Software Design and Modeling Practices in an Online Software Engineering Course: The Learners' Perspective

Mahum Adil[a], Ilenia Fronza[b] and Claus Pahl[c]
Free University of Bozen/Bolzano, Italy

Keywords: Global Software Engineering (GSE), GSE Education, Scrum, Software Modeling, Software Design.

Abstract: *Background.* Global Software Engineering (GSE) education is an established practice in academia. Several methods and tools support communication and programming activities, but earlier development stages, such as software design and modeling practices, are less explored.
Aim. The goal of this work is to analyze the learners' perspective during an online Software Engineering course. In particular, we focus on planning/organization activities and socio-technical challenges during the software design and modeling process.
Method. We used a mixed-method approach to collect data from 30 undergraduate students enrolled in an online Software Engineering course. We combined questionnaires and interviews to analyze four GSE elements (i.e., communication practices, team collaboration, task allocation and distribution, and usage of collaboration tools). Moreover, we analyzed the socio-technical challenges faced by the teams.
Results. Brainstorming is the most common practice used for planning software design and modeling activities. According to students, the usage of variant design notation is among the technical challenges. Despite the challenges, students would prefer to continue working in distributed teams.
Conclusions. The result shares the lessons learned that can be helpful to build best practices for managing software design and modeling activities in GSE project-based courses. It includes the need to define standard architectural terminologies, standard list of collaboration tools, early identification of architectural artifact dependencies, frequent design reviews, and face-to-face kick-off meetings.

1 INTRODUCTION

In the past two decades, software development has evolved from small co-located development teams to large geographically distributed teams (Šmite et al., 2010). In today's global economy, many software companies operate in a distributed environment to counter the concerns related to the cost of project development, acquire high-skilled resources, and increase global production to satisfy the foreign market (Ebert et al., 2016). This paradigm shift in the software development process is widespread after the COVID-19 pandemic (Sako, 2021), and Global Software Engineering (GSE) is a common practice to organize software engineering activities in geographically distributed development teams (Shafiq et al., 2020). In academia, many researchers promoted GSE education through practical projects to provide the knowledge and real-time experience of working in distributed environments (Fortaleza et al., 2012; Fronza et al., 2022). Consequently, GSE education provides software engineering students with knowledge, skills, and understanding of working in collaborative settings (Bass et al., 2015) and is continuously evolving to prepare future software engineers to work in distributed environments. This widespread adoption in academia and the software industry led to several research works to support distributed software development (Dikert et al., 2016). Most courses presented in the literature focus on the code-centric development activities (Bosnić and Čavrak, 2019) and address various communication (language and culture difference) and coordination (time-zone difference) challenges in the GSE environment (Saleem et al., 2019). However, it is unclear within academic and software industry how collaboration technologies are used for software design and modeling practices (Capilla et al., 2016).

The goal of this work is to analyze the learners' perspective during an online Software Engi-

[a] https://orcid.org/0000-0001-6452-6085
[b] https://orcid.org/0000-0003-0224-2452
[c] https://orcid.org/0000-0002-9049-212X

Adil, M., Fronza, I. and Pahl, C.
Software Design and Modeling Practices in an Online Software Engineering Course: The Learners' Perspective.
DOI: 10.5220/0010978000003182
In *Proceedings of the 14th International Conference on Computer Supported Education (CSEDU 2022) - Volume 2*, pages 667-674
ISBN: 978-989-758-562-3; ISSN: 2184-5026

neering course. In particular, we focus on planning/organization activities and socio-technical challenges during the software design and modeling process. We used a mixed-method approach, combining interviews and questionnaires to collect data on four GSE elements (i.e., communication practices, team collaboration, task allocation and distribution, and usage of collaboration tools). The results show that students appreciated working in a distributed environment, whereas confirmed common socio-technical challenges while working in a distributed environment. We share the lessons learned from this experience to enhance the planning and organization of software design and modeling process in GSE education.

The paper is structured as follows. In Section 2, we provide background information on distributed software development and software design in GSE settings. In Section 3, we define our research objective, and in Section 4, we describe the research methodology. In Section 5, we discuss in detail the research finding, then in Section 6, we share the lessons learned from this study. In Section 7, we discuss the limitations. Section 8 concludes the paper and suggests possible directions for future work.

