CODILA: A Collaborative and Distributed Learning Activity Applied to Software Engineering Courses in Latin American Universities

César A. Collazos, Sergio F. Ochoa, Sergio Zapata, Fáber D. Giraldo, M. Inés Lund, Laura Aballay, Gisela Torres de Clunie

Abstract— **Software engineering education has been recognized as an important challenge in computer science undergraduate programs. Instruction in such area requires not only to deliver theoretical knowledge, but also to perform practical experiences that allow students to assimilate and apply such knowledge. This paper introduces a computer-supported Collaborative and Distributed Learning Activity (CODILA) that has been used to allow undergraduate students to acquire not only particular knowledge on software engineering, but also specific competencies or skills to collaborate in a distributed development team. The article also presents two experiences in which CODILA was applied. Such experiences involved students and instructors from five Latin American Universities. The obtained results were satisfactory and indicate this collaborative learning activity could be appropriate to address the instructional process in other disciplines.**

Index Terms— **Computer-Supported Collaborative Learning Activity, Distributed Collaboration, Software Engineering Education, Collaboration Skills.**

I. INTRODUCTION

SOFTWARE engineering is a highly relevant area in the academia and also in the industry. Typically, there is an important demand for well-trained software engineers, since the code in consumer products is doubling every two years approximately [4, 16]. Professionals who have finished their studies in Computer Science or Informatics have many job opportunities, because there is an unsatisfied demand for these professionals.

Every day the software industry is increasing the skills required for these professionals. New trends in software development such as offshore and distributed software development require professionals with new skills [17]. These professionals must be able to perform, for example, asynchronous teamwork, on-demand collaboration, and computer-mediated interactions. In distributed software development, these interaction capabilities are so important than the technical skills.

Nowadays, it is assumed that software engineers must be trained not only in scientific and technical aspects of software engineering, but also the social capabilities that allow them to be effective into the team work [4]. Changes in software practices require changes in the software engineers' education. Therefore, educational institutions delivering computer science programs must prepare their undergraduate students to work in a more interconnected and social software development scenario [5]. The research community has recognized the complexity of developing specific skills in the students [16]. Adopting a practice-oriented instructional approach is considered a best practice in software engineering education [6].

Software engineering not only addresses software products and processes, but also the teamwork. The software development process and the team work include collaboration as a key concept. Therefore, Computer Supported Collaborative Learning (CSCL) activities could be useful to address the challenge to transmit scientific, technical and social skills to the future software engineers.

This paper presents a CSCL activity, named CODILA (Collaborative and Distributed Learning Activity), which was designed to support the teaching-learning process in software engineering courses. This activity supports the delivery of scientific and technical knowledge, and also social skills required by the students in their professional activities. A CODILA involves geographically dispersed students that form small collaborative groups to complete a particular assignment given by the instructors. This activity has been evaluated in a preliminary way with students from five Latin American universities. The obtained results are highly encouraging.

Next section 2 briefly introduces software engineering education. Section 3 presents some related work. Section 4 describes the proposed collaborative learning activity. Section 5 presents the experimentation process and the obtained results. Finally, section 6 presents the conclusions and future work.

II. SOFTWARE ENGINEERING EDUCATION

The IEEE Computer Society defines software engineering as: "The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software" [1]. This includes scientific, technical and empirical knowledge. As was mentioned before, during the last years the software engineering industry has changed the paradigm of using collocated development teams, by several other paradigms oriented to distributed teams. Open source projects, offshore developments and the outsourcing are example of it.

COLLABORATECOM 2010, October 9-12, Chicago, USA

DOI 10.4108/icst.collaboratecom.2010.49

Moreover, the society has been demanding for more and more software products, which usually involves a high innovation degree. The possibility to successfully perform these software projects depends on an adequate supply of proficient and up-to-date software developers [29]. In this scenario, the universities delivering computer science programs play an important role to help face those challenges.

Software engineering is still an emerging discipline, compared with most engineering areas [1]. The Software Engineering Body of Knowledge is in permanent evolution. However there is consensus about the need of including an important number of practical activities in software engineering courses. Some of these practical activities reported in the literature want to include students in real software projects conducted by the industry [23] and also to develop software projects with co-located and distributed student teams [14, 26]. Students must have the opportunity to experience real world practices whenever they can. The smaller the gap between software engineering education and real world practice, the easier students adapt to industry after they graduate [18]. This will improve the quality of the professionals.

