An Experiment on Teaching Coordination in a Globally Distributed Software Engineering Class

Martin Nordio H.-Christian Estler Bertrand Meyer
Department of Computer Science, ETH Zurich, Switzerland
firstname.lastname@inf.ethz.ch

Nazareno Aguirre
University of Rio Cuarto, Argentina
naguirre@dc.exa.unrc.edu.ar

Elisabetta Di Nitto
Politecnico di Milano, Italy
dinitto@elet.polimi.it

Rafael Prikladnicki
PUCRS, Brazil
firstname.lastname@pucrs.br

Anthony Savidis
University of Crete, Greece
as@ics.forth.gr

Abstract

The importance of planning and management skills in software development is very difficult to convey in software engineering courses. We present the synopsis of an assignment whose purpose is to demonstrate the significance of such skills, including effective communication, team coordination and collaboration, and overall project planning. The assignment is organized in the context of a distributed software engineering course carried out in collaboration with 12 universities in South America, Europe and Africa. The assignment is a globally distributed contest issued before *most development activities related to the course's software project are performed, aiming at favoring the collaboration between students prior to project development. The contest does not involve any programming, and is not related to the project development activities. Instead, it consists of making teams in different countries compete in collaboratively solving a set of very simple tasks. The complexity of the activity is in team collaboration and coordination, and their lack is evident when the tasks are not correctly solved, or not solved in time. Despite the simplicity of the assignment, students have found it useful in helping them understand the significance of management and planning challenges in distributed software development. Moreover, the assignment helped in team building, by creating a better team atmosphere and contributing in identifying team members better suited for management.*

1. Introduction

Including software development projects as part of software engineering (SE) courses is a widely adopted practice. A main motivation for including projects in SE courses is that they allow us to reproduce, in part, the conditions under which a real software project is developed, forcing students to face the technical and management difficulties of an industrial setting. As actual development experiences, projects in SE courses help instructors to illustrate various issues, such as the importance of SE methods and practices (e.g., by showing students how to apply software development and project management methodologies), or introducing the students to social issues regarding software development (e.g., problems in communication and how to deal with them).

978-1-4799-4970-0/14/$31.00 © 2014 IEEE CSEE&T 2014, Klagenfurt, Austria

In recent years, some SE courses have incorporated *globally distributed projects* [17, 1, 15]. In these cases, projects are developed by teams located in different countries, illustrating the challenges of distributed software development. Besides the organizational barriers the instructors face [3] and the communication and collaboration issues the students face [13], these projects, distributed over several countries, continents and time zones, make SE concepts such as requirements engineering, project management, and API design even more relevant [14]. For example, a bad API design in a component developed in Brazil might produce integration problems for a component developed in Greece, producing delays or even the failure of the entire project [16].

Globally distributed projects offer an interesting setting for teaching SE. This setting, however, introduces the risk that students mainly face communication and cultural issues limiting the exploration of other SE challenges such as requirements engineering, API design, and software verification. Moreover, despite the clear barriers that cultural, time zone and language differences constitute, many students still tend to underestimate the criticality of effective management and planning; when they finally realize their importance, it is sometimes too late. Thus, a particularly difficult problem is how to prepare students to address these challenges, in particular communication and project management ones, and how to make sure students understand their crucial importance, before the course project is developed.

We present an approach for dealing with the above described problem, consisting of a *globally distributed contest*. It is carried out in the context of a distributed software development course, called DOSE, organized in collaboration with 12 universities located in 11 countries in South America, Europe and Africa. The contest is a globally distributed activity issued *before* most development activities related to the course's distributed software project are performed, aiming at favoring the collaboration between students prior to the project development. The contest does not involve any programming, and is not related to the project development activities. Instead, it consists of making teams in different countries compete in collaboratively solving a set of very simple tasks. The complexity of the activity is in team collaboration and coordination, and their lack is evident when the tasks are not correctly solved, or not solved in time. Tasks of an assignment are distributed among the members of a group, with its participants located in 2-3 countries.

