Distributed Software Development Course: Students' and Teachers' Perspectives

Juraj Feljan, Ivica Crnković
Mälardalen University
School of Innovation, Design and Engineering
Software Engineering Division, Västerås, Sweden
{juraj.feljan, ivica.crnkovic}@mdh.se

Ivana Bosnić, Marin Orlić, Mario Žagar
University of Zagreb
Faculty of Electrical Engineering and Computing
Zagreb, Croatia
{ivana.bosnic, marin.orlic, mario.zagar}@fer.hr

***Abstract*— Students and teachers do not necessarily have the same understanding of a course – of the purpose, the objective, and in particular of the course elements – the way the course is performed, the examination procedure, and similar. In distributed-development courses, in which students and teachers are dispersed over different locations, this difference can be larger than in "ordinary" courses, but also less visible, due to limited communication. In this paper we discuss these different perspectives, their rationales, possible consequences on the course performance and on the result, as well as lessons learned from students' feedback.**

***Keywords*-Distributed software development; Education; Global software engineering**

I. Motivating Example

As a standard procedure in our Distributed Software Development (DSD) course [1], at the end of the course this year [2] we have asked the students to fill in a questionnaire. We received a very interesting and long answer from one of the students, containing a combination of suggestions for improvements and better alternatives for different elements in the course. This comprehensive comment initiated an exchange of discussion e-mails, and finally a proposal from our side to the student to read and comment on two papers about the DSD course challenges [3][4]. The student's answer was: "*I have read the papers and I have to admit I was very surprised by them. I ... realized ... that most of what I have said in my feedback you had already knew but for various reasons you were not able to address. In general, I think I agree with the "Ten tips", in fact now having read it I realize that many of the things I found unnecessary during the course actually had a good reason behind them. .... Although, I am sure it's not as simple as I think (but it should be*!)".

The last sentence in the student's answer raises several questions:

a) how to design a DSD-type course to be as simple and straightforward as it can be, and how to avoid or overcome unnecessary complexity,

b) what are the differences in students' and teachers' perspectives that are specific for DSD-type courses,

c) can such differences influence (negatively or positively) the course performance and results?

In this paper we discuss these questions through the students' feedback and the teachers' experience. In particular, we discuss the experience of a coauthor that was first a student and then became a member of the teaching staff. We present different course elements (the development processes, the student cooperation, the communication between the students and teachers, the examination procedure) interesting to look at from two different angles (students and teachers). We discuss the differences in student-teacher perspectives and pose several questions related to the lessons learned and best practices.

DSD course has been carried out for nine years between two sides, Mälardalen University in Sweden and University of Zagreb, Croatia, with occasional participation from University of Paderborn, Germany. The course is designed as a combination of lectures, guest presentations and distributed projects. Student projects are the largest part of the course. The projects are shared and the students work together like in a local project. The examination elements are common, as are the students' grades. Projects offered to students are sized for 6 – 8 team members (3 – 4 per site), carried out over the period of one semester, 16 – 18 weeks. One of the students acts as a project leader, and one student on the other site acts as a local team leader. One of the teachers plays the role of the project supervisor. Project customers can be either teaching staff members, or external customers, such as companies or project proponents in various SE contests.

The rest of the paper is organized as follows. Section 2 lists students' comments, observed issues, proposals for improvements from the course evaluation report, as well as the teachers' rationale based on the current experience. Section 3 compares the two perspectives from the viewpoint of one of the coauthors who was involved in the course both as a student and a teaching staff member. Section 4 concludes the paper.

II. The Students' Feedback

During the course execution we gather students' data from several sources: (i) *Initial questionnaire* – students provide a short overview of their interests (professional and hobbies), experience in software design and development; (ii) *Periodic polling* – once a week, students express their current feelings in a "How happy am I?" poll; (iii) *Final questionnaire* – students are required to fill-in an exhaustive final questionnaire, which reflects their experiences and thoughts on distributed work in detail; (iv) *Course evaluation* – an internal course evaluation, which is anonymous and optional. We encourage the students to fill it, as they can help us see the possible problems in the course. By answering 15 questions of the evaluation they discuss the topics such as: concept of lectures and projects, cooperation between sites, student workload, project support, and course administration. All these elements are both numerically graded and commented. We especially advise students to give us their comments in a free-text form, as this reveals more information about the topics than numeric values can.