The strong globalization of the technology markets requires that many software projects must also be executed in distributed form. The global software development grows day to day [12]. This is another challenge to software engineering education. Educating professionals for these distributed contexts implies not only to train in technical topics but also in social abilities to perform collocated and distributed teamwork. Educational institutions should enrich these practical experiences in software engineering to incorporate multisite, multicultural, and multi-language projects, which most students will certainly face in their professional life [14]. Using CSCL activities appear as an interesting strategy to transmit these capabilities to the students.

III. Computer Supported Collaborative Activities in Software Engineering

Collaborative learning involves intellectual work together to pursue particular learning results [3]. CSCL activities are the result of integrating a learning process to computer-mediated environments. In order to receive the full benefit of social learning, students must interact with each other, share information, negotiate and coordinate actions.

This approach also allows students to give/obtain different perspectives of a problem or issue [3]. The computer support considered in these activities provides students the opportunity to interact with peers that could be located in different places. An important part in this process includes learning a concept, topic or skill through a discussion with other person. Learning together is a model used in higher education to promote reflection on learning, either through joint projects or helping others to acquire a particular knowledge. In the dialogue between students we can obtain multiple perspectives, create a cognitive conflict, promote the development of critical skills and acquire capabilities for professional debate, objectivity and discursive reflection, essential aspects in a collaborative scenario [13].

Each CSCL activity follows a sketch that is performed by the actors (i.e. students, teaching assistants and instructors) to reach a goal. Typically the goal is related to acquisition of a particular knowledge or capability. These activities involve interactions among the participants through computers. Unfortunately, the research work in this area indicates that computer mediation creates potential obstacles to student-student interaction. Specifically, team members tend to experience a slower development of trust, cohesion, efficiency and knowledge sharing, when interact through computers. It usually impacts negatively on the effectiveness of the interaction among participants [8]. Therefore, developers of collaborative learning supporting tools must be creative to try promoting effective student-student interactions [27].

Currently there are proposals for including collaborative models in teaching of several computer science areas, such as Artificial Intelligence [24], Programming [28] and Expert Systems [11]. There are also initiatives that incorporate new teaching-learning strategies in software engineering courses. For example, Manjarres et al. [23] have specified a participatory approach to conduct a software engineering undergraduate course in Spain. The participatory approach involves a practical activity consisting of developing a project based on free software. Such project involves analysis, design and implementation of a software application to manage partners and volunteers of an Open Source organization.

The students are included to an already existing development team inside the organization. These students collaborate with the rest of the team using technology. This collaboration process allows students to learn about engineering best practices and the intrinsic values of the free software development paradigm [23]. Schümmer et al., have described the design of a collaborative project activity to teach distributed Software Engineering Practice using a blended learning design that addresses group formation and trust building problems [31]. Mesa et al. [25] have proposed a strategy for teaching software engineering from the perspective of Problem-Based Learning (PBL), making a coordinated effort for the proper execution of project. Shim et al. [30] have proposed a mechanism to adapt the PBL methodology that is especially designed to be integrated into software engineering classroom in order to promote collaborative learning environment. This approach helps students better understand the significance of social aspects and provides a systematic framework to enhance teamwork skills. Favela et al. [14] proposed to use a software project as a tool to help computer science students in Mexico and USA to collaborate among them. Ochoa et al. [26] proposed a similar strategy, but to support the development of a software project in which the students worked and collaborated in a distributed way.

Many authors have proposed different strategies in order to support the teaching-learning process in software engineering education. However, these experiences do not capture and formalize the rationale behind the design of these activities.

Therefore these successful experiences become difficult (and almost impossible) to reuse. The Collaborative and Distributed Learning Activity (CODILA) proposed in this article has also shown to be useful for students not only to acquire particular knowledge on software engineering, but also the specific skills to collaborate in a distributed development team. This activity has been specified in a way that allows instructors to reuse it using just a low effort

IV. CODILA: STRUCTURE AND DYNAMIC

Although the proposed activity has been designed to support software engineering education, it could be used in any other distributed instructional scenario. The CODILA considers the participation of students and instructors involved in one or more software engineering courses. Although these courses could belong to institutions from the same country, it is recommended to count on groups located in different countries. Thus, the students can feel the experience to work in a more international development scenario.