The first main goal of the contest is to prepare students to face communication and project management issues: participating in this contest allows them to learn from their own mistakes and find better ways of organize the group. The second main goal is to unify the group: since the group members are located in different countries, students have never met each other in person; it is important then to increase the *trust* within groups, to facilitate that their members share a common vision for the project. Despite the simplicity of the assignment, students have found it useful in helping them understand the significance of management and planning challenges in distributed software development. Moreover, the assignment helped in team building, by creating a better team atmosphere and contributing to identifying team members better suited for management. The approach is novel in that it does not involve programming or project related activities, while being applicable to highly distributed teams, and pursuing not only better management lessons, but also better team building strategies.

In this paper we describe the contest and its evaluation performed in DOSE 2012 and compare the outcomes of the projects to those in DOSE 2011 (which did not feature the contest). We believe this contest can also be organized in courses where all the team members are co-located, for example, organizing it as an on-line activity, maintaining its main design goals.

2. DOSE: A Globally Distributed Course

Since 2007, ETH has been organizing a course about "Distributed and Outsourced Software Engineering" (DOSE) with a novel component: *a globally distributed project* [14, 15]. The DOSE course targets masters students with good experience in programming and some prior knowledge in SE. Since the beginning of DOSE in 2007, we have collaborated with 17 universities located in South America, Europe, Africa, and Asia. In 2012, the project was developed in collaboration with the following universities: (1) ETH Zurich, Switzerland; (2) Cairo University, Egypt; (3) ITMO, Russia; (4) Odessa Polytechnic National University, Ukraine; (5) Politecnico di Milano, Italy; (6) Pontificia Universidade Catolica do Rio Grande do Sul (PUCRS), Brazil; (7) State University of Nizhny Novgorod, Russia; (8) University of Crete, Greece; (9) University of Debrecen, Hungary; (10) University of Rio Cuarto, Argentina; (11) University of Zurich; (12) Universidad Politecnica de Madrid, Spain. Overall, there were 169 people involved in the DOSE project: 147 students and 22 instructors.

The goal of the course is to prepare students for the challenges of distributed software development, including time zones and cultural difference challenges [7, 13, 8], requirements engineering and API design challenges [16], and project management challenges.

The project topic for DOSE 2012 was a game platform for networked multi-player card games where players could log in and choose a game to play. The project contained 22 game subcomponents, each corresponding to a different game, being developed by one group. A group is formed by 2-3 teams located in 2-3 different countries. For each game, one team implemented the logic of the game and the other team implemented the graphical user interface and the network communication. In the case of 3-teams groups, the third team implemented an Artificial Intelligence component. The project is organized in the following four phases: Phase 1: Scope document (2 weeks); Phase 2: Requirements (2 weeks); Phase 3: Interface specification (2 week); Phase 4: Implementation and Testing (6 weeks). The contest was performed after *Phase 1*, during the third week of the project.

3. Contest Setup

To participate in the contest, each group has to register with *6 members*, each selecting a *role* in the contest: A, $B1$, $B2$, $C1$, $C2$, or $C3$. Role A has been called the *group manager*, and he/she could be located at any country. The students in roles $B1$ and $B2$ had to be located in two different countries, and the students in roles $C1$, $C2$ and $C3$ had to be located in at least two different countries (three if possible). Given that DOSE had some groups with teams in two countries, and some groups with teams in three countries, we only requested roles $C1 - C3$ to be located in two different countries.

The contest took place the third week of the DOSE project; students have started to work in the project, and using e-mail and Skype as communication tools. One week before the contest took place, the students received the information about the registration and the roles. This information only mentions that a task had to be solved, and it would not be complex. During the competition, they were allowed to use any communication tool (Skype, e-mail, etc.). When the competition started, each participating student received his role card by e-mail, containing: (1) a description of the contest, (2) a list of tasks to solve, (3) an array of integers, and (4) a comic.

