Teaching a Global Software Development Course: Student Experiences Using Onsite Exercise Simulation

Jouni Lappalainen | Nirnaya Tripathi | Jouni Similä

University of Oulu
Department of Information Processing Science
P.O. Box 3000, 90014 University of Oulu FINLAND
+358 294 481982

Jouni.Lappalainen@oulu.fi | Nirnaya.Tripathi@oulu.fi | Jouni.Simila@oulu.fi

ABSTRACT

Over the past decade, major advancements in software development have occurred in the global context. Global software development (GSD) is an effective strategy, and many higher educational institutions have been offering GSD courses. These courses are usually organized together with another institution situated in a different location. However, conducting such a course with more than one institution is not so economical since it involves greater collaboration among various institutions than in the case of a general onsite course. In this paper, we present an onsite simulation that deals with the specifics in the field of GSD training and teaching. We analyzed the students' learning reflections with a phenomenographic approach to validate the relevance of the design science construct of the course model containing an onsite simulation. Based on the analyzed data, it is possible to organize a GSD course on a single location with the aid of role-play simulation. The presented course model can help an institution prepare its students to solve most of the common problems faced in industrial GSD settings.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education – *computer science education*

General Terms

Design, Experimentation

Keywords

Course structure; Global software development; Phenomenography

1. INTRODUCTION

Many software companies often experiment with their remotely located software development centers to establish round-the-clock development and access quick information on local market conditions [11]. These practices have resulted in an increasing number of software engineers operating in global software development (GSD) centers around the world. The increase in the GSD trend means that future software practitioners will be operating in GSD centers at some point in their careers. Therefore, it is important for educators to provide future software practitioners with proficient knowledge of GSD [22]. However, the traditional software engineering (SE) courses and training offered by higher educational institutions do not fully cover the new challenges of GSD.

According to [21], teaching GSD involves preparing students for new theories, tools, and management skills. Many universities organize their GSD education with practical work segments and partner organizations (usually other educational institutions, either domestic or international). This strategy aims to prepare the students as realistically as possible for the challenges they will face while working in GSD industrial projects. Organizing this kind of collaboration usually involves various activities by the course instructors. These activities can include matching teaching and course schedules, ensuring that the students' previous studies correspond closely enough to each other, and organizing potential communication infrastructures [14]. Therefore, although the pedagogical benefits of a course with this kind of learning possibility can be shown [25], the resources and coordination needed from the organizing institution are notably more than those of a traditional course comprising lectures and written work only. Considering the large percentage of publications on GSD courses, relatively very few (if any) present experiences in a GSD course organized in an onsite environment [21].

The motivation behind our study is that despite the increasing number of GSD courses offered, teaching them in higher educational institutions is usually coordinated with at least two or more participating institutions. This is causing higher overhead costs in terms of the time spent in coordinating and planning these joint ventures than if the courses were taught onsite in one location, without compromising the relevance of the practical experience to the students' learning. To address such concerns, we propose the following research question (RQ):

"How should the practical work involved in delivering an onsite, higher-education GSD course be organized efficiently and economically, while providing students with a relevant view of the challenges presented by the global environment to software development projects?"

The rest of this paper is organized as follows. Section 2 reviews previous work and explains the theoretical background of this paper. Section 3 describes the research method used to conduct this study. Sections 4 and 5 present our artifact in the form of a course organization model and its evaluation. This paper

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

ICSE '16 Companion, May 14-22, 2016, Austin, TX, USA
© 2016 ACM. ISBN 978-1-4503-4205-6/16/05…$15.00
DOI: http://dx.doi.org/10.1145/2889160.2889198

concludes with a discussion of the results, conclusions, and possible directions for future work.

2. PREVIOUS RESEARCH

Economic globalization in the 21st century has brought significant changes to nearly all industries and particularly increased the distribution of software development. However, empirical studies showed that it was not an easy task to reap the benefits from cheaper, faster, and better development of software systems, products, and services [29]. A systematic literature review (SLR) of empirical evidence in global software engineering (GSE) [29] concluded that most of the 59 analyzed studies were based on intra-organizational industrial collaboration between two geographically distributed sites. The study also found out that the majority of empirical research represented problem-oriented reports but lacked in-depth analyses of solutions. However, seven best practices were identified and recommended for practitioners. Other SLR studies followed similar lines, but one study [27] found 330 papers containing 127 solutions, which were mapped to their respective GSE models. Finally, another study [22] presented a lighter and more focused SLR, identifying barriers and solutions to GSD and collaboration. We use the eight barriers found by [22] as a framework for analyzing our empirical findings and validating our course model to be of relevance for the industry.

In 2009, Monasor et al. [21] conducted an SLR in the field of GSD training and education. They found 38 primary studies, with the clear majority showing an increasing tendency from 2007 to 2009. Most of the research involved case studies (68%); 29 were conducted in a university context. Most of the studies were from the USA (26). The types of studies were classified as learning environments (10), e-learning approaches (3), simulators (2), teaching GSD in the classroom (7), GSD training in the classroom (13), and teaching GSD in the company (3).

In the context of education in GSE, several joint GSD courses with different universities had been organized so that students and future practitioners could benefit from a vast set of knowledge and experiences in GSD settings [16, 18]. The joint courses aimed to offer both theoretical classes and practical exercises. One master course was based on two universities collaborating through a class video recording system. Students at both universities had access to course source materials and video playback [16]. Another joint master's course intended to provide students from two countries with complementary knowledge about software design methods and project management practices [18].

A joint global software program was also organized with several universities from different countries. The program aimed to facilitate the practical and cultural scopes of GSD [19]. However, the general problem reported in this context was related to the coordination and collaboration among different universities [21]. Besides the collaboration among many universities in delivering global software education, it is also important to include the industry perspective during the educational process [6]. In this context, a global studio project was initiated in which universities joined together to offer their students the opportunity to learn GSD skills in industry settings. Some authors considered that the collaboration between academia and industry would be important for GSD learning. [6]

The pedagogical approach selected in this study for the GSD course, specifically for its practical work section, was simulation. This should not be confused with students running some specific simulation software on their computers but considered more like a role-playing exercise. There is some agreement among the researchers of this particular teaching approach that the terms *simulation*, *game*, and *role-playing* are interchangeably used in the literature, maybe somewhat mistakenly [2]. As far as the differences in the terminology exist, regarding this study, the goal of the method is to engage the students to use their knowledge in a safe, nonproduction environment and play a part in a given problem situation to gain new insights into what issues they might encounter in a real-world setting of the same scenario. This is closest to Errington's problem-based approach to role-play [5].