During 9 years of course delivery, 255 students have evaluated the course. The complete questionnaires from the period 2003-2011 can be seen at [5]. Here we present the results from answers to two questions: “As a whole the course was”, and “The course has fulfilled my expectations”, answered in the range of 1-5 (1 meaning “not at all”, and 5 – “completely”). Table 1 shows the answers distribution per year.

TABLE 1. RESULTS FOR TWO ANSWERS FROM THE QUESTIONNAIRE
Statement 1: As a whole the course was (1:bad – 5:excellent)
Statement 2: The course has fulfilled my expectations (1:no – 5:yes)

Year	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**Average**
Students #	21	52	44	26	11	28	36	15	22	**28.33**
Statement 1	4.71	4.58	4.32	4.08	4.36	4.14	4.67	4.53	4.23	**4.40**
Statement 2	4.48	4.29	4.02	3.88	4.27	4.11	4.19	4.18	4.09	**4.17**

From the answers we can see that students are continuously pleased with the course, and their expectations are fulfilled (though slightly less than their satisfaction with the course).

However, here we will focus on the question “What can be improved in the course?”. From this question we have obtained 235 improvement responses. From these responses, 19 have stated that they have no improvements to suggest. We have grouped the remaining 216 responses in 8 categories, according to their main theme, in a two-pass process. In the first pass, the possible categories have been identified and roughly grouped. In the second pass, a more precise mapping was made. Some responses could be grouped in more than one category; for the calculations, they have been included in the strongest resembled category, but they can be addressed in different categories in the improvements description. The list of 8 improvement proposal categories, which contain more than 10 suggestions, is given in Table 2.

Students’ improvement proposals are presented as follows, including our rationale in italic formatting.

TABLE 2. THE MAIN CATEGORIES OF THE SUGGESTIONS, WITH THE NUMBER OF STUDENT PROPOSALS INCLUDED

Proposal category	**#**
technical resources	32
knowledge level	31
project selection and assigment	31
lectures	29
course organization	27
workload	20
course advising	15
grading	14

A. Technical Resources

Videoconferencing is an important part of a flawless DSD course, as we meet together for lectures and project presentations every few weeks. Some technical glitches occur from time to time, mostly connection issues or temporary low bandwidth. Students especially emphasized that the sound quality heavily influences the remote communication. In addition, students often asked us to provide additional meeting rooms with equipment for distance communication, where they could meet with their remote team members.

Student projects that include a server-side component are hosted on virtual machines provided by the course staff. Source code versioning is done using *SVN*, which is also hosted on our virtual server. As we could not guarantee non-stop uptime of the machines nor backup service, and server crashes sometimes occur, students stated that server infrastructure should be improved. Several students proposed to substitute *SVN* with more modern versioning tools, like *Mercurial* or *Git*. Students would have also appreciated more technical support regarding their virtual machines, as each team gets a virtual machine and is responsible for its administration, but most students do not have enough knowledge to work on this.

The project Web page is an important communication means. The students upload all their work there and have some communication possibilities. Since the Web page is managed by the Campus Content Management System developed at University of Zagreb, the students had to get used to the way of publishing news or documents, so some of them suggested improving the Web interface.

To reduce technical problems with the communication equipment, the staff on both sites prepares in advance and tests the equipment early before the lectures start. However, there are some events beyond our control, so we prepare additional backup plans (other tools to use, or even partial local lecture while the problem is fixed). We emphasize strong flexibility needed to solve server problems on the spot, and help students when crashes occur. To address the Web site proposals, we provided a CMS user manual in English, and initial lectures include step-by-step explanations of the tasks that need to be done at the Web site in the first weeks. We can conclude that the technical level still does not reach the quality one would expect in a seamless communication.

B. Knowledge Level

A lot of students were concerned with misbalance in knowledge levels of students enrolled to the course, ranging from students who have poor knowledge in programming and basic software engineering disciplines, to ones who have specific knowledge not required for their particular project. This made more knowledgeable students seriously demotivated, and caused different team problems, as low-knowledge students often would not be eager to learn. They proposed that we make an effort to seriously warn students about the course requirements and organization, as well as to make some kind of pre-course test and evaluation.

Besides lacking software engineering skills, low English language proficiency of some participants was mentioned, which made it hard to communicate, discuss and work together.