The contest description explains that every group member has a role card containing a different array (named by the role card) of a different length, a comic and a list of tasks to solve. Most tasks, precisely all except one, focus on executing some computation on the array; this array is

small, containing 32–38 randomly generated integers, which range from 1 to 150. The comic was different for each student, and it was used to assign a task that involves all team members. However, this task was also very simple and consisted in finding the name of the girl that appears in the comic. The tasks have a unique identifier and they might have been assigned to more than one student, however, it was sufficient that only one of them completed it. There were a total of 17 different tasks distributed in the six role cards, but the students did not know who had which tasks. The tasks were simple; for example, a task could be finding the maximum element in the arrays of roles $C1$, $C2$ and $C3$, thus involving the arrays of other team members. The most complex task was: *Find the element at the 12th position of the combined sorted array $C1 + C2 + C3$ where all the duplicates have been removed. The array should be sorted in ascending order.*

All role cards mentioned that the solution had to be sent through the person with role A. The description of all role cards was the same except for role A: only the leader knew from his role card that just 8 out of the 17 tasks of the entire group had to be solved, and it was sufficient that one participant solves the task. The role cards were designed in a way that if they distributed the tasks, and organized well, they would finish earlier. The role cards $B1$ and $B2$ contained the same tasks; to solve them, the students only needed the arrays $B1$ and $B2$. The role cards $C1$, $C2$ and $C3$ were designed in the same way. Only one task was included in all the role cards; it consisted of calculating the length of the combined sorted array $A + B1 + B2 + C1 + C2 + C3$. The *call for participation* and the role cards can be found at `http://se.inf.ethz.ch/people/nordio/papers/dose_contest`.

4. Assessment

4.1. Contest Outcome

Table 1 shows the list of groups that participated in the contest with their final position (POS), the countries where they were located, the time taken to solve the tasks, the number of correct answers obtained, whether they solved more tasks than the 8 tasks required, and the communication tools used during the contest. In the case of the use of text chat for communication, the table also shows the total number of messages (# MSG), total number of characters without spaces (# CHAR) and the number of messages per minute (# MSG PER MIN). Since the groups had to solve only 8 simple tasks, we estimated that the assignment could be solved in 20 minutes, assuming the groups are well organized. Notice that the group in the 9th position solved the tasks in 29 minutes, although they got one result wrong. Most of the other groups solved the exercises in about 45–100 minutes. Also, most of the groups solved more tasks than the 8 required. In some cases, the reason was that the person in role A either did not tell the others what tasks to solve or he told them too late. In a few cases, the person in role A did tell the other members, although some members did not see the message and solved more tasks anyway. It is worth noticing that the average number of messages per minute is quite high: 2.0–4.5 in most cases; this means that every member would read and sometimes write a message every 15–30 seconds.

Table 1 shows that about one third of the groups reported at least one wrong result. In most cases, this wrong result was reported in the more complex tasks, which requested to sort a combined array, for example $C1 + C2 + C3$. We believe the tasks to be at an adequate level of complexity: tasks are not complex, but at the same time they are not trivial.

Table 1: **Contest Outcome in DOSE 2012.**

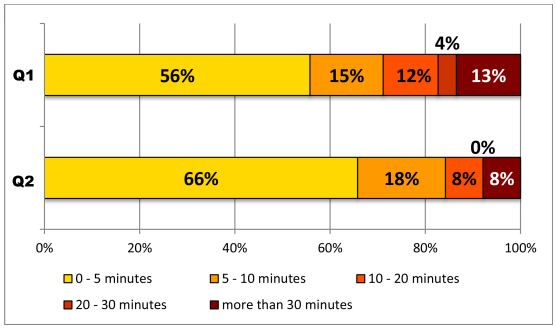
#	Countries	Time	#Correct Answers	Solved more tasks?	Tools	# MSG	# Char	#MSG per Min
1	Greece and Italy	0:44hs	8	yes	text chat	184	8,794	4.2
2	Brazil, Spain and Switzerland	1:02hs	8	yes	text chat	161	17,347	2.6
3	Brazil and Greece	1:03hs	8	no	text chat, web docs, e-mail, dropbox	282	16,154	4.5
4	Brazil and Greece	1:05hs	8	yes	voice chat, text chat	69	4,982	1.1
5	Argentina and Brazil	1:07hs	8	yes	voice chat	n/a	n/a	n/a
6	Argentina, Brazil and Greece	1:30hs	8	yes	voice chat, text chat, e-mail	172	7,229	2.0
7	Argentina and Greece	1:37hs	8	no	voice chat, text chat, web docs, e-mail	207	12,866	2.1
8	Argentina and Brazil	1:43hs	8	yes	text chat, web docs, e-mail	115	7,579	1.2
9	Argentina, Greece and Switzerland	0:29hs	7	yes	voice chat, text chat, web docs, e-mail	83	4,205	2.8
10	Argentina and Italy	0:51hs	7	no	text chat, web docs	213	11,376	4.2
11	Egypt, Greece and Italy	1:26hs	7	yes	text chat	173	12,048	2.0
12	Argentina and Italy	0:44hs	6	yes	text chat, web docs, e-mail	180	10,310	4.1
13	Argentina and Italy	0:51hs	5	yes	text chat	147	7,992	2.9
14	Argentina and Greece	1:19hs	5	yes	text chat, web docs	197	10,627	2.5

