudshoorn and J. Noll, Preparing tomorrow's software engineers for work in a global environment, *IEEE Software*, vol.34, no.1, pp.9-12, 2017.

[7] R. Colomo-Palacios, E. Tovar, A. G. Crespo and J. G. Berbis, Identifying technical competences of it professionals: The case of software engineers, *International Journal of Human Capital and Information Technology Professionals*, vol.1, no.1, pp.31-43, 2010.

[8] D. Joseph, S. Ang, R. H. L. Chang and S. A. Slaughter, Practical intelligence in it: Assessing soft skills of it professionals, *Commun. ACM*, vol.53, no.2, pp.149-154, 2010.

[9] T. Clear, S. Beecham, M. Daniels, J. Barr, R. McDermott, M. Oudshoorn, A. Savickaite and J. Noll, Challenges and recommendations for the design and conduct of global software engineering courses: A systematic review, *ACM Proceedings on ITiCSE*, pp.1-39, 2015.

[10] R. Damasevicius, R. Maskeliuna and T. Blazauskas, Faster pedagogical framework for steam education based on educational robotics, *International Journal of Engineering and Technology*, vol.7, pp.138-142, 2018.

[11] R. Reuter, T. Stark, Y. Sedelmaier, D. Landes, J. Mottok and C. Wolff, Insights in students' problems during UML modeling, *2020 IEEE Global Engineering Education Conference (EDUCON)*, pp.592-600, 2020.

[12] I. Hadar and O. Hazzan, On the contribution of UML diagrams to software system comprehension, *J. Object Technol.*, vol.3, no.1, pp.143-156, 2004.

[13] D. Budgen, A. J. Burn, O. P. Brereton, B. A. Kitchenham and R. Pretorius, Empirical evidence about the UML: A systematic literature review, *Software Practice and Experience*, vol.41, no.4, pp.363-392, 2011.

[14] Y. Lu and C. A. Alexandru, Systematic review of UML diagramming software tools for higher education software engineering courses, *The United Kingdom and Ireland Computing Education Research (UKICER) Conference*, Swansea Wales, UK, pp.1-7, 2023.

[15] D. Jiang and J. Lin, Project-based learning with step-up method – Take CDIO abilities cultivation in computer specialty for example, *Proc. of the 8th International CDIO Conference*, Queensland University of Technology, 2012.

[16] A. Rodrigues and S. Santos, A framework for applying problem-based learning to computing education, *Proc. of Frontiers in Education Conference (FIE 2016)*, 2016.

[17] J. M. Olivares-Ceja, B. Gutierrez, A. Kress, J. Staufer and P. Brockmann, Project-based learning in an international classroom to teach global software engineering, *Proc. of International Conference on Education and New Learning Technologies (EDULEARN17)*, 2017.

[18] A. K. Peters, W. Hussain, A. Cajander, T. Clear and M. Daniels, Preparing the global software engineer, *Proc. of the 10th International Conference on Global Software Engineering (ICGSE 2015)*, pp.61-70, 2015.

[19] M. Hidalgo, H. Astudillo and L. M. Castro, Challenges to applying role playing in software engineering education: A taxonomy derived from a rapid literature review, *SN Computer Science*, vol.5, no.6, 694, 2024.

[20] G. Hofstede, G. J. Hofstede and M. Minkov, *Cultures and Organizations: Software of the Mind*, 3rd Edition, McGraw Hill, 2010.

[21] D. M. Marutschke, V. V. Kryssanov and P. Brockmann, Balanced, unbalanced, and one-sided distributed teams – An empirical view on global software engineering education, *IEICE Trans. Inf. & Syst.*, vol.E105, no.1, pp.1-9, 2022.

[22] Ö. F. Ursavaş, *Conducting Technology Acceptance Research in Education: Theory, Models, Implementation, and Analysis*, Springer Texts in Education, Springer International Publishing, 2022.

[23] P. Lai, The literature review of technology adoption models and theories for the novelty technology, *Journal of Information Systems and Technology Management*, vol.14, no.1, pp.21-38, 2017.

[24] D. Marikyan, S. Papagiannidis and G. Stewart, Technology acceptance research: Meta-analysis, *Journal of Information Science*, 2023.

[25] D. M. Marutschke, P. Brockmann and V. Kryssanov, Agile, hybrid teaching methods of distributed project management and intercultural skills for global software enginerring during the pandemic, *Proc. of International Conference of Education, Research and Innovation*, 2022.

[26] P. Morais, M. J. Ferreira and B. Veloso, Improving student engagement with project-based learning: A case study in software engineering, *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol.16, no.1, pp.21-28, 2021.

[27] M. S. D. P. Nayak and K. A. Narayan, Strengths and weaknesses of online surveys, *Technology*, vol.6, no.7, DOI: 10.9790/0837-2405053138, 2019.

[28] M. V. Selm and N. W. Jankowski, Conducting online surveys, *Quality and Quantity*, vol.40, pp.435-456, 2006.

[29] E. T. Hall and M. R. Hall, *Hidden Differences: Doing Business with the Japanese*, Anchor Press/ Doubleday, 1987.