These knowledge-related problems are our strong concern, as they highly influence project work. A part of these issues is related to the considerable heterogeneity of students, coming from a number of countries and having studied in programs of various qualities. A part of solution can be found in having a short pre-course test, which briefly evaluates their programming skills, as well as English language skills. The administrative constraints do not allow us to reject a student who wants to enroll, but we can advise against enrolling, depending on the test results. 2011/2012 was the first year where we gave this test, and received only one low-knowledge result. We also occasionally held lectures in specialized areas which several students needed (for example lectures in UML).

C. Project Selection and Assigment

Students asked for more project proposal options that they could choose from, more freedom in choosing the projects, and in choosing the technologies. Also, the students argued that some projects are more demanding than the others. Students enjoyed our industry cooperation, and would have liked more projects with real customers involved. SCORE student competition participation [5] was seen by some students as unnecessary workload, while some others proposed that not all team members should be involved in SCORE, if they don't want it – to keep the team motivation higher.

Regarding the project assignment process, students would have liked that the staff focuses more on the skills required per project, and map students to projects better. There was a suggestion that we should interview students one-by-one before assigning them to a project.

An interesting objection regarding cultural issues occurred – at Swedish side, in a strong multicultural environment, there should be a balance of nations involved in a team; otherwise a kind of favoritism can occur.

We could propose more projects than we would run in a particular course instance, but this would also bring problems such as harder decision-making on the final list of projects, and misbalance in staff's workload. We should make an effort to balance the project difficulty, whenever possible, or at least balance the main requirements complexity of the projects. The project assignment process has changed over the years, and now includes both a poll (which projects would students like to work on) and a self-evaluation of technologies used by a student. It may seem that this self-evaluation goes in the way of "choose the project freely" argument, but we try to optimize the students' satisfaction, by assigning them to the projects which they like as much as possible. Self-evaluation is just an additional tool that we use to balance the student teams. We support the idea of more industry customers, but one should keep in mind various aspects of such cooperation, advantages and challenges, as described in [6].

D. Lectures

We received diverse comments regarding the lectures. Some comments were focused on reducing the number of lectures, especially the ones which are basic, or not necessary in relation to the projects. On the other hand, there was a demand for more lectures regarding technologies that could be used in the projects. Guest lectures were especially welcome, with guests from industry, experienced in global software development, who can discuss their real-world experiences. Having a lecture from former DSD students who could speak about their course experiences was another improvement proposal. Comments like "a little more life in lectures wouldn't hurt" were also present several times.

Another thing to note was the comments about cultural differences. Some students felt we overemphasize cultural issues, which don't seem to be important to them. Other students would have liked to hear more about them, although we already give a lecture about these.

A proposal for more guest lectures is commendable. As an addition to the lectures, a former DSD student was invited this year to speak about his comparison between DSD experience and real-world distributed project. We also feel the need to give more lectures about project management, and the importance of professional project documentation. Having experience with past course instances, we do feel the need to address cultural differences, although students are unaware of these at times. Our experience has shown that the differences are not so visible, until there is a problem in the team: in that case, cultural issues become a strong, highly visible factor.

E. Course Organization

This was a group of diverse improvement proposals. There was an always-present wish for face-to-face contact, which unfortunately could not be realized, due to financial reasons. But, to make the first contact easier, students asked for some ice-breaking sessions, as well as proposed to have additional innovative and fun moments during the course, to break the "serious" course atmosphere.

Evaluation in 2009/2010 resulted in several comments about the number of students and presentations. Due to a larger number of teams and presentations and short amount of time for each presentation, the students felt their work was not valued appropriately. Additionally, they demanded stricter enforcing of presentation time limits.

Students also argued about too many deadlines, especially the initial ones, which are quite dense. Finally, some students were not satisfied with the high amount of polls and questionnaires required during the course.

Ice-breaking is done through several activities: in the first lecture there is a fun quiz about famous Croatian and Swedish people, and geographical locations of teams involved. Students are also asked to introduce themselves in a number of ways: posting images, describing their interests and hobbies, etc. The year with the highest enrollment was 2009/2010. It was a struggle to give everybody the attention required, fit all activities into course hours, especially *during the presentation slots. However, we agree that presentation durations should have been enforced stricter. Reorganization of deadlines would be possible, however, students do not realize the "hidden" reason – we have learnt that if the students were not forced to start communicating hard from the beginning, things would not go smooth during the project. Regarding tight deadlines, the staff should take care of ensuring these deadlines are met with decent work done, instead of just fulfilling the deadline by submitting required documentation, which would be rewritten from scratch afterwards.*

F. Workload

Overall, students often complained about the workload in the course. Sometimes this was expressed as a wish for less demanding projects, but more often it was a wish for "more time", or "more ECTS" points, as students in general like to work on challenging projects.