As a teaching method, simulation (as defined above) has achieved mixed success over the decades it has been actively used. Its strengths lie in its ability to demonstrate acquired knowledge and provide opportunities for learners to apply that knowledge in practice in a risk-free environment. Simulation has been used in a wide array of disciplines, such as drama, education, psychology, law enforcement, social sciences, literature, languages, environmental sciences, engineering, medicine, management, marketing, media, and information technology—to name a few [2, 32]. Coupled with the range of years that simulation has been used in one way or another for pedagogical purposes and the derivation of simulation from other established teaching methods such as sociodrama [7], its characteristics make simulation a viable tool to be considered for teaching GSD. Given the interpersonal nature of GSD and especially the communication-rich tasks at the early stages of a GSD project, a teaching method that highlights interactivity, understanding the activities of learners' counterparts and peers, the application of theoretical knowledge to practice, and the development of decision-making skills [2] is suitable for any course in the SE field. To this end, simulation has been reported as an appropriate method for various SE-related courses [1, 20, 31].

Given the aforementioned challenges of teaching GSD, such as engaging students and providing them with useful experiences, coordinating and organizing GSD courses between institutions, the relevance of GSD courses for the industry, and the reported experiences regarding them, there is clearly a gap in the existing literature about the topic. Despite many good reports about conducting different GSD courses, they usually present course models that involve more than one institution [23]. Thus, they are more complex and costly to organize. Other reported course experiences focus more on the lecture aspect of teaching, leaving students without the practical, hands-on type of exercises that would prepare them better to meet the challenges of GSD in industrial settings. This study aims to report on how to conduct the kind of GSD teaching that provides students with some grasp of the practical challenges of GSD and organizes this teaching in an economical way by delivering the entire course onsite. In Monasor et al.'s [21] categorization, this would fall into GSD teaching and training in the classroom.

3. RESEARCH METHODOLOGY

We chose design science research as the general approach to our study [12, 13]. Hevner et al. [13] formulated the conceptual research framework in terms of the environment, the research itself, and the knowledge base. The environment in our case consists mainly of the people, organizations, and technology needed for the design, implementation, and evaluation of the GSD course in question. These are described in detail in the following subsections and in Section 4. This research is divided into the develop/build and justify/evaluate phases. The artifact in our case is the GSD course, especially the simulation of the course exercises. The development of the course is partly described in Subsection 3.1 and more fully in Section 4. The course artifact and the exercises, as the main focus of this study, are evaluated in Section 5. Finally, the main knowledge base has already been described in the Introduction and in Section 2. The main methodological base is explained in Sections 3 and 5.

3.1 Case Description

We developed a master's level course on software development in the global environment (SDGE) for SE students. This course began in 2011 and has been taught each academic year since then. It aimed to study the challenges and solutions for developing software in the global setting. We took two approaches in the course. One approach involved giving face-to-face lectures on the theoretical foundations of GSD, while the other entailed conducting a simulation exercise in a real-world GSD project.

During the lectures and writing assignments, the topics mainly focused on issues such as the distribution of work items in GSD, communication issues due to multisite development, time zone dissimilarities, and multicultural concerns. The lectures also reviewed the current research aspects of GSD and related case studies from industry. The simulation exercise demonstrated distributed software development by a virtual team with the support of appropriate methods and tools. We expected that throughout the course, students should apply their knowledge gained during the lectures, such as dealing with cultural differences and collaboration techniques in dispersed teams in their simulation exercise. The last part of the simulation exercise required the students to write a reflection report.

3.2 Data Collection

During each iteration of the course, data was gathered from the students participating in the exercises. This data took the form of reflection reports. At the end, there were 199 student reflection reports to analyze, which were gathered from 2011 to 2013 (see Table 1). In these reports, the students analyzed their learning and its application possibilities. Writing such a report was (as mentioned) part of the student coursework and thus usually had information on other issues as well, such as course feedback and first-order descriptions of the tasks performed during the exercises. However, as far as the idea of reflection as a tool for learning and deeper understanding is concerned, this was not its main purpose [24]. Therefore, the reflections used in this study were analyzed and their contents were used only from their appropriate sections, where information about the learning could be gathered.

Using the material collected from students as a data source is commonly employed in many fields of research, including education research [28, 31] and computer science [15]. Moreover, as this study researched on student perceptions about their learning one reliable source of data is the students themselves. The reflective nature of the descriptions and analyses of learning take into account the problems of objectivity and reliability. These problems are mitigated at least to some extent by acknowledging reflective practice as a viable research approach in general [8, 9, 24, 30], as well as by the fact that this study's researchers were not the ones who produced the reflections and thus the research data. Therefore, the analysis and review of the objectivity and reliability of the data could be, and in this case, were applied after the reflection reports had been written.

The data was gathered by giving students an assignment that briefly described the required a reflective report. Some questions were also included, which were perceived as helpful in guiding the reflection when answered thoughtfully. Some of the questions in the reflection assignment were as follows: "What were the most fundamental learning outcomes and my thoughts after the exercise?" "What helped and advanced my skills and development in the assignments? What didn't aid in any way but in fact might have hindered my progress?" "How do I regard myself as a professional in the software development community now? Where do I see changes in my professionalism (in relation to my views before this course)?"