4.2. How the Groups Proceeded to Solve the Tasks

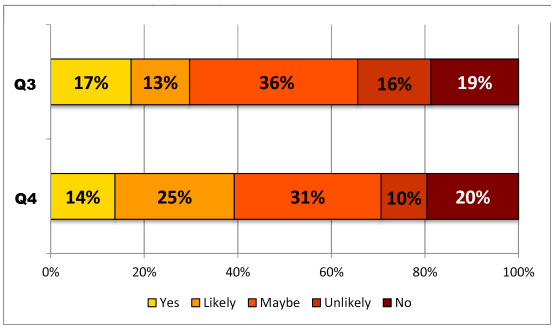
In the process of solving the tasks assigned, more than 60% of the groups did not have any coordination at all. Basically, each group member started to solve his own tasks independently, and just requested the necessary data from the other members. The main source for the lack of coordination was due to the person in role A not doing his job appropriately. However, in several cases, role A tried to coordinate the activities of the other members, but the group members did not follow his advice. In the feedback provided by the students, they reported:

- *Everybody went crazy trying to immediately solving their tasks without any kind of organization.*
- *Our task solving was complete chaos inside a GoogleDoc.*
- *It was hard to keep updated with all the messages in the chat, and also it was a bit stressing not to know which tasks I should complete first since in my role card, I had some common tasks and some tasks that were not relevant for the contest.*

In several cases, the process employed for solving the tasks was the following. First, a member started by coordinating the group. As a first step, all the tasks were collected and added to a web document, or broadcasted to all the members via text chat. Then, each member took one task to solve, and started to work on that task. Only one person was working on each task. To obtain information of the arrays, the also added them to the shared document, or sent the information via text chat. Once someone finished a task, the result was added to the shared document, or informed to his peers, and he took another task. Although this approach proved effective, notice that the group members did not split tasks into smaller teams, and instead they solved the problems sequentially.



(a) Task notification and group manager election



(b) Changing group communication after contest

Figure 1: **Q1: How long did it take until role A notified the group which tasks to solve? Q2: How long did it take until your group decided to define a project manager? Q3: Will you review and maybe change the way your group communicates in the DOSE course? Q4: will you assign one person the role of project manager (if not already done) for your DOSE project?**

4.3. Impact for Distributed Projects

In order to evaluate the impact of the contest in revising group organization and communication strategies, we used an on-line questionnaire, which was answered by the participating students. Immediately after the students finished the exercise, they received an e-mail explaining the contest's setup. We provided this information to make sure that the students understood the full picture of how the exercise was supposed to work, and to let them analyze how they could have performed better as a group. Subsequently, the students filled in the questionnaire; the provided feedback represents 75% of the students participating in the contest. Figure 1a shows the findings about group management. Question *Q1* asked how long it took role A to share with the rest of the group the information about which tasks needed to be solved. While the majority of answers indicated that the information was shared early on, we also observed that about 30% of the students had to wait at least 10 minutes before they were informed which tasks they had to solve. In fact, a significant number of students, 13%, had to wait more than 30 minutes. This delay had a high impact in the performance of the groups: 80% of the groups solved tasks which were not required.

We also observed a similar time-span distribution for question *Q2*: the election of the group manager. The feedback also shows an interesting aspect of this selection: in 30% of the groups, the tasks were not coordinated by role A, which would have been the natural choice.