Currently, the course is worth 7.5 to 8 ECTS, which can be translated to 200 - 240 hours of work total, depending on the university. Some students propose the prioritization of requirements to make the project easier. During the requirements gathering phase, students formalize the final list of requirements in agreement with their supervisor. Mostly due to their inexperience, or their wish to show their skills and motivation, they often make demanding promises – so we advise them to start small, while thinking of possible features that can be added later on.

G. Course Advising

A number of students stated they would have liked more support and advising from their supervisors. The proposals ranged from more support in the beginning (including more technology advising) so they could follow a good path, to the request for a greater support during the course, supervising the progress of the whole team, but also each member individually, done by one-on-one interviews. An interesting proposal was having a supervisor on each of the sites, as it can be hard to supervise the remote site.

One of the course goals is to prepare students for real world work, where individual decisions need to be made. We do not want to lead the students too much. However, we need to work on closer support and monitoring, especially with regards to students proposals about grading.

H. Grading

There were several improvements the students proposed concerning grading. They felt the staff should make a more thorough analysis and testing of the final product in the end (e.g. a student asked to focus on actual finishing of the project, instead of grading an "illusion" that it works), and that the grades should be more influenced by the product quality. Students who gave their best should have been better awarded, with a greater distinction to the ones who invested less effort. There was a feeling that we put too much accent on working hours (reported by the students), instead of trying to determine the actual results of each student.

The grading system we have is very detailed, with more than 20 different criteria that refer to the final results, the quality of coding, the documentation quality, the project organization and team work. The weakness of our grading system is that it is difficult to know how much each student contributed (which is usually not a problem in good teams, but in teams with weaker students and weaker results). Another problem (which we realized later) was that the criteria were not explained clearly enough. Finally the grading is given at the end of the course, so until the end of the course the students do not know their potential grade.

I. Other Issues

There were a few other issues referred to in the evaluations, not significantly represented. A point made by some students most of the years was that project requirements were not clear, and they did not understand what is required of them. They were also not happy with the amount of documentation required in the project, especially if they felt the documentation is not useful. There were also some team issues, which were not directly related to course improvements. They described mainly the need for greater responsibility and motivation of each student in a team.

III. The Two-perspectives Experience

One of the coauthors participated in the course first as a student and then four years later as a member of the teaching staff. Here we present several aspects of the course observed by the same person, but from different perspectives – the student and the teacher perspectives.

Motivation for choosing the course. The teachers' motivation for giving a DSD course is quite clear – to provide the students with insights from this increasingly common way of working in the industry. However, as a student, the coauthor opted for the course mostly thanks to its reputation of being challenging, useful, fun and different from other courses, rather than for getting educated in distributed development. Influenced on one side by the curriculum which primarily educated programmers, not software engineers, and on the other side by an individual lack of experience and knowledge on how development is done in industry, the coauthor was not drawn to the course to gain deeper insight into distributed development, but to increase his programming skills and to meet students from another university. It can be said that as a student, the coauthor did not fully understand the intention behind the course. It is therefore important that the staff clearly conveys to the students the motivation for the DSD course.

Work motivation. The coauthor supervised one project team, and felt that his involvement would set an example to the students, and thus affect their performance. He additionally had an impression that when the team's performance was reviewed, his performance as a supervisor was also being reviewed. These two aspects were sufficient to keep the work motivation high throughout the course.

As a student, the coauthor was mostly motivated by the love for programming, and by a general team desire for the project to succeed. Another important motivating aspect was the good work done by (most of) the team on the remote site. Two aspects that had a negative effect on the motivation were the following: there was one team member who repeatedly failed to deliver what he had promised; the supervisor could have shown more interest in how the project was advancing. In general, the coauthor had no issues of maintaining the motivation high, either as a student or as a teacher. However, the sources of the motivation were obviously different in the two cases.

Communication. As a student, the coauthor had no problems in communicating his ideas to the other team members. However, later on in the course he realized that he had been the source of a communication issue. A team member on the remote site had documented an API used in the project. The coauthor, being an inexperienced student, instead of referring to the documentation, kept asking questions about the API directly via e-mail or instant messaging. This created unnecessary communication overhead and frustration for the remote team members.