3.3 Data Analysis

For the data analyses, we took a similar approach as that used in [17]. The first and second authors conducted the analyses, while the third author monitored the whole process. The data analysis was arranged into different phases or steps, as outlined in [28]. At the first phase, the researchers familiarized themselves with the collected data by closely reading the reflection reports and gaining a high-level overview of the collected data. During the second phase, the researchers created the codes. Compiling common themes in the discussion topics in the reports generated these codes. During this phase, only the key points in the reports that summarized the topic discussed were considered; thus, the material was condensed. This phase maps to steps 3 and 4 in [28]. In the third phase, the codes were refined and redefined by continuously reading the reports. At this phase, the final set of themes was created from the codes, mapping to steps 5, 6, and 7 of [28]. In the end, we produced Table 2, which shows the structure of the categories and codings, while the textual descriptions of these categories are presented in Subsection 5.2. In Table 3, these categories are mapped against the theoretical framework, which is used to validate the course design.

The significance of the elements in the reports was assessed mainly by two of the three factors stated in [28]: the frequency of the element and the content of it. We found that due to the nature of the data collection, a lesser impact should be attributed to the position (third factor) of each element since most student reflections followed the format and structure outlined in the task instructions. This does not mean, for example, that the 'most fundamental learning outcomes' section was not regarded at all, but it was taken into account as just one of the potential elements that the students discussed, and it was analyzed as an element of interest if the other two indicators also supported it.

4. OUR ARTIFACT: COURSE ORGANIZATION MODEL

In this section, we describe our approach in developing the GSD course for our university students from diverse cultural backgrounds. The students were taking two different master's degree programs that targeted both international and local students. The students' nationalities for each course iteration are listed in Table 1. The number of local students from 2012 onwards increased because after the first remarkably successful implementation in 2011, the course was made obligatory to all SE students.

The international master's degree program comprised 120 points in the European Credit Transfer and Accumulation System (ECTS), of which GSD was a 5-ECTS course. This course was divided into two parts—one consisted of lectures, and the other entailed exercise work. The lectures involved all the students' participation, as well as assignments, which included reading articles and writing analyses. The total time spent on the lectures amounted to 70 hours (20 hours of lecture attendance, 30 hours of assignments, and 20 hours of additional readings). For the assignments, each student read, summarized, and analyzed selected academic articles. The lectures comprised ten 2-hour sessions. Some of the topics covered included strategic issues in distributed development (offshoring, near-shoring, outsourcing, and open source software; cost-benefit-risk analysis; the triad of coordination, control, and communication; team building (e.g., virtual teams); software process paradigms in the global environment (planned and agile); methods and tools for distributed software development; issues related to allocation of tasks; communication issues arising due to distance and time zone differences; infrastructure support; geographical dispersion; lack of information communication; coordination complexity; cultural issues; technical issues related to information and artifact sharing; architectural design; and finally, knowledge management issues. The lectures and seminars also reviewed current research aspects of GSD and related case studies from industry. The exercises demonstrated distributed software development by a virtual team with the support of appropriate methods and tools.

Table 1. Nationalities of students taking the SDGE course, by year

Course iteration code and year (left), with countries and number of students participating in each iteration (right).	
I1: 2011	Austria (1), China (4), Finland (3), India (1), Iran (2), Morocco (1), Pakistan (2), Slovakia (1), Spain (1), Turkey (1). Total (17).
I2: 2012a	China (8), Colombia (1), Ethiopia (1), Finland (25), India (1), Kenya (2), Morocco (1), Nepal (1), Nigeria (1), Pakistan (1), Russia (1), Spain (2), Tanzania (1), Turkey (1), UK (1), USA (1). Total (49).
I3: 2012b	China (8), Ethiopia (4), Finland (43), Ghana (1), India (1), Iran (2), Nigeria (3), Pakistan (2), Vietnam (1). Total (65).
I4: 2013	Belgium (1), China (7), Ethiopia (2), Finland (45), France (1), Iran (1), Kenya (1), Nepal (2), Nigeria (3), Pakistan (1), Turkey (1), UK (2), Zambia (1). Total (68).

The course, lectures, and exercises were designed to aid in achieving several learning goals. After completing the course, the students should be able to do the following:

- Define the key success factors of GSD and the potential problems in the coordination of projects where teams are separated by physical and/or temporal distance.
- Describe and evaluate the collaborative technologies that best support distributed software development, and choose the methods and tools for distributed software development.
- Apply the GSD practices in a student project, and use the supporting tools throughout the project life cycle.

The exercises offered in the course were performed over 4 to 5 weeks. They included an introduction session for all the students in the beginning and a reflection discussion seminar in the end. During the initial session, the students were introduced to the exercises' simulation idea and the problem domain of the software product to be developed. In between these classes, the students were divided into teams of 3–5 members and organizations of 4 or 5 teams. The objective of the exercise was to highlight the problems that arise in a GSD project because of the distributed and global nature of the project organization. This was attempted to be achieved through a simulation of the geographical and temporal distribution of different teams within an organization by prohibiting direct face-to-face communication among the teams. Between different implementations in the course, the division into teams was done by either the instructors or the students themselves but always with the aim and restriction that each team should consist of members from at least two different cultural backgrounds to create the potential for learning from various ways of working, as well as to simulate the global nature of the assignment.

The roles for each team in the organization were predetermined as follows: a headquarters team coordinating the project, a customer team ordering the product to be developed and providing business requirements for the product, one or more user teams consisting of end-user representatives and working as part of the customer team's company, and one or more subcontractor teams charged with the technical requirements analysis and product development. Additionally, each team was given a geographical location that they were told to maintain in the simulation with regard to the usual work hours in that location's time zone. This organization of teams, time zones, and responsibilities was inspired by a schematic in [4].