2 RELATED WORK

Researchers and practitioners have been studying GSE well over the decade. Beecham et al. (Beecham et al., 2013) identified various socio-cultural challenges related to communication, coordination, and management in a geographically distributed environment. These challenges focus most on the human aspect of distributed software development, such as diversity and inclusion, communication, team building, and social relation (Hoda et al., 2017). Various studies discussed the importance of GSE to manage the software development process, remote communication, and associated effects on collaboration between distributed teams (Colomo-Palacios et al., 2014). Various agile frameworks are used in the education and industry sectors to manage the software development process in a distributed environment. Distributed agile development has emerged in academia to manage the collaboration of geographically distant teams working on a project (Jalali and Wohlin, 2012).

There are three main approaches for GSE education in academia, starting from a project-based collaboration between multiple universities, to open source projects with industrial clients or serious-game based simulation (Vizcaíno et al., 2019) to give students hands-on practice of GSE scenarios (Beecham et al., 2017). Geographically distributed project-oriented Software Engineering (SE) courses provide opportunity for undergraduate students to work on a project in teams and learn resilience to tackle socio-technical challenges. Cavrak et al. (Čavrak et al., 2019) studied team resilience in agile teams, by analyzing the product and process quality of student teams working in distributed environments. A set of practices were suggested to mitigate stress within a team, including cautious team organization to balance contributing and non-contributing members, continuous awareness, and the introduction of possible challenges (e.g., changing requirements). Kropp et al. (Kropp et al., 2016) discussed agile collaboration and coordination practices in software engineering courses. The authors emphasized the importance of collaboration tools to enhance SE education in a collaborative environment. Teaching software engineering courses in a distributed environment comes with several challenges (Hoda et al., 2017). To address these challenges, many researchers and practitioners proposed strategies and recommendations on how to counter diversity and inclusion (Olayinka and Stannett, 2020), communication (Vallon et al., 2018), team building (Iftikhar et al., 2017), and social relation (Clear and Beecham, 2019). Thus, research in the GSE education field focused on team coordination and management; however, software design and modeling activities in a distributed environment have not been discussed in depth.

Software Design in GSE Education. Software design is a continuous discovery process and requires a software architect to discover requirements, context, design decisions, and project element concerns (Capilla et al., 2016). This information helps to design model formation by progressing through a small design coalition to a consensus team model (Jolak et al., 2020). In GSE settings, software design and modeling require a careful selection of practices to support knowledge sharing and architectural plans across all teams. There are limited proposed collaboration tools to manage software architecture activities in a GSE setting. Portillo et al. (Portillo-Rodríguez et al., 2012) conducted a systematic mapping study to get an overview of the available collaboration tools and their features in various phases of GSE-based projects: most of the extracted tools for software modeling were solution proposals that support Awareness (i.e., visual and session awareness) design feature. Jain and Suman (Jain and Suman, 2015) presented a systematic literature review to discuss the technical and non-technical challenges for GSE-based projects. The authors shared the best practices to address the challenges; moreover, they highlighted the need for collaboration tools for software modeling and design, to support conflict resolution and concurrency in distributed teams. Capilla et al. (Capilla et al., 2016) presented a 10 year review of research on software architecture knowledge management. The results found no systematic architectural knowledge management approach used with the available tools to capture stable design decisions. Another systematic literature review (Sievi-Korte et al., 2019) found limited research available on how software development and management is done during the design phase in distributed environments.

3 RESEARCH OBJECTIVE

The goal of this work is to analyze the learners' perspective during an online Software Engineering course. In particular, we explore four GSE elements, i.e., communication practices, team collaboration, task allocation and distribution, and usage of collaboration tools for software design and modeling. The scope of the research is divided into the following research questions:

RQ1 - Planning and Organisation. How are software design and modeling activities planned and organised in an online course by distributed teams?

RQ2 - Operational Support. What are the technical challenges faced in modeling, team coordination, and communication support that affect the design process in distributed teams?

RQ3 - Project Management. How are the challenges addressed in an online course to improve the distributed team management for software design and modeling process?