Students participating in a CODILA can play one of the following roles: expert, student or mediator. The expert is the instructor responsible to deliver a particular knowledge to the students (e.g. through one or more lectures) and also to assign them the practical activities to evaluate knowledge acquisition and the collaboration process. The students are the central actors and final recipients of the teaching-learning process. They are the formal participants of a software engineering course in each of the participating institutions. The mediator is an instructor of a course participating in a CODILA activity, but s/he is not the person acting as the expert in such activity. There could be various mediators in a CODILA and their main task is to monitor the activities of students and group of them. The mediator also supports the students' work by answering basic technical questions and issues related to development of the educational process. At least there is one mediator by each participating university.

Typically, a cooperative learning process involves several tasks that must be developed by the cognitive mediator (or facilitator) and the group of apprentices in order to achieve a final goal [2]. Collazos et al. [9] have divided these activities into three phases according to its temporal execution: pre-process, in-process and post-process. Thus, pre-process tasks are mainly coordination and strategy definition activities, and post-process tasks are mainly work evaluation activities. These phases will be accomplished entirely by the facilitator. However, the tasks concerning the in-process phase will be performed, to a large extent, by the group members. Based on that classification, our proposal defines stages for each one of the learning processes tasks. Additionally, group work considered in a CODILA includes individual tasks. Such activities must be well coordinated and completed to achieve success. Moreover, the group must have clear objectives and each person must have a clearly specified role for the group success [19].

A. Phases of the Model

A CODILA follows a linear process and involves the five stages shown in figure 1: (1) Preparation, (2) Lecture, (3) Local Practice, (4) Distributed Practice and (5) Evaluation.

During *preparation* stage, the instructors in charge of the courses involved in the CODILA will collaborate in order to define the parameters of the activity. It includes defining the topic to teach, the activity main goal, the instructor who will act as expert, the group size and composition, the technologies that will support the collaboration process, schedule of the activities, and the students monitoring and evaluation processes. It is important to consider that the students involved in a same CODILA must have similar levels of expertise and knowledge in the topic to be delivered in such activity. It promotes richer discussions negotiations among students and provide an opportunity to all of them to take advantage of the experience. It is possible to reduce part of this preparation process if the instructors reuse the CODILA design used in a previous year. Clearly it is part of the pre-process phase. The lecture, local practice and distributed practice define activities corresponding to the in-process phase.

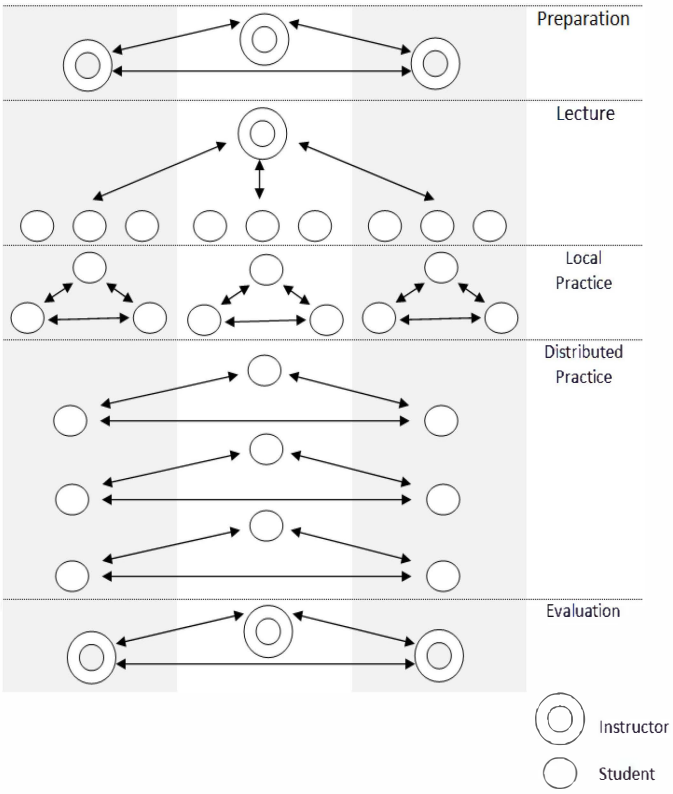

Fig. 1. Structure of a CODILA

During the *lecture* (it could involve more than one lecture), the instructor playing the role of expert delivers the knowledge about a particular topic to all students participating in the activity. It is a synchronous activity in which participate all the involved groups. This activity is similar to a traditional class, where the instructor teaches a topic to a group of students through an expositive class. However, in case of CODILA the class involves local and remote participants. Therefore, the interactions could be face-to-face and computer mediated (e.g. through videoconference). The main goal of this activity is to

give a brief introduction of the selected topic. The activity duration is between 60 and 90 minutes.