The students were also given the task of using agile project management principles during this exercise. It was meant to increase the need for communication among the teams, for instance, between the customer and headquarters teams to enable rapid feedback on the deliveries made during the assignment weeks. The rest of the application of the agile principles varied considerably from team to team because of the diverse backgrounds in project management and software process studies and experiences of the student participants from different cultures. Each week, the teams were each given an assignment according to their respective roles. These assignments required teamwork and coordination among the teams; for all but the first iteration of the course, these assignments only consisted of the initial project kickoff and requirements elicitation phases of the software process. When the course was first offered, the implementation included some design and code rewriting; composition and refactoring were required, but these were considered a distraction from the main learning goal of the exercise—the communication and coordination aspects of a GSD project. The assignments were to be started by the customer team ordering a piece of university guide software from the headquarters team to be used in guiding students, staff, and visitors to various locations within a university, with the help of displays across the campus and a mobile application. More detailed requirements were left to the innovation of the teams, usually the user and customer teams, with some help for the technical requirements from the subcontractor teams. The headquarters team would then go and outsource parts of the software development to its subcontractors, while maintaining the overall product ownership and composition responsibilities, as well as the business relationship with the customer. For its part, the customer team would discuss its business requirements regarding the software with the headquarters team and ask a set of end users (the user teams) to provide user requirements for it. For weekly deliveries, each team was implicitly required to communicate with other teams and coordinate what and when they would need to deliver in order to manage the partial delivery of the software components at the end of each week. At the end of each week, the headquarters team was tasked to organize a steering group meeting with at least the representatives from the customer team and the instructors to clarify issues and maintain overall coordination of the project. This also served as a review meeting of what was delivered of the weekly assignments. At the end of the 3–4 week simulation part of the exercises (except for the first course iteration), the headquarters team was expected to deliver a prototype of the user interface for the system ordered by the customer team, as well as a backlog of the requirements in the form of user stories for the entire system. The teams were also required to submit their logs of their entire electronic communication history.

For the technical aspects of the simulation, the students were only required to use email and the Adobe Connect Pro (ACP) videoconferencing software for communication. They were free to use other tools, as long as all communication was carried out without being co-located with the representatives of other teams and without direct face-to-face communication. The instructors used emails to send the weekly assignments; thus, the students were required to monitor their emails, which many teams also chose as a means of communication with other teams. The ACP software was used for the weekly steering group meeting, with the aim of introducing to the students a closed-source solution for videoconferencing, in addition to all open-source solutions that were freely available to them. Most teams opted to use the conferencing tools with which they were familiar (such as Skype, Google hangouts, or even Internet relay chat) for communication among the teams, but they used ACP for the weekly meetings with the instructors and other teams. Other tools used by the teams for this assignment were Wiki pages, Google docs, and the telephone.

5. EVALUATION

To determine if and how well the exercises raise the relevant issues of GSD for the student participants, two analyses were conducted. First, the student reflection reports were coded, using the aforementioned phenomenographical approach. By assigning a code to each learning-related student quote—a word describing the quote's content, its keyword—and combining these codes into larger themes, the main issues faced by the students during the exercises could be calculated by simply counting the frequencies that each code appeared in the students' reflections. Second, by comparing these themes and the codes they contained with the commonly faced barriers in GSD that are reported in the literature [22], the exercise model's relevance for practice can be evaluated. These two analyses are reported in the following two subsections, respectively.

5.1 Course Artifact

One course evaluation survey was conducted with the students, in the fifth iteration of the GSD course in 2014. This survey aimed to find out the students' opinions on the course in the GSD context. Out of 70 students, 22 participated in the survey. The survey included the following questions, among others: "Were the lectures conducted in a manner that ensured maximum preparation and participation of the students? "Did the lectures provide a means of elaborating on and developing students' own understanding of the global software development subject?" The students were asked to express their views by choosing a number ranging from 1 to 5 for each question, where 1 meant "strongly agree," and 5 indicated "strongly disagree." The survey results showed that more than 50% of the students strongly agreed that the lectures were conducted in a manner that ensured their maximum preparation and participation. Around 95% agreed that the lectures provided a means of elaborating on and developing their own understanding of the GSD subject. Around 82% approved that the onsite simulation helped them to recognize the various aspects of GSD. Based on these results, the overall evaluation was that the students agreed that the course's organizational model was helpful to them in learning the different aspects of GSD. Also, the final grades of all the students were analyzed. The average of final grades turned out to be 3.85 out of 5. Therefore, from this we conclude that from our perspective the students were able to reflect aspects of GSD from this GSD course.

5.2 Student Reflections

Two researchers coded 199 student reflections and cross-checked them for the themes that emerged. These themes were the individual students' personal characteristics, teamwork, communication issues, technical problems and issues with the tools used in the exercise, software process-related issues, the organization of the exercise and the teams within it, project coordination issues, and problems with managing a software project. Many comments dealt with more than one subject, but with these theme divisions, all codes could be covered. Table 2 shows these themes and the codes within each that were most frequently found from the reflections. This frequency includes all course implementations covered in this study, but Table 2 lists these code frequencies by implementation to highlight the changes (or the lack thereof) between implementations. The somewhat lower counts for the codes in earlier implementations can be explained by the changes in the total number of students participating in the exercises, which are visible in Table 1. Many other codes were also found that could be categorized under the theme in question, but for brevity's sake, only the most frequent ones have been included below.

For each theme in Table 2, the codes they contain are explained with some examples of the student quotes that fall under them. It should be pointed out that while these quotes are shown as examples of their respective codes in this context, analyzing the quotes indicates that they can and most have other codes attached to them as well.

5.2.1 Personal Characteristics

The quotes under this theme dealt with the student's own or his or her teammates' attributes regarding their possible previous work experiences and professionalism, abilities to function as skilled professionals in a software project, and psychological factors, such as the motivation to participate in and be responsible for the assignments given.

"Though I have about 4 years of experience in software development work, I [have] not been attending this kind of higher-level meetings with actual customer teams in my real-time work." (experience)

"I was really excited about this exercise and got to work right away when we received [the] first assignments, and we did [a] good job then, but later, when things didn't go that smoothly because of the information problems with other team[s], I found

out that I didn't have that much enthusiasm to perform these tasks than I did earlier." (motivation)

"If I conclude it in [a] few lines, I would say, it is very important that each member [of the] team should have [the] necessary skills to perform the tasks [or] else, it would not [be] possible to complete [the] GSD project even if you [had] good communication skills." (skills)

5.2.2 Teamwork

The students reflected on their learning with regard to the issues in working in teams in general, what happened when people from different cultures tried to work together, what sort of problems arose from the division of responsibilities among teams and team members, what kinds of effects the roles of teams and team members had on working, how to make teams work and perform as a single unit rather than a group of separate individuals, and what part the concept of trust plays in a GSD project.