4 RESEARCH METHOD

Course Design. The context of this study is a online Software Engineering course offered in a fourth semester bachelor degree in Computer Science and Engineering at the Free University of Bozen/Bolzano, Italy. Students were grouped in 9 distributed teams of 3-4 members. Each team was asked to provide a software design specification report for two projects, i.e., two different case studies: web-based applications for car rental systems and university project management systems. A set of guidelines was given in each project regarding the different types of Unified Modeling Language (UML) diagrams. The first project lasted four weeks, while the second project was six weeks long. The estimated development effort was 10-12 person-months.

The course instructor acted as customer and software architect to provide guidance and administer the progress of the projects. A Scrum-based process was implemented to monitor progress weekly, and to foster discussion on issues relating to project backlog, in-progress, completed, and arising problems. The instructor organized weekly sprint meetings in 9 separate video-conference rooms for project discussion. The final evaluation of the course projects was based on the expert assessment by the course instructor. Table 1 shows adopted Scrum and GSE practices in the online course to investigate planning and organization, operational support, and project management of software design and modeling process in a distributed environment.

Table 1: Scrum and GSE practices in the online course.

Scrum Practices	GSE Practices
Frequent synchronous communication (sprint planning meeting, weekly Scrum meeting)	Distributed software design
Synchronous/asynchronous communication practices, weekly meeting with software architect (course instructor)	List of tools for synchronous / asynchronous communication
Frequent integration of design diagrams, backlog management	List of tools for communication and software design
Continuous communication, weekly Scrum meeting, frequent delivery of design diagrams	Distributed software design

Participants. A total of 30 undergraduate students (27 male, 3 female) were enrolled in the course. All the students had a similar background: they were all second-year bachelor students who majored in Computer Science and Engineering and, by regulation of the University, all of them passed the courses of math and introduction to programming, mandatory to be admitted to the second-year study programme. Even though all students reside in the same country, there were certain language and cultural differences which were out of the defined scope in our study.

Data Collection. Figure 1 shows the mixed-method approach we used for data collection and analysis. Table 2 shows the mapping between questionnaires/interviews and the three RQs.

In our two-stage data collection process, we combined a quantitative and qualitative method at the

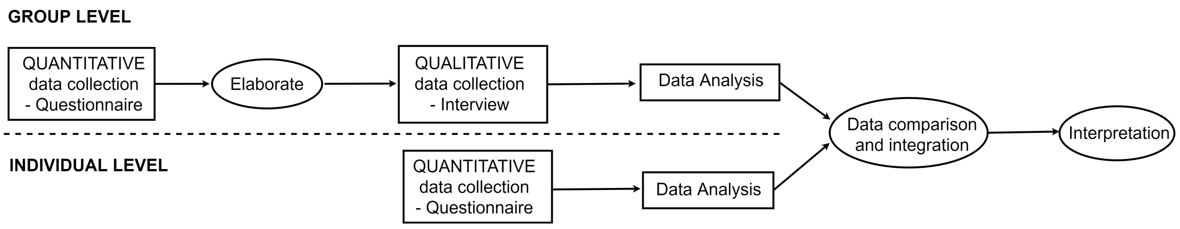
GROUP LEVEL
QUANTITATIVE data collection - Questionnaire
Elaborate
QUALITATIVE data collection - Interview
Data Analysis
Data comparison and integration
Interpretation
INDIVIDUAL LEVEL
QUANTITATIVE data collection - Questionnaire
Data Analysis

Figure 1: Mixed-method approach for data collection and analysis.

Table 2: Quantitative and qualitative data collection mapped to RQs.

RQ	**Quantitative Analysis**		**Qualitative Analysis**
	Group Questionnaire	Individual Questionnaire	Group Interview
RQ1		Did your team agree on a sequence of steps to take design decisions? If yes, define them and If no, why? [Text] How much time (in hours) your team spent to understand and discuss the project specification? [Text] How much time (in hours) your team spent on design diagrams? [Text]	What method your team used to do brainstorming for requirement analysis while designing diagrams?
RQ2	What communication tool(s) or other tools did you use for discussion and management in your team? [Text] What software design tool(s) did you use in your project? [Text]	Do you think that your team faced any informal communication challenges during project? [Likert Scale] Do you think that your team faced any technical challenges while designing diagrams? [Likert Scale]	
RQ3	What were the challenge(s) in your team while designing diagrams and managing the project work given in the list? (you can choose more than one): - availability of tools - handling of tools - technical constraints - team coordination - other? which? [please comment, if applicable]	How frequent did your team disagree on the design decisions? [Likert Scale] Were you always aware or were informed of changes in the design diagrams? (e.g., if changes were made by somebody else) [Likert Scale]	What were the key strategies used for addressing the observed and/or reported challenges/problems?

group level, which was then augmented with a quantitative analysis at the individual level to analyze the students' perspective of working in a distributed environment. At the group level, first, we shared a questionnaire with the 9 teams to collect information on the collaboration tools used and possible related challenges. Then, we conducted semi-structured interviews with the 9 teams to clarify the results collected from the initial questionnaire.