Table 2 shows the students feedback about whether the contest illustrated the communication and project management challenges in distributed software development. The results suggest that the contest was successful in exhibiting these challenges. One of our motivations for the contest was to show the students some of the challenges of group communication and management. In order to determine if the contest fulfills this goal, we asked the students if they would revise their communication and management strategies based on the experience of the contest. Given five answer choices, ranging from *"Yes"*, *"Likely"*, *"Maybe"* and *"No"*, more than 60% of the students answered *"Yes"*, *"Likely"*, or *"Maybe"*. This suggests that the contest has an important impact on how students see communication and management in the project. In addition to questions with predefined answers, we also asked students to provide information about what they learned from the contest. In total, 67% of all students provided such optional feedback. Of those, more than

Table 2: **Students' ratings of the contest. Each question was answered with a value between 1 and 5 where 1 represents *"Not at all"* and 5 represents *"Very much"*. The table reports mean, median (μ) and standard deviation (σ).**

Question	Mean	μ	σ
Q5: The contest demonstrates challenges of communication	3.9	4	0.9
Q6: The contest shows importance of group management	3.9	4	1.0
Q7: The contest was enjoyable	4.0	4	1.1

42% mentioned that the contest was useful to learn about group management and 40% indicated that the exercise is helpful to understand the challenges of communication in distributed groups. Samples of the students' feedback in this respect, are the following:

- *It was very clear the need of a central figure that could assign the responsibilities, specially for the tasks that were shared among many of us.*
- *It is important that someone leads the group with a common idea, that everyone has a job to do, and if someone does not finish his/her work all the project remains incomplete.*
- *Communication can be pretty chaotic under the time pressure; an agreement about tools for communication is necessary; project management is pretty important for any project.*
- *I learnt that communication is very important and it is fundamental having someone who has the control of what everyone is doing.*

4.4. Impact for the DOSE Project

Impact on Group Organization. One of the goals of the contest is to demonstrate the importance of having a project manager in a distributed project. In previous editions of DOSE, it has been difficult to convince students to define a project manager: either they do not define a project manager, or they realise at the end of the project that they need it. Figure 1b shows the findings about changing group communication and project manager after the contest. The figure shows that 66% of the students have answers "yes", "likely, or "maybe" for revising their communication, and 70% have answers "yes", "likely, or "maybe" for assigning a project manager. These values are higher compared to the previous week, where most of the groups reported that they do not need a project manager.

The discussed contest was organized in the 2010 and 2012 editions of the DOSE course. In 2010, the contest was mandatory, and all the groups participated in it. The results obtained after the contest were very positive. We analyzed how the groups reorganized their internal structure: groups that did not have a project manager, decided to assign one and extensively discussed who would perform this role. Other groups reported that, after the contest, they decided to change the project manager because they thought the current project manager did not have the right skills for it, or he did not like the task. All groups reported that they understood the importance of the project manager, and they tried to assign the best match for that role.

In the 2012 edition of DOSE, the contest was optional. As we discuss below, this allowed us to measure how groups that participated in the contest performed in terms of group effectiveness,

compared to those that did not participate. In 2012, besides group reorganizations similar to those of 2010, students also reported a re-structuring in the way they organized meetings, and in the mechanisms for deciding how to start working on new assignments. When a new project assignment was published by the instructors, some groups that participated in the contest first organized a group meeting to agree in the scope of each team subcomponent, as opposed to teams directly starting with the new assignment as soon as they received it from the instructors.

Project Outcome. In 2012, 14 groups participated in the contest while 7 groups have not participated. This enabled us to measure the impact of the contest by comparing the performance of the groups that participated in the contest with the performance of those groups that did not participate in it. As an overall measure of the impact, we defined and measured *project success*. We observed that 93% of the groups that participated in the contest successfully implemented the course project, producing an application that integrated the subcomponents of the different teams. On the other hand, only 43% of the groups that did not participate in the contest successfully implemented the course project.

In order to measure project success, we used the demos provided by the students of the corresponding developed applications. The degree of success was defined by assigning one of the following values to each of the developed applications: 1: *project failed*. No demo was provided and the application does not compile or run; 2: *partial success*. A demo was provided, but the subcomponents were not successfully integrated; 3-5: *full success*. A demo was provided and all the subcomponents are successfully integrated; the value 3 was assigned to simpler projects; 4 to good projects and 5 to very good projects.