Table 2. Themes, codes, and their frequencies

Course iteration	*I1*	*I2*	*I3*	*I4*
Personal characteristics				
Experience	0	5	8	12
Motivation	6	2	8	5
Skills	4	0	0	6
Teamwork				
Teamwork	5	9	23	23
Cultural differences	10	7	21	25
Responsibilities	7	7	20	19
Roles	3	7	15	23
Team building	5	7	11	8
Trust	4	2	8	5
Communication				
Communication	24	47	91	126
Language barrier	4	6	7	17
Feedback	0	1	5	5
Understanding	0	0	0	11
Technical interaction				
Support tools	3	15	27	34
Technical	6	1	7	10
Videoconferencing	0	0	3	0
File format	0	0	0	1
Process				
Scheduling	4	18	29	27
Requirements	2	14	20	10
Workflow	4	3	21	8
Deadlines	2	5	6	14
Process	0	5	1	11
Organizations				
Spatial separation	4	11	28	13
Time zones	4	14	8	23
Organization	2	5	5	15
Realism	3	2	4	14
Customer	0	0	12	0
Coordination				
Coordination	11	16	53	45
Confusion	0	1	5	4
Management				
Delivery	13	9	30	29
Management	6	4	21	50

"I think another reason would be language and cultur[al] differences, which cause cooperation problems [among] us. [In] accomplishing the development tasks, I think [that] our group [doesn't] have a clear division of labor and [doesn't] have enough communication." (cultural differences, responsibilities)

"With [N. N.], we were discussing together in [our native language] about the exercises and [their] purpose [...] We were asking each other's opinion about what we should return and when and how. [N. N.] was the 'key player' in my group." (teamwork, roles)

5.2.3 Communication

While understandably the most frequently cited topic in the students' reflections, given the learning objective of the exercise, communication issues did show that it was generally difficult to get one's message across to other people in the GSD environment. Other topics relating to communication issues dealt with people with different linguistic backgrounds, problems in receiving feedback and responses from other people, and understanding the meaning of what was communicated even if the language used was not a problem.

"I observed how video and email communications [could not] fully substitute [for] face-to-face communication; for example, during the meeting using Google hangout, some of the chairpersons in the meeting [were] native speakers of [the] English language, and during the meeting, it [was] difficult to follow those native speakers' speech." (communication, language barrier)

"It is interesting to see how different the requirements seem when they are only in text form. It makes [you feel that] the product [is] less understood when nobody has explained it in an imaginative ecstasy, making you grasp some of that enthusiasm, too. You just do what is told in that document, and that's it. If something is missing from the documentation, it's harder to make a note of it to HQ for review than just go down the hallway to tell [...] the right person [about it]." (communication, understanding)

5.2.4 Technical Interaction

All comments regarding technical difficulties and issues with the tools used were categorized under the theme of technical interaction. This also included issues regarding videoconferencing as a means of communication, problems with file formats and different platforms such as operating systems while transmitting data from one person to another.

"I realize [...] how hard it [is] to communicate with streaming software if we have to use WLAN, or [the] Internet is slow; also, it's hard to come up [with] any good conversation or make any good points when one's voice is breaking [from] time to time." (technical, support tools)

"[The] most important thing that I learned during this exercise was getting used to videoconferencing. I [had] not been in [a] videoconference of this magnitude before." (support tools, videoconferencing)

5.2.5 Process

Thoughts regarding the general software production process and its parts were visibly present in the students' reflections. Issues with scheduling suitable meeting times and fitting the work into other responsibilities were common musings, as were requirements gathering as a process phase in general although this could at least be partly explained by the scope of the exercise. Other issues dealing with process included how to organize the workflow in a GSD environment; how to deal with the deadlines, given the restrictions imposed by the workflow and different teams participating in it; and also issues regarding the process models used, such as scrum.

"Setting up [the] times for meeting was also difficult since there [were] other courses, and people also [had] their individual schedules, so I think it's really important to keep the deadlines. This was a problem, especially in the last week of the simulation; because some groups had difficulties mak[ing] it to the deadlines, our work was delayed as well." (scheduling, deadlines)

"I don't think we at any point actually had a real, coherent idea of what the software to be implemented would actually do and who would actually use it. We had a number of scattered details, we had some documentation, but we never got together and actually hammered out canonical answers to questions like: Who are the relevant stakeholders and their interests? Why would people use the program in the first place? What is the scope of the software, and what is outside of that scope?" (requirements)

"At the planning stage of the project, we discussed [about] hav[ing] a specific day for our team meeting because it [would] provide us [with a] uniform working environment. However, we could not act as we planned it because delays from other teams prevent[ed] us [from] mov[ing] forward." (workflow, scheduling)

5.2.6 Organizations

The students encountered issues such as how the exercise was organized to simulate spatial separation in GSD by eliminating face-to-face communication, as well as temporal separation by assigning a dedicated time zone to each team, which fell under the theme of organizations. Moreover, particular roles within the organizations, such as the comments received from customers, along with the generic feedback on how the entire exercise was organized and how well it represented the real situation from the students' perspective, were mentioned under this theme.

"The thing that didn't help me was the fact that we couldn't meet other teams face to face. I know that that was one of the main points [of] this exercise, and I surely understand why. Working with people from multiple locations around the globe is difficult, not only because [of] different time zones and cultures but also because meeting and getting familiar with each other [are] more challenging. Because of these factors, there is [a] great possibility [of having] major misunderstandings that may increase [the duration] and costs of a project." (spatial separation, time zones)

"When we [could] not meet within our team during our time zone's work hours, we basically had to do the work outside of the time zone and only send emails and return deliverables in the morning. So maybe it would be better to have everyone in the same time zone and just located apart from each other so there would still not be any face-to-face meetings between teams." (time zones, organization)

"Problems like time difference[s], inadequate specifications, people who do not answer their e-mails or telephone[s], combined [with] the fact that they might be on the other side of the globe, make things somewhat harder. This exercise can, however, point out that working with people in the same building as you are [in] is different [from] working with people in other locations, and you should be prepared for that." (time zones, realism)

5.2.7 Coordination

This theme consisted of two codes only: first, the coordination of work, schedules, and the work product itself, and second, the confusion caused by the lack of coordination. However, having quite a high combined frequency, they were considered viable candidates for constituting a theme on their own.