The *local practice* is performed during the session after the lecture. This is a practical activity that tries to help students to assimilate the knowledge delivered by the expert during the lecture. The local practice involves teams composed by 3-4 students belonging to the same institution, whom work collaboratively during a class to solve an assignment. The groups members are randomly selected and we structure the tasks so that each group member can make an equal contribution. Every participating institution must count on a person who is in charge of the activity, an instructor playing the role of mediator (i.e. the monitor). This person provides students the material required for the activity, and also s/he monitors the whole process. The activity duration depends on the type of assignment given by the instructors, but it is recommended to perform it in a class of 90-120 minutes.

The *distributed practice* must be started the session after the local practice. However it could involve 1 or 2 weeks of the students work. Similar to the local practice, students form groups to solve the assignment given by the instructors. The main differences with the previous one are mainly two: (1) students participating in a distributed practice belong to different institutions, and hopefully different countries; and (2) the assignment is larger than the previous one. Since students are geographically dispersed, they need to interact, communicate and cooperate with their teammate using technology, and also dealing with different time zones. Similar to the previous case, groups are selected randomly, including at least one member of every participant university in every work group.

The team members must collaborate to solve the assignment. This aspect corresponds to some of the elements are mandatory to include in a collaborative activity: equal participation and individual accountability [20]. The activity to be performed could be executed in synchronous, asynchronous or multi-synchronous way.

Finally, the learning process performed by the students is evaluated. The *evaluation* stage intends to quantify the learning results related to this process, and also find opportunities to improve the performed activity. The evaluation involves several aspects: acquired knowledge, supporting technologies, and collaboration capabilities. This stage is done in a collaborative manner with the active participation of all the participant universities. We utilize users' satisfaction surveys, usability tests and co-evaluation mechanisms to conduct the evaluation process, this last one has not been done in the experimentations we have developed, but for further work it is one of the elements we are going to include. It corresponds to an activity of the post-process phase.

B. Team Work Dynamics

Work groups define their working strategy. This stage is very important because it determines the level of participation and communication they need to do, in order to obtain a good work.

Two experiences were done using CODILA. In the first one, the students were free for choosing the best way to interact with their partners. In the second one, the students worked as collaborative teams in order to obtain the learning benefits of the proposed activity [10]. Each group must follow an activity dynamic similar to JIGSAW[1]. In this instance, the expert distributes material to students and also the goals of the practical activities to be performed by each group. These practical activities are divided into as many parts as members have the group. Each student is responsible for one of these parts and must perform his/er task individually. Therefore, each student must be formally and individually tested to determine the extent to which s/he has mastered and retained the targeted academic content and abilities [21].

After that, students form pairs of specialists conformed by participants from different groups, but with the same assignment. These pairs discuss their work, and eventually may correct or adjust the results of their individual work. This instance of specialization is performed in a co-located way.

Finally each student returns to his/er original group and explains/discusses the results with their teammates. This activity is repeated with each team member; therefore, when this knowledge sharing process finishes, all members could learn about the different thematic involved in the practical activity. This activity could be performed in a distributed way if the students belong to different universities. At the end, each group should consolidate results achieved by their members and agree on a single group result that is then evaluated by the expert.

V. Experimentation Scenarios

Two experiences were done using CODILA in advanced software engineering courses. The first experience was in 2008 and it involved computer science students from the following universities: University of Quindio (Colombia), University of Chile (Chile) and National University of San Juan (Argentina). The second experience was in 2009 and it involved courses from four Latin American universities: University of Cauca (Colombia), University of Quindio (Colombia), Technological University of Panama (Panama), and National University of San Juan (Argentina). Next sections describe these experiences and presents obtained results.