Table 3-column 2012 shows the project outcome classified by groups participating or not in the contest. This table reports minimum ($\mathbf{m}$), median (μ), maximum ($\mathbf{M}$), mean, and standard deviation (σ). As it can be appreciated in this table, a better outcome is observed for the groups that participated in the contest, compared to those groups that did not participate in it. We performed a *t-test* in order to check if the differences are statistically significant. We applied a significance level of $\alpha = 0.05$, i.e., an observed difference is considered significant with a probability of 95% if the corresponding p-value is less than 0.05. The difference in quality is statistically significant with a p-value of 0.0001.

Table 3: **Outcome of the 2012 as well as 2011-2012 projects classified in the groups that participated in the contest and the groups that did not participate. For each measure, the table reports minimum ($\mathbf{m}$), median (μ), maximum ($\mathbf{M}$), mean, and standard deviation (σ).**

	2012					2011 and 2012				
	$\mathbf{m}$	μ	$\mathbf{M}$	Mean	σ	$\mathbf{m}$	μ	$\mathbf{M}$	Mean	σ
Groups participating in the contest	1	4	5	3.57	1.02	1	4	5	3.57	1.02
Groups not participating in the contest	1	1	3	1.57	0.79	1	3	4	2.56	1.09

A threat to validity in comparing the groups that participated in the contest with those that did not is that, since the contest was optional, the groups not participating in the contest might had been less motivated than the other groups, and thus this initial motivation was what actually affected the final outcome, as opposed to the contest. To address this issue, we extended the above analysis including data from DOSE 2011, in which the contest was not organized (and thus groups were not segregated by contest participation). The second column in Table 3 shows the project outcome of the 2011 and 2012 projects. It includes new data points for the groups not participating in the contest, and therefore has the same values for the groups participating as in the column 2012. The

outcome of the projects participating in the contest is still better than the ones not participating. Applying the *t-test* shows that the differences are statistically significant with a p-value of 0.007.

5. Related Work

This paper focuses on a motivational contest whose goal is to prepare the students for the challenges of distributed software development (GSD). As far as we know, this is the first globally distributed contest used to prepare students of GSD courses. This contest is carried out in the context of a DOSE course, teaching GSD. There exist several courses similar to DOSE, teaching globally distributed software development (GSD), whose results have been reported previously [11, 12, 10, 1]. These include the works reported in two editions of the CTGDSD workshop [4]. Let us describe some of these courses, and compare their settings with the setting used in DOSE.

Gotel et al. [10] report some lessons learned from the development of a project across three globally distributed educational institutions in the United States, India and Cambodia. They present the problems faced in the projects including cultural aspects, project planning and communication (with a twelve hours time difference). In the first four years [19], they implemented the projects using a traditional waterfall model, while in the last two years, they applied an agile process using Scrum. In DOSE, we have always employed a waterfall process model, with a strong emphasis on API design using design by contract and emphasis on good and frequent communication. However, some more agile notions are being introduced, e.g., by incorporating some test driven development ideas to strengthen API design.

A similar teaching experience is presented by Bosnic et al. [1] involving distributed software engineering in collaboration with Croatia and Sweden. The course has been taught for several years facing challenges such as motivation of students, and organizational issues. Damian et al. in [5, 6] report on the teaching experience developing software requirements specifications in geographically distributed software development with three universities (located in Canada, Australia, and Italy), focusing on the time zone aspect and the cultural differences. Other examples of GSD courses are: Bruegge et al. [2] in collaboration with universities in the United States and Germany; Richarson et al. [18] in collaboration with the United States, Germany, Ireland, and India; and recent GSD courses such as [9, 20, 11].

Our experience, in particular the last three editions, involved several universities with a more heterogeneous settings and a wide rage of time zones: some of the groups had a twelve-hour time difference, others had a seven-hour time difference, and others only two-hour time difference. Another main difference, besides the number of universities participating in the course, is how we organize the project. It is organized as *one* main system consisting of several subcomponents, with each development group contributing a subcomponent to the whole development. Using this setting, we experience more issues of distributed software development, and require better coordination, project management, and tool support to be applied.

Our experiences with DOSE have evolved very quickly. Nordio et al. [15] described our first experiences in distributed software development. Since then, various ideas were gradually incorporated, this happening in parallel with our project settings becoming more challenging. This gradual evolution of our courses and projects are described in some of our previous and related work [14].