"The most important lesson to be learned was definitely how the lack of coordination affect[ed] the progress. We knew when we were supposed to deliver our results, but we didn't have reliable information about when we [would] get the necessary instructions. This made it impossible (or at least risky) to schedule a meeting if we weren't sure that they [would be] delivered to us in time." (coordination)

"There was [a] good amount of info[rmation] on the assignments, [on] what everyone should be doing, but still, there was confusion among groups about what they should do and with who[m]. This is something that I think needs more work by everyone involved in this simulation." (confusion)

5.2.8 Management

While being close to the theme of coordination, management issues, such as people management in a project, were perceived as different from coordination, which warranted a separate theme. Another major issue from the students' reflections that fell under the management theme was that of delivering something on time and with reasonable content—this could be either a work product or simply just promptly answering an email query.

"In [the] overall progress, missing a clear leader did not affect the team performance, but [the] presence of one would have enabled us to organize tasks better. I suppose we did find selecting a team leader a difficult task." (management)

"I think it was really interesting to see how to keep a project running and [the] high satisfaction among the groups. [By this,] I mean [the] trust between the customer and HQ and also between the HQ and [the] subcontractor. Our role[s] [were] to keep together all pieces of the project and control the flow of information. We had some problems get[ting] some things like prototypes delivered to us just in time, and because of that, we [had] to explain [...] to the customer why these happen[ed]." (management, delivery)

5.2.9 Top Five Challenges Faced

During our data analyses, we found many challenges that students conveyed during the course simulation. However, in this section, we describe only the top five challenges that were identified, as reported by the students in their reflection reports. These were communication, coordination, management, delivery, and support tools.

Communication: The first and most challenging experience noticed during the simulation exercise was communication. During the exploration, more than 288 pieces of evidence of communication as a barrier were found in the reflection reports. In the GSD context, where teams and stakeholders were dispersed remotely, communicating the work with the stakeholders was a challenge. As one student advocated, "*The exercise [has] made [it] very clear that communication in projects is essential. Communication also creates some pressure to the person in command. By pressure, I mean you have to be able to trust the other participants.*"

Coordination: The next challenge observed during the analysis was the coordination among the teams and within each team. Around 125 supportive pieces of evidence were found regarding this issue during the simulation exercise. In a remote environment, it would not be easy to coordinate among different stakeholders, as well as distributed teams, to complete the work. As one student described, "*During [the] handoff, which in this case means when giving results to another team to work on, there should be a clear and thorough understanding of what is to be handed off. This problem occurred a few times during the course of the simulation, as we got results very much different [from] what was expected a couple of times.*"

Management: Management was reported as the third challenge, based on the students' experiences. Overall, 81 pieces of evidence were found regarding this concern. As one student stated, "*Working as a part of a larger organization was also quite difficult. We had to ask for some clarification on some of the assignments from HQ. For example, we didn't know if we were supposed to use a certain way of doing the time estimates so that our estimates would be comparable to the other subcontractor groups' time estimates.*"

Delivery: The fourth challenge was identified in the context of the delivery of work items. Around 81 associated pieces of evidence were found regarding delivery during the GSD simulation. The delivery of project work in the GSD context in accordance with the release plan was a challenge. As one student mentioned, "*Setting up [the] times for meeting was also difficult since there [were] other courses, and people also [had] their individual schedules, so I think it's really important to keep the deadlines. This was a problem, especially in the last week of the simulation; because some groups had difficulties mak[ing] it to the deadlines, our work was delayed as well.*"

Support tools: The fifth challenge observed was from the perspective of the support tools. Around 79 pieces of evidence were found in this regard. As one student explained, "*[The] biggest problem we faced was people not checking their mails on time, and using conference calls didn't work really well since the connection in the university [wasn't] that good, unless you get to use the university network.*"

5.2.10 Comparison to Learning Objectives

Because of the issues raised by the students in their top five challenges, we can state that the learning objectives of the course (as described in Section 4) were met. This is due to management, communication, coordination, and delivery being the key factors in a GSD project (see Table 3), which comprised the first learning objective. The second objective, being able to select and use proper tools for the GSD environment, was identified as its own top challenge. These top challenges, combined with the students' actual completion of the exercise, showed that the third learning goal of being able to apply the learned issues in practice was also achieved.

5.3 Correlation to Theoretical Framework

While the themes presented above somewhat differ from the barriers discussed in [22] when the individual codes within the themes are examined more closely, the codes can be mapped to highlight the barriers presented. This mapping is shown in Table 3, with the codes and their total frequencies mapped for each barrier. Table 3 also lists the percentage of each code dealing with the barrier in question compared to the total codes for each course iteration. While the individual percentages may be relatively low, if combined by iteration, the codes from the reflections cover all but the barrier of the product architecture, which is understandable, given the focus of the exercise. Moreover, the combined percentages of each iteration for the codes discussing the barriers are 50–61%, meaning that at least every other student comment relating to what they learned during the exercise deals with the common barriers in GSD.