A. First Experience

During the second term of 2008, the authors used CODILA to teach students to estimate the development effort of a software project. The goal was to transfer students the knowledge and skills related to software estimation techniques. The students were in the final years (9th – 10th semester) of Computer Science programs in their respective universities.

In this experimentation, the lecture (second stage of CODILA) could not be performed simultaneously in the three universities, because there were some technological problems generated by the ISPs. An instructor of the University of Chile

[1] www.jigsaw.org

(UCh) gave the class through a videoconference that involved 19 students from the National University of San Juan (UNSJ). This online class was conducted using the VoIP functionality of the Skype software and Conference XP. These platforms allow participants to interact through videoconferences and a chat, and also share slides and a computing desktop. Then, the same instructor repeated the lecture to students of the University of Quindio (UQ), in which participate 22 students. The working group of UQ used the network RENATA (High Performance Network to Support Education) [32] as the basis for collaborative virtual environment with the UCh. Academic Networks, such as RENATA or CLARA (High Performance Network to Support Latin American Cooperation) [33], facilitates collaborative academic work, allows sharing information and accessing to remote resources with a high bandwidth. These networks also facilitate the communication and teamwork among researchers who are geographically dispersed in different regions, thus facilitating the development of joint academic and research experiences.

Finally, the instructor gave the class in traditional way (expository and co-localized) to students at the UCh. This group consisted of 21 students from an undergraduate program in Computer Science.

Distributed practice (i.e. the second phase of the model) was performed through distributed working sessions, in which the students participated in both modes (synchronous and asynchronous). The instructors formed 21 students groups, each group consisting of 3 students, one student from each country involved in the experience. The formation of the groups was randomly, respecting the above condition.

Students had to solve a practical assignment, which involved an important complexity level. Students were free to choose the communication tools that they used. Each student was provided with the teammates contact information; therefore they were able to communicate with each other according to their needs.

During this experience students were highly motivated to interact with peers from other universities and, beyond the constraints of mediated communication technologies; the motivation was maintained through the whole process. Students used, at their option, multiple media such as chat, VoIP, email, and video-conferencing. Students had to "negotiate" with each other to arrive at agreements about technologies used for communication, ways to solve the problem, joint work schedules, and distribution of tasks. These instances for negotiation pushed them to use skills and capabilities few used by them in regular courses, but that are very important in their software engineering instruction. Finally, the instructors performed an evaluation of the experience, which is the last phase of the model. It consisted of a users satisfaction surveys and also a student's co-evaluation.

B. Results of the First Experience

The results of the academic practices were good and satisfaction of all students was high. However, we had some technological limitations that influenced the results of the collaboration process. For example, students have had some problems to perform synchronous work because the network bandwidth was low during several time periods.

Initially the three involved universities were in different time zones; however the University of Chile changed its time zone during the experience. In the students' opinion, this aspect does not represent a problem for them if the involved time zones are close.

The anonymous user satisfaction survey utilized in the evaluation stage indicated the students were very pleased with the learning experience. In Figure 2 shows the distribution of answers corresponding to the item "the experience of this type of learning was interesting". The Y axis indicates the number of responses that were received.

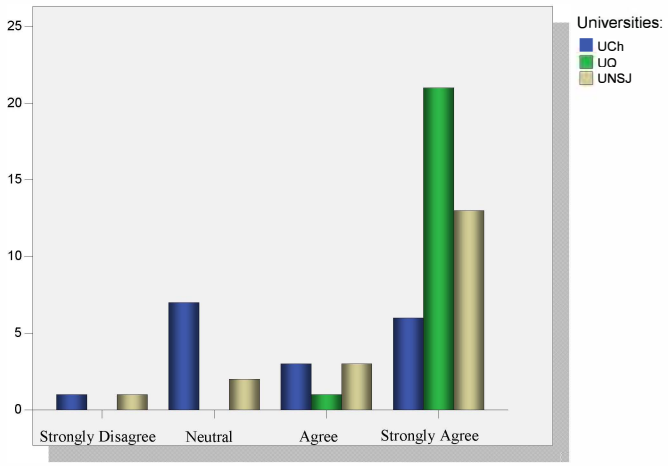

Fig 2. Value of the experience from the students' perspective

Students thought the experience enriched their knowledge and interaction skill. The vast majority of the surveyed students (almost 90% of them) said the experience allow them to improve their social skills to propose and support ideas during a discussion with peers, participate positively in work groups, and build and acquire knowledge of peers. Figure 3 shows the distribution of students answers related to the item "the learning activity was useful to enhance/improve your social skills" (in a scale from 1 to 5), where 1 is "strongly disagree" and 5 is "strongly agree". The Y axis indicates the number of responses that were received.