6. Conclusion

We presented an exercise used in a globally distributed project, whose purpose is to help students understand the importance of effective communication and project management. This exercise, or-

ganized as a globally distributed contest, shows that students face various critical challenges of communication and project management in a distributed project. The contest's assignments were designed to emphasize the importance of communication and management, and thus required solving very simple, not programming related, tasks. These assignments included redundant and as well as non required tasks, and some of them referred to data that was split into different members of the group, making planning, management and communication essential for a successful completion of the tasks. Overall, we found that students were in general surprised by the simplicity of the tasks and how ineffective they were in solving them, concluding that the problem was indeed in communication, coordination and planning. The results obtained from the contest suggest that it has a high impact on how the students plan and organize their projects in the future, and in particular within the DOSE course.

References

[1] I. Bosnic, I. Cavrak, M. Zagar, R. Land, and I. Crnkovic. Customers' Role in Teaching Distributed Software Development. In *CSEE&T*, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[2] B. Bruegge, A. Dutoit, R. Kobylinski, and G. Teubner. Transatlantic project courses in a university environment. In *APSEC 2000*. IEEE Computer Society, 2000.

[3] I. Crnković, I. Bosnić, and M. Žagar. Ten tips to succeed in global software engineering education. In *ICSE 2012*. IEEE Press, 2012.

[4] CTGDSD workshop. `http://www.cpathi18n.org/dsd/ctgdsd3/index.php`.

[5] D. Damian, A. Hadwin, and B. Al-Ani. Instructional design and assessment strategies for teaching global software development: a framework. In *Proceedings of ICSE*, 2006.

[6] D. Damian, F. Lanubile, and T. Mallardo. Investigating IBIS in a Distributed Educational Environment: the Design of a Case Study. In *Proceedings of DiSD 2005*, 2005.

[7] J. A. Espinosa, N. Nan, and E. Carmel. Do Gradations of Time Zone Separation Make a Difference in Performance? A First Laboratory Study. In *ICGSE*, pages 12–22. IEEE, 2007.

[8] H.-C. Estler, M. Nordio, C. A. Furia, B. Meyer, and J. Schneider. Agile vs. structured distributed software development: A case study. *Empirical Software Engineering*, pages 1–28, August 2013.

[9] P. Gloor, M. Paasivaara, C. Lassenius, D. Schoder, K. Fischbach, and C. Miller. Teaching a global project course: experiences and lessons learned. In *CTGDSD '11*, 2011.

[10] O. Gotel, V. Kulkarni, L. Neak, C. Scharff, and S. Seng. Introducing Global Supply Chains into Software Engineering Education. In *SEAFOOD*, 2007.

[11] M. Hawthorne and D. Perry. Software engineering education in the era of outsourcing, distributed development, and open source software: challenges and opportunities. In *ICSE*, 2005.

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

[13] M. Nordio, H.-C. Estler, B. Meyer, J. Tschannen, C. Ghezzi, and E. D. Nitto. How do distribution and time zones affect software development? a case study on communication. In *ICGSE*, 2011.

[14] M. Nordio, C. Ghezzi, B. Meyer, E. D. Nitto, G. Tamburrelli, J. Tschannen, N. Aguirre, and V. Kulkarni. Teaching software engineering using globally distributed projects: the dose course. In *CTGDSD '11*, 2011.

[15] M. Nordio, R. Mitin, and B. Meyer. Advanced hands-on training for distributed and outsourced software engineering. In *ICSE 2010*, 2010.

[16] M. Nordio, R. Mitin, B. Meyer, C. Ghezzi, E. D. Nitto, and G. Tamburelli. The Role of Contracts in Distributed Development. In *SEAFOOD*, 2009.

[17] R. Ocker, M. B. Rosson, D. Kracaw, and S. R. Hiltz. Training students to work effectively in partially distributed teams. *Trans. Comput. Educ.*, 9(1):6:1–6:24, Mar. 2009.

[18] I. Richardson, A. E. Milewski, N. Mullick, and P. Keil. Distributed development: an education perspective on the global studio project. In *ICSE*, 2006.

[19] C. Scharff. An evolving collaborative model of working in students global software development projects. In *CTGDSD*, 2011.

[20] E. Stroulia, K. Bauer, M. Craig, K. Reid, and G. Wilson. Teaching distributed software engineering with ucosp: the undergraduate capstone open-source project. In *CTGDSD '11*, 2011.