The duration of each interview was 10 minutes, in which we asked five main open-ended questions. At the individual level, we shared a questionnaire (using an online form) with the 30 students to collect their feedback on working in a distributed environment. We received 20 responses on team communication and coordination practices, possible socio-technical challenges, and their effect in the distributed environment. During data collection, we considered the following four main GSE elements: communication practices, team collaboration, task allocation and distribution, and usage of collaboration tools.

Data Analysis. We transcribed and analyzed all the interviews using thematic analysis (Braun and Clarke, 2012) for qualitative coding. We executed the six steps suggested by Braun and Clarke (Braun and Clarke, 2012), i.e., we transcribed the group interviews into a written document, extracted initial codes, finalized codes and grouped them into themes, and provided analysis concerning themes and research questions. Three main categories of themes emerged from the analysis: 1) *technology dependence* - screen sharing, 2) *scaling Scrum practices* - dividing into sub-teams, 3) *importance of expert opinion* - weekly feedback from expert.

5 RESULTS

In this section, we describe our results against the research questions of the study.

RQ1: How Are Software Design and Modeling Activities Planned and Organized in an Online Course by Distributed Teams? In group interviews, five teams mentioned technology dependence, i.e., they used the screen sharing feature to plan and design diagrams collectively. Four teams used scaling Scrum practice, i.e., they divided tasks among sub-teams to enhance coordination and communication. Through the individual questionnaire, we identified the practices used by the teams to plan and support the software design process (Table 3): brainstorming is the most common practice reported by students: *At the start of each project our team first did a brainstorming session to elicit functional and non-functional requirements, and then after defining all the requirements, we started designing the UML diagrams.*

Table 3: Common design modeling practices.

Design Modeling Practices	%
Initial brainstorming	65%
Task allocation	25%
Division of team into sub-team	10%

The results show that 55% of the students spent 4-5 hours per week for team collaboration and understanding project specifications so that the design diagrams remain consistent throughout the project. Moreover, 60% of students spent more than 10 hours per week defining the UML diagrams.

RQ2: What Are the Technical Challenges Faced in Modeling, Team Coordination and Communication Support That Affect the Design Process in Distributed Teams? As shown in Table 4, 68% of students reported communication challenges while working in the distributed team: *Scheduling meetings in time slots available for everyone was challenging. Also, to work without tools like pen and paper to sketch ideas resulted in a communication gap.*

To collaborate within teams, students used various collaboration tools (Table 5) for project management

Table 4: Team communication and technical challenges.

Communication challenges	%
Unable to share idea	30%
Arranging team meeting	30%
Audio distortion	8%
Technical challenges	**%**
Variant design notation	30%
Live editing not supported	20%
Internet connectivity	20%
Limited access as free user	15%

and development. According to the students, the efficient user interface was the main reason for choosing the tools. However, while using collaboration tools 85% of the students faced technical challenges (Table 4) that were mainly related to limited support of commercial collaboration tool as a free user or internet connectivity issue causing voice distortion. *The technical challenges we faced were mainly related to the automated tool used for design diagrams. Although the tool we used was efficient but it lacked some flexibility regarding the graphical adjustments.*

Table 5: Common team collaboration tools.

Goal	Tool names
Communication	WhatsApp, Telegram, Microsoft Teams
Software Design	LucidChart, Draw.io

RQ3: How Are the Challenges Addressed in an Online Course to Improve the Distributed Team Management for Software Design and Modeling Process? During the group interviews, five teams shared the importance of expert opinion to address any issue/challenge in the design process. Based on collective results, 65% of students faced disagreement within the team while developing UML diagrams, whereas 35% students rarely disagreed on the design decisions. The results show that the most common challenges were the different understanding of the course guidelines for building UML diagrams or different interpretations of the requirements, which led to communication challenges within the teams: *We had disagreements because of different interpretations of the case studies. Sometimes there were ambiguities in the statement of the problem which could lead us to have different interpretations which therefore lead us to disagreement in the design phase.*

6 DISCUSSION

Based on the results, we identified design and modeling activities as relevant factors for the success of the distributed teams. Students prefer working in the distributed environment; however, they suggested a balanced approach in which initial face-to-face meetings serve to gain a better understanding of the project and explore the expertise of each member for efficient and well-structured task allocation and distribution.