Table 3. Results mapped against GSD barriers from literature

Main category barriers [22]	I1	I2	I3	I4
Language and cultural distance				
Cultural differences, language barrier	7.95%	5.00%	5.15%	6.10%
Temporal distance				
Time zones, asynchronous	2.84%	5.38%	1.47%	3.34%
Geographic distance				
Spatial separation	2.27%	4.23%	5.15%	1.89%
Process and management issues				
Delivery, management, coordination, scheduling, requirements, workflow, deadlines, process	23.86%	28.46%	33.27%	28.20%
Fear and trust				
Experience, trust, responsibilities	6.25%	5.38%	6.62%	5.23%
Infrastructure				
Support tools, technical, videoconferencing, platform	5.68%	6.15%	6.80%	6.40%
Organization				
Organization, customer	1.14%	1.92%	3.13%	2.18%
Product architecture	No indication			

6. DISCUSSION

Given that the exercise in question aimed at highlighting the problems of a GSD project, one of which was how to effectively communicate from a distance, it should not be surprising that the single most commented issue in the student reflections was communication. This further indicated that the exercises showcased the main learning objective as planned in the course model. Themes about teamwork problems and individual performance abilities in a project are potential subjects for discussion in any given work assignment that students complete in teams. Perhaps somewhat unexpected was the rather low amount of feedback concerning the technical problems and support tools although these issues were predominant during the exercises, at least from the instructor's viewpoint, based on the discussions with the students and the amount of instruction requests during the exercises. On the other end of the spectrum was the amount of feedback discussing more traditional SE problems, such as how to follow the defined software process and manage a student's own project team and the software product under development, which did not necessarily stem solely from the simulated distributed setting of the exercise. However, the literature review also identified process and management issues as a main barrier in GSD, which was also the most prominently discussed topic in the student reflections. Partly because of this and other feedback as well, during each iteration, more often than every other topic, the students discussed what they learned from the barriers identified in previous GSD literature. Without conducting a longitudinal study where the same students would participate when they eventually became involved in a GSD project in an industrial setting, this gives a strong indication as is possible towards the practical and industrial relevance of the exercise, as organized by the model suggested above.

A set of seven guidelines for design science research is presented in [13]. Our study takes them into account, as follows. The course model presented is a design artifact that addresses a relevant problem to both academia and industry, outlined by the RQ (guidelines 1, 2, and 4). This model was evaluated based on empirical data gained from experienced teachers' three-year development, implementation, and systematic improvement of the course by applying scientifically verified teaching methods. The data was analyzed with an established method of phenomenography, which was rigorously followed by the authors (guidelines 3, 5, and 6). This paper represents the 7th guideline of communicating the results to a relevant scientific community.

Summarizing the findings in relation to the RQ, we state that organizing the GSD course in an onsite context is a feasible approach, of which our artifact—the course organization model—is a practical demonstration. More than half of the analyzed data describes the encountered challenges, such as communication, coordination, management, and language and cultural distance during the course simulation. These challenges are often considered common during software development in the global environment. However, the socio-cultural aspects weren't considered much as a challenge here since the student teams were formed to reflect a multicultural environment. The virtual teams created within one institution require less time for coordination, when compared to the coordination of virtual teams performed in several institutions. Hence, our results demonstrate that the onsite simulation can be efficient, economical, and can offer a relevant perspective of the challenges posed by the global environment to a software development project.

Monasor et al. [21] did not report (nor have we have found) any empirical data available for comparing coordination efforts between same-site and multisite GSD courses except for [23], but there is ample evidence of increased coordination needs for real multisite GSD efforts to justify increased coordination needs for multisite GSD courses as well [3, 10]. Interestingly, just recently, Paasivaara et al. [23] reported on learning global agile SE using same-site and cross-site student teams. According to the students they interviewed and the data they gathered, they mainly found no significant difference in learning while working in local versus global teams. However, while their experiment was basically cross-site with some parts of the work done in the same-site mode, they did not report anything about the costs of organizing their course model.

6.1 Validity Discussion

This section discusses validity in terms of the construct, internal validity, external validity, and reliability, as defined by [26]. In construct validity, it is important to know whether the researchers took the right measures to examine the phenomenon under study and collected extensive data so that its scope sufficiently addressed the research questions asked. The reflection report, with its probing questions, was designed to prompt the students to reflect on whether our research objective was met, from their perspective. We also reviewed the reflection report along with a student representative to ensure that the students would properly understand the questions. The emphasis on internal validity is intended to establish a causal relationship and is mainly used for explanatory and causal studies. Since our study was of an exploratory nature, internal validity was not considered.

External validity states how the results of a study can be generalized. The results of this research were limited to a single-institution context. However, for institutions interested in an onsite simulation for their GSD course, this study's findings can be relevant and valuable. Reliability is concerned with how the data and its analysis depend on the specific researchers. During our data analysis, the first and second authors both analyzed the reflection report data to discover the challenges faced in the simulation exercise from various viewpoints. The third author audited and monitored the entire process.

7. CONCLUSIONS

This paper has described our research on an onsite simulation of a GSD course, which is efficient, economical, relevant, and previews the challenges faced by the students in GSD projects. In the reviewed literature, various studies have reflected on conducting a GSD simulation, but such simulations usually involve collaboration among several institutions. However, the literature offers vague explanations about carrying out an onsite simulation within one institution. This paper fills this research gap by describing the course organization model.

According to the data analysis results, the main component of the higher-education GSD course is practical work that utilizes multicultural student teams engaged in an educational role-playing simulation of a GSD project. This project should focus on highlighting potential problems in communication, teamwork, the software process, management, technical framework, and individual professional development. Our recommendation for other teachers is that multicultural student teams should be created in the beginning of the exercise, teams should not have face-to-face discussion and they should simulate the time-zone differences among them. From the data analysis and its comparison with the previously published work on industrial GSD projects and their challenges, it can be concluded that the course (especially its exercises conducted according to the model presented in this paper) gives the participating students insights about GSD that are similar to those working in industrial GSD projects.

This study focuses on a single institution but is supported by its three-year experience in implementation as its source material.

Therefore, further studies could examine whether the results would be similar in another higher educational institution, with possibly different kinds of student backgrounds. Moreover, the effects of continuing the exercise for a longer duration would appear to have no added benefits, except for the possible inclusion of the product architecture barrier in the highlighted GSD problems, but this issue could be studied further.

8. REFERENCES

[1] Andrianoff, S.K. and Levine, D.B. 2002. Role playing in an object-oriented world. *ACM SIGCSE Bulletin*. 34, 1 (Mar. 2002), 121.

[2] Armstrong, E.K. 2003. Applications of Role-Playing in Tourism Management Teaching: An Evaluation of a Learning Method. *The Journal of Hospitality Leisure Sport and Tourism*. 2, 1 (Apr. 2003), 5–16.

[3] Carmel, E. 1999. *Global software teams: collaborating across borders and time zones*. Prentice Hall, NJ, USA.