As a teacher, the coauthor mostly communicated with the team via the project leader. They both come from the same country, and thus speak the same language and share a similar working culture which minimized the potential for misunderstandings. The communication with the rest of the team did not yield problems either. On the other hand, the students reported some communication issues within the team, mostly due to varying English proficiency levels.

Perception of project work. As teachers, we try to give students the complete picture behind project work in a distributed team, covering all of the important aspects, such as management of a distributed team, following a certain development process, writing good documentation, programming etc. Nevertheless, it can be easy for the students to get blinded by their particular role, thus not getting a holistic view. As a student, the coauthor was more a programmer than a software engineer, and so was his perception of his role in the project and the project work. Apart from initially participating in requirements gathering, he was mostly focused on programming tasks, and did not take much interest in other aspects of the project work. Only at the end of the course it became apparent to the coauthor that he should have gotten more involved, despite the fact that he enjoyed his programmer role. His biggest regret was not participating in the software design phase.

Project requirements. We deliberatly deliberately gave students vague requirements, in an effort to accurately simulate the real world, and to train students in requirements gathering and analysis. However, the students are often not aware of this. This is visible from the questionnaires, where the students frequently complain on not getting clear requirements, and it was the case for the coauthor. He was disappointed with the customers because "they themselves were not sure what they wanted from the product". As an inexperienced student, he did not know that this is often the case in the real world. So instead of a valuable lesson, the effect of vague requirements was frustration. Therefore, as teachers we should explain to the students the rationale behind vague requirements.

Technical aspects. Regarding the technical aspects, the coauthor has kept a consistent view both as a student and as a teacher – various tools are crucial to alleviate the distance factor. Most of the tools that the coauthor used during project work, he still finds relevant as a teacher: communication tools (e-mail, instant messaging, video-conferencing, forums), code sharing/versioning tools, project management and bug tracking tools.

From having had both a student and teacher perspective comes the following tip: as teachers we should remember that for most students this is often the first encounter with working in a bigger project group, and it is easy for them to get overwhelmed and lose focus of what we try to convey in the course. We should therefore not be reluctant to keep stressing on the important points mentioned in this paper – however clear and trivial they might seem to us, the students often have a different perspective.

IV. Conclusion

We listed a set of issues that we collected from the students' evaluation reports. Some of the issues are typical for any type of course (like involvement of the supervisors, boring lectures), some of them are related to the project-type courses (like team issues, project process issues), some of them are the results of the diversity of students (cultural differences, different skills) and the differences between sites (the students and – to some extent – the teaching staff base their assumptions of the distributed environment on local experiences), and finally some are related to technical limitations in distance communication. While for most of the issues it is difficult to find the distance as the exclusive source of the problems, we can state that in general, due to the distance and different traditions and local rules, meeting the challenges is more difficult, more efforts are required, and the results might not be as good as expected. A continuous questioning of the procedures, and a continuous emphasis on communication is a way to be more aware of possible problems.

References

[1] I. Bosnić, I. Čavrak, M. Orlić, M. Žagar, and I. Crnković, "Avoiding Scylla and Charybdis in Distributed Software Development course," in *Proceedings of the 2011 community building workshop on Collaborative teaching of globally distributed software development*, 2011, pp. 26–30.

[2] "DSD course, the official site." [Online]. Available: http://www.fer.hr/rasip/dsd/. [Accessed: Feb12, 2012].

[3] I. Crnković, I. Bosnić, M. Žagar, Ten Tips to Succeed in Global Software Engineering Education, in *Proceedings of International Conference on Software Engineering, Education track, 2012 (to be published)*

[4] I. Čavrak, M. Orlić, I. Crnković, Collaboration Patterns in Distributed Software Development Projects, in *Proceedings of International Conference on Software Engineering, Education track, 2012 (to be published)*

[5] DSD course, Web Page on MDH site, [Online], Available: http://www.idt.mdh.se/kurser/cd5610/2011/ [Accessed: Feb12, 2012].

[6] I. Bosnić, I. Čavrak, M. Žagar, R. Land, and I. Crnković, "Customers' Role in Teaching Distributed Software Development," in *CSEET '10 Proceedings of the 23rd IEEE Conference on Software Engineering Education and Training*, 2010, pp. 73-80