We present the following Lessons Learned (LL) to build best practices for managing software design and modeling activities for online course projects.

LL1. Define Standard Terminologies for the Project. During the interview, some teams observed that sometimes they used variant terms for the same subject in design diagrams, which caused inconsistency in the software architecture diagrams and communication gaps within teams. The lack of consensus on the standard terms to be used in the project is indeed a well-known issue in GSE (Peixoto et al., 2018). Therefore, educators and researchers need to use standard terminologies to manifest a clear understanding for students to articulate terms related to software modeling and design artifacts.

LL2. Define the Standard List of Collaboration Tools to Use. The instructor provided a list of possible collaboration tools for project management, team communication, and design diagram. Thus, each group chose a different set of collaboration tools. However, many studies reported that standard tools provide better support and interaction in a distributed environment (Sarma and Hoek, 2002; Lanubile, 2003). Based on the collected interview data, Table 6 suggests a list of collaboration tools for software design and modeling in a distributed environment.

LL3. Early Identification of Architectural Artifact Dependencies. During our interviews, some teams mentioned that they worked in sub-teams; this affected their entire architectural plan because the modules were tightly coupled, and communication with other sub-teams was needed. While working in a distributed environment, it is crucial to plan subsystem and architectural dependencies to define clear responsibilities for the team: Bosch and Bosch-Sijtsema (Bosch and Bosch Sijtsema, 2010) discussed an architecture-centric framework to reduce architectural artifact dependencies within teams. This shows the importance of letting students identify dependencies early in the project to increase cohesion and efficiency in software design and modeling.

LL4. Frequent Design Reviews. In a distributed environment, it is crucial to keep the team synchronized and aware of all the design change decisions (Cusumano, 2008). In this study, students reported being aware of the changes done within the project whereas, there was variance in the diagrams in respect to the initial discussion. This shows the importance of communication and frequent design reviews within the team to facilitate coherent design diagrams.

LL5. Introduce Face-to-Face Kick-off Meetings. Based on the results, many students prefer working in a distributed environment; however, they reported challenges in eliciting project specification and prioritization as they could not transfer ideas in video-conference meetings. Students suggested having a kick-off meeting in the start of the project so that all members can meet, do brainstorming to explore project specifications, and allocate the initial tasks based on each member's expertise. This will be then followed up with weekly progress online meetings to discuss project backlog. In a recent study by Smite et al. (Smite et al., 2021) on working in a distributed environment, many practitioners also supported the idea to reinforce face-to-face team meetings to increase team cohesion, problem-solving, and knowledge sharing process. Moreover, it will be helpful to achieve lesson learned LL3 to capture emerging aspects at early stage of software design process.

7 LIMITATION AND VALIDITY

We acknowledge that the number of participants (i.e., 30 students) might represent a possible limitation to the validity of the results of this study. Furthermore, the individual questionnaire received 20 responses, which again might affect the results. Moreover, we used thematic analysis (Braun and Clarke, 2012) to evaluate qualitative results, which could suffer from issues related to data transparency. To limit this problem, while extracting results, we validated all the answers against each question to build consistency between the research questions and reported answers.

Using a mixed-method approach to collect data helped us to increase validity. To avoid any potential bias, the third author (i.e., the course instructor) was not involved in the interviews. Furthermore, we followed the validity analysis and threat guidelines provided by Wohlin et al. (Wohlin et al., 2012) to mitigate research bias. Possible threats to construct validity is related to how empirical evidence was used to report research findings. To mitigate threats to construct validity, we created an anonymous questionnaire to ensure data confidentiality and avoid evaluation apprehension. To increase external validity, the materials of the research study can be used by other researchers within the Software Engineering field involved in software modeling and design practices to have a broader perspective for distributed environment in academia.

Table 6: Suggested tools for software design and modeling in GSE projects.

