Work in Progress - Benefits of Cooperative Learning in a Multidisciplinary Course

Carlos Álvarez, David López, Daniel Jiménez-González and Javier Alonso
Technical University of Catalonia, calvarez@ac.upc.edu, david@ac.upc.edu, djimenez@ac.upc.edu, alonso@ac.upc.edu

Abstract - The implementation of multidisciplinary courses presents both a challenging task and a powerful tool due to the students’ different knowledge areas’ contributions. In this paper we present the experience had during a technical course where students from different programs had to cooperate to develop both technical and professional skills. The basis of our proposal was to make students work in pairs formed by students from different programs. They had to use cooperative work and peer-to-peer learning that would allow them to cover the course’s technical and non technical objectives. Our first results have shown that all students have achieved good learning results despite their different backgrounds.

Index Terms - Cooperative Learning, Multidisciplinary Teams, Open Source courses.

 MOTIVATION

The ability to function in a multidisciplinary team is one of ABET’s professional skills [1]. Multidisciplinary courses expose students to a wide variety of ideas and methods that help them learn how to apply their knowledge in different ways and combine ideas while also using the tools of different disciplines. However, most multidisciplinary courses only take place in the first year [2][3], when basic concepts are taught. After the first year, technical courses involving students from different disciplines are perceived as problematic because they don’t share a common knowledge. For this reason, the professor usually has two options: either having students with poor learning results or lowering some of the technical objectives.

At the Technical University of Catalonia’s Barcelona School of Informatics, a course in Free Software (Open Source) [4] is offered. Due to lab limitations, only 24 students may enroll the course. Twelve of these places are offered to Computer Science Engineering students, and 12 more to other students from the rest of our University (all coming from technical study courses such as architecture, civil engineering, communications engineering and mathematical science). The variety of topics in this course (from social subjects to advanced systems administration) is hard to manage.

The first edition of the course though was discouraging because very few non-computer science students achieved the technical objectives. Besides, computer science students didn’t cope as well as the rest of the students in terms of course core skills like oral presentations or teamwork.

To overcome that situation, we’ve made students work in pairs formed by a computer science student and a non-computer science student. Though we know this kind of grouping is against their will because they tend to pair up according to knowledge and/or affinity levels, we encourage them to cooperate during their practical lessons, teach each other concepts and prepare and present one project together.

We think cooperative work and peer-to-peer learning necessarily trigger a synergy between them because they’re teaching and learning from their peers at the same time while carrying out a project with a person with a different way of thinking and knowledge, which results in their personal growth and development of skills such as teamwork and communication. This approach to learning allows for the course to cover all topics and achieve its original technical and non-technical objectives with good results.

In the next sections we will briefly explain the characteristics of our course, the preliminary results of our study and the evaluation plan we will follow.

COURSE ORGANIZATION

This course has 3 contact-time hours per week over a period of 15 weeks. Every week, 2 hours are devoted to laboratory sessions, which include installing a Linux distribution, process management, programming shell scripts, network configuration, security, and network services (Apache, SSH, FTP, NFS, NIS and Samba). There is 1 hour lecture per week. Some lectures address the free software philosophy, its history and legal, social and ethic implications. The rest are dedicated to the public presentations of the projects carried out by the students, which can be highly technical (e.g. “How to install a local area network”) or related to social issues (e.g. “Computers and sustainability”). The final mark depends on the lab mark (40%), the final exam mark (40%) and the project public defense mark (20%) results.

PRELIMINARY RESULTS

This course has run over six terms. Figure 1 shows the final exam marks obtained by Computer Science (CS) and non Computer Science (non-CS) students during five editions of the course. Marks range from 0 to 10, where 5 is the pass mark. Results obtained by non-CS students in the first edition were disappointing as an important subset of these students did not pass the exam (but nearly all of them passed the course thanks to the lab and project public defense marks). However, non-CS students’ results improved when we applied our methodology.

October 22 – 25, 2008, Saratoga Springs, NY
On the other hand, CS students’ marks remained very similar throughout the terms, showing that the technical objectives were not reduced.

Although in terms of qualifications the results show that the objectives are achieved, we have gathered different data to evaluate our proposal. Last term (fall 2007) students were surveyed about our methodology. We asked students to rank the three assertions showed in Table 1. We used a 4-level choice scale to make students rank their most to their least preferred options from 1 (completely disagree) to 4 (completely agree). In this table, we show the mean of the answers of three different groups of students: CS students whose partner was another CS student (CS partner), CS students whose partner was a non-CS student (Non-CS partner) and finally non-CS students (that always work with a computer science student). (We have CS students working in pairs because, unfortunately, the places offered to other students are not always filled; in those cases, we use CS-CS pairs as a control group.) Results show that every student that worked with a partner from other program considered it a positive experience: while CS students who worked with another CS student considered that his/her partner scarcely affected his/her final mark, students with another program partner considered that his/her partner helped them to master the subject and to obtain a better grade. Surprisingly, the best results were for CS students, while our intuition told us quite the opposite.

We wish to thank our students for their enthusiasm and participation in our experiments. We also thank Ceila Hidalgo for the proof-reading work. This work has been supported by the Spanish Ministry of Education and Science (project TIN2007-60625), and the Barcelona School of Informatics (Facultat d’Informàtica de Barcelona – FIB).