[4] Carmel, E. and Espinosa, J.A. 2011. *I'm Working While They're Sleeping*. Nedder Stream Press, USA.

[5] Errington, E. 1997. *Role play*. Hihger Education Research and Development Society of Australasia, Australian Capital Territory, Australia.

[6] 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* (Zurich, Switzerland, June 02-09, 2012), IEEE, Piscataway, NJ, 1–5.

[7] Garvey, D. and Garvey, S. 1967. Simulation, Role-Playing, and Sociodrama in the Social Studies. With an Annotated Bibliography. *The Emporia State Research Studies*. 16, 2 (Dec. 1967), 5–34.

[8] Harvey, M. and Baumann, C. 2012. Using student reflections to explore curriculum alignment. *Asian Social Science*. 8, 14 (Nov. 2012), 9–18.

[9] Heiskanen, A., Newman, M. and Eklin, M. 2008. Control, trust, power, and the dynamics of information system outsourcing relationships: A process study of contractual software development. *Journal of Strategic Information Systems*. 17, 4 (Dec. 2008), 268–286.

[10] Herbsleb, J.D. and Mockus, A. 2003. An empirical study of speed and communication in globally distributed software development. *IEEE Transactions on Software Engineering*. 29, 6 (Jun. 2003), 481–494.

[11] Herbsleb, J.D. and Moitra, D. 2001. Global software development. *IEEE Software*. 18, 2 (Mar. 2001), 16–20.

[12] Hevner, A. and Chatterjee, S. 2010. *Design Research in Information Systems*. Springer US, New York, NY. New York.

[13] Hevner, A.R., March, S.T., Park, J. and Ram, S. 2004. Design science in information systems research. *MIS quarterly*. 28, 1 (Mar. 2004), 75–105.

[14] Honig, W.L. and Prasad, T. 2007. A classroom outsourcing experience for software engineering learning. *ACM SIGCSE Bulletin*. 39, 3 (Jun. 2007), 181–185.

[15] Höst, M., Regnell, B. and Wohlin, C. 2000. Using students as subjects—a comparative study of students and professionals in lead-time impact assessment. *Empirical Software Engineering*. 5, 3 (Nov. 2000), 201–214.

[16] Keenan, E., Steele, A. and Jia, X. 2010. Simulating Global Software Development in a Course Environment. In *2010 5th IEEE International Conference on Global Software Engineering* (Princeton, NJ, August 23-26, 2010), IEEE, Los Alamitos, CA, 201–205.

[17] Kinnunen, P. and Simon, B. 2012. Phenomenography and grounded theory as research methods in computing education research field. *Computer Science Education*. 22, 2 (Jun. 2012), 199–218.

[18] Lago, P., Muccini, H. and Babar, M.A. 2008. Developing a Course on Designing Software in Globally Distributed Teams. In *2008 IEEE International Conference on Global Software Engineering* (Bangalore, India, August 17-20, 2008), IEEE, Los Alamitos, CA, 249–253.

[19] Lago, P., Muccini, H., Beus-Dukic, L., Crnkovic, I., Punnekkat, S. and Vliet, H. Van 2007. Towards a European Master Programme on Global Software Engineering. In *20th Conference on Software Engineering Education & Training (CSEET'07)* (Dublin, Ireland, July 03-05, 2007), IEEE, Los Alamitos, CA, 184–194.

[20] Levine, D. 2000. Helping students through multiplicities. *Journal of Computing Sciences in Colleges*. 15, 5 (May 2000), 289–295.

[21] Monasor, M.J., Vizcaino, A., Piattini, M. and Caballero, I. 2010. Preparing Students and Engineers for Global Software Development: A Systematic Review. In *2010 5th IEEE International Conference on Global Software Engineering* (Princeton, NJ, August 23-26, 2010), IEEE, Los Alamitos, CA, 177–186.

[22] Noll, J., Beecham, S. and Richardson, I. 2010. Global software development and collaboration: barriers and solutions. *ACM Inroads*. 1, 3 (Sep. 2010), 66–78.

[23] Paasivaara, M., Blincoe, K., Lassenius, C., Damian, D., Sheoran, J., Harrison, F., Chhabra, P., Yussuf, A. and Isotalo, V. 2015. Learning Global Agile Software Engineering Using Same-Site and Cross-Site Teams. In *Proceedings of the 37th International Conference on Software Engineering* (Florence, Italy, May 16-24, 2015), IEEE, Piscataway, NJ, 285–294.

[24] Rarieya, J. 2005. Promoting and investigating students' uptake of reflective practice: a Pakistan case. *Reflective Practice*. 6, 2 (May 2005), 285–294.

[25] Richardson, I., Moore, S., Paulish, D., Casey, V. and Zage, D. 2007. Globalizing Software Development in the Local Classroom. *20th Conference on Software Engineering Education & Training (CSEET'07)*. (Dublin, Ireland, July 03-05, 2007), IEEE, Los Alamitos, CA, 64–71.

[26] Runeson, P. and Höst, M. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering*. 14, 2 (2009), 131–164.

[27] Schneider, S., Torkar, R. and Gorschek, T. 2013. Solutions in global software engineering: A systematic literature review. *International Journal of Information Management*. 33, 1 (Feb. 2013), 119–132.

[28] Sjöström, B. and Dahlgren, L.O. 2002. Applying phenomenography in nursing research. *Journal of Advanced Nursing*. 40, 3 (Nov. 2002), 339–345.

[29] Šmite, D., Wohlin, C., Gorschek, T. and Feldt, R. 2009. Empirical evidence in global software engineering: a systematic review. *Empirical Software Engineering*. 15, 1 (Dec. 2009), 91–118.

[30] Youssef, L.S. 2012. Using student reflections in the formative evaluation of instruction: a course-integrated approach. *Reflective Practice*. 13, 2 (Apr. 2012), 237–254.

[31] Zowghi, D. and Paryani, S. 2003. Teaching Requirements Engineering through Role Playing: Lessons Learnt. *Proceedings of the 11th IEEE International Requirements Engineering Conference*. (Sep. 2003), 233 – 241.