Type	Name	Features	Drawbacks
Project management	Jira	Features for agile teams, such as Scrum board, Kanban board, roadmaps, agile reports, project tracking and management. Mobile application supported	Free access up to 10 users
Software design diagram	Draw.io	Web-based application to create flowcharts, network diagrams, UML diagrams, ER models, etc.	The web application lags if worked for long hours
Team communication	Microsoft Teams	Integration with the Office 360 suite, teams and channel creation, chat function, document storage, screen sharing, group video and audio call, support desktop/mobile application	No unified search bar for available products

8 CONCLUSION AND FUTURE WORK

GSE project-based courses teach development practices in a distributed environment. In this work, we analyzed planning and organization activities of software design and modeling process in an online Software Engineering course. We addressed the socio-technical challenges students faced in modeling software design diagrams within distributed teams.

We used a mixed-method approach to assess four GSE elements – communication practices, team collaboration, task allocation and distribution, and usage of collaboration tools. The results indicate that most teams used initial brainstorming as design modeling practice although most teams reported that they faced difficulty to share idea in distributed environment. Furthermore, most teams shared that the variant design notations in each design tools caused difficulty in project management and design development. However, there was no major effect reported by teams, in fact most teams supported the idea of working in distributed environment.

Instructors can benefit from the presented empirical evidence to explore emerging aspects of software design and modeling activities that could be of value in GSE project-based courses and apply our lessons learned to their contexts. In the future, we plan to study planning and organization practices used for software design and modeling in the industrial sector. This will provide a comparative analysis of both sectors to propose solutions to enhance software design practices for the distributed environment.

REFERENCES

Čavrak, I., Bosnić, I., Ciccozzi, F., and Mirandola, R. (2019). Resilience of distributed student teams to stress factors: A longitudinal case-study. *Information and Software Technology*, 114:258–274.

Bass, J. M., McDermott, R., and Lalchandani, J. (2015). Virtual teams and employability in global software engineering education. In *2015 IEEE 10th International Conference on Global Software Engineering*, pages 115–124.

Beecham, S., Clear, T., Damian, D., Barr, J., Noll, J., and Scacchi, W. (2017). How best to teach global software engineering? educators are divided. *IEEE Softw.*, 34(1):16–19.

Beecham, S., OLeary, P., Richardson, I., Baker, S., and Noll, J. (2013). Who are we doing global software engineering research for? In *2013 IEEE 8th International Conference on Global Software Engineering*, pages 41–50.

Bosch, J. and Bosch Sijtsema, P. (2010). Coordination between global agile teams: From process to architecture. In *Agility Across Time and Space*, pages 217–233. Springer.

Bosnić, I. and Čavrak, I. (2019). Project work division in agile distributed student teams - who develops what? In *2019 ACM/IEEE 14th Int. Conference on Global Software Engineering (ICGSE)*, pages 162–171.

Braun, V. and Clarke, V. (2012). Thematic analysis. *APA handbook of research methods in psychology. Research designs: Quantitative, qualitative, neuropsychological, and biological*, 2:57–71.

Capilla, R., Jansen, A., Tang, A., Avgeriou, P., and Babar, M. A. (2016). 10 years of software architecture knowledge management: Practice and future. *Journal of Systems and Software*, 116:191–205.

Clear, T. and Beecham, S. (2019). Global software engineering education practice continuum special issue of the acm transactions on computing education. *ACM Tr. Comput. Educ.*, 19(2).

Colomo-Palacios, R., Casado-Lumbreras, C., Soto-Acosta, P., García-Peñalvo, F. J., and Tovar, E. (2014). Project managers in global software development teams: a study of the effects on productivity and performance. *Software Quality Journal*, 22(1):3–19.

Cusumano, M. A. (2008). Managing software development in globally distributed teams. *Communications of the ACM*, 51(2):15–17.

Dikert, K., Paasivaara, M., and Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, 119:87–108.

Ebert, C., Kuhrmann, M., and Prikladnicki, R. (2016). Global software engineering: Evolution and trends. In *2016 IEEE 11th International Conference on Global Software Engineering (ICGSE)*, pages 144–153.

Fortaleza, L. L., Conte, T., Marczak, S., and Prikladnicki, R. (2012). Towards a gse international teaching network: Mapping global software engineering courses. In *Proceedings of the Second International Workshop on Collaborative Teaching of Globally Distributed Software Development*, CTGDSD '12, page 1–5.

Fronza, I., Corral, L., Wang, X., and Pahl, C. (2022). Keeping fun alive: an experience report on running online coding camps. In *Proceedings of the 44nd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET '22)*, New York, NY, USA. ACM.

Hoda, R., Babar, M. A., Shastri, Y., and Yaqoob, H. (2017). Socio-cultural challenges in global software engineering education. *IEEE Transactions on Education*, 60(3):173–182.

Iftikhar, A., Alam, M., Musa, S., and Su'ud, M. M. (2017). Trust development in virtual teams to implement global software development (gsd): A structured approach to overcome communication barriers. In *2017 IEEE 3rd International Conference on Engineering Technologies and Social Sciences (ICETSS)*, pages 1–5.

Jain, R. and Suman, U. (2015). A systematic literature review on global software development life cycle. *SIGSOFT Softw. Eng. Notes*, 40(2):1–14.

Jalali, S. and Wohlin, C. (2012). Global software engineering and agile practices: a systematic review. *Journal of software: Evolution and Process*, 24(6):643–659.

Jolak, R., Savary-Leblanc, M., Dalibor, M., Wortmann, A., Hebig, R., Vincur, J., Polasek, I., Le Pallec, X., Gérard, S., and Chaudron, M. R. (2020). Software engineering whispers: The effect of textual vs. graphical software design descriptions on software design communication. *Empirical Software Engineering*, 25(6):4427–4471.

Kropp, M., Meier, A., and Biddle, R. (2016). Teaching agile collaboration skills in the classroom. In *2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET)*, pages 118–127.

Lanubile, F. (2003). A p2p toolset for distributed requirements elicitation. In *Proc. of the International Workshop on Global Software Development (GSD 2003), Portland, Oregon, USA*.

Olayinka, O. and Stannett, M. (2020). Experiencing the sheffield team software project: A project-based learning approach to teaching agile. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pages 1299–1305.

Peixoto, A., González, C. S. G., Strachan, R., Plaza, P., de los Angeles Martinez, M., Blazquez, M., and Castro, M. (2018). Diversity and inclusion in engineering education: Looking through the gender question. In *2018 IEEE Global Engineering Education Conference (EDUCON)*, pages 2071–2075.

Portillo-Rodríguez, J., Vizcaíno, A., Piattini, M., and Beecham, S. (2012). Tools used in global software engineering: A systematic mapping review. *Information and Software Technology*, 54(7):663–685.

Sako, M. (2021). From remote work to working from anywhere. *Commun. ACM*, 64(4):20–22.

Saleem, N., Mathrani, S., and Taskin, N. (2019). Understanding the different levels of challenges in global software development. In *2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE)*, pages 76–77.

Sarma, A. and Hoek, A. (2002). Palantir: Increasing awareness in distributed software development. In *International Workshop in Global Software Development at ICSE*.

Shafiq, M., Zhang, Q., Akbar, M. A., Kamal, T., Mehmood, F., and Riaz, M. T. (2020). Towards successful global software development. In *Proceedings of the Evaluation and Assessment in Software Engineering*, page 445–450. Association for Computing Machinery.

Sievi-Korte, O., Beecham, S., and Richardson, I. (2019). Challenges and recommended practices for software architecting in global software development. *Information and Software Technology*, 106:234–253.

Smite, D., Moe, N. B., Klotins, E., and Gonzalez-Huerta, J. (2021). From forced working-from-home to working-from-anywhere: Two revolutions in telework.

Šmite, D., Wohlin, C., Gorschek, T., and Feldt, R. (2010). Empirical evidence in global software engineering: a systematic review. *Empirical software engineering*, 15(1):91–118.

Vallon, R., Spiesberger, P., Zoffi, M., Zrelski, C., Dräger, C., and Grechenig, T. (2018). Teaching global software engineering in a remote customer environment. In *2018 IEEE 10th International Conference on Engineering Education (ICEED)*, pages 63–68.

Vizcaíno, A., García, F., García, I., Guzmán, R. D., and Ángeles Moraga, M. (2019). Evaluating gsd-aware: A serious game for discovering global software development challenges. *ACM Tr. Comput. Educ.*, 19(2).

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media.