The most influent aspect of CODILA was the need of interact and negotiate with peers from other countries. Many of these students keep the contact with their teammates up to today.

This experience, besides evaluating of collaborative learning and the intensive interaction between students from different cultural and social contexts, may also be considered as a distributed laboratory to control experiments in software engineering practices.

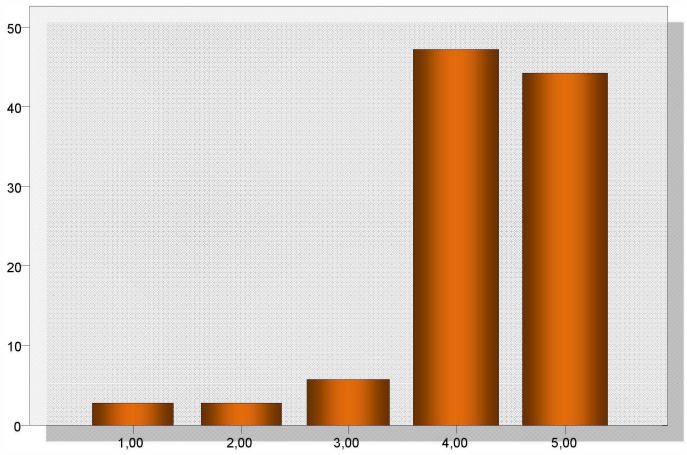

Fig. 3. Value of CODILA as tool that helps improve students' social skills

Table 1 shows the grades obtained by students participating in each group. These grades range from 0 to 10 and the minimum for approval is 4.

Table 1. Grades of the Groups

Group	Grade
1	8
2	6
3	7
4	7
5	4
6	8
7	7
8	8
9	6
10	10
11	7
12	7
13	10
14	9
15	7
16	10
17	7
18	10
19	9
20	7
21	8
AVERAGE	**7.7**

Most students got good marks. These marks were higher than those they usually get in other practical assignments in the same course, but that does not involve collaboration. These preliminary results could be indicating that CODILA helped students to improve their motivation and performance. They were four groups that got the maximum grade, one of those received additional comments by the instructor due to the quality of the work developed. In the group with a grade of 4, the participants were not interested into collaborate. Therefore they split the task in three parts, and then they integrated their contributions and submitted the result to the expert. Clearly, the quality of the obtained product was poor. Instructors monitoring the activities feel comfortable with the experience and they were available to repeat the experience. In fact, two of them repeated the experience.

C. Second Experience

During the second term of 2009, CODILA was used in a similar scenario. In this opportunity participated undergraduate students of advanced software engineering courses from four universities: 36 students from University of Cauca (Colombia), 23 students from University of Quindio (Colombia), 20 students from National University of San Juan (Argentina), and 56 students from Technological University of Panama (Panama). University of Quindío and National University of San Juan involved the same courses than the first experience, but with different students. During the experience, the instructors involved students in two types of work teams: collaborative and ad hoc. Collaborative teams followed the CODILA process to support the students work, and the ad hoc teams were free to organize the work at their convenience.

In this experience, all students participated in the lecture that the expert of University of Cauca delivered about the topic: usability evaluation. The lecture shows techniques of inquiry, inspection, test and mechanisms for evaluating accessibility in interactive environments. The lecture was delivered synchronously for all participants (i.e. local and remote students). Remote students received the lecture through a videoconference supported by Microsoft LiveMeeting.

During the next stage, students were distributed in collaborative groups and ad hoc groups (control groups). Collaborative groups adhered to the CODILA process, while ad hoc groups were free to organize the work as they want. This experience involved 17 collaborative groups and 16 ad hoc groups. Each group consisted of 4 or 5 students, involving at least one member of each participating university. The practical assignment given to the students was the following one: "Evaluating the usability of the UTP Web site". For the experiment we used the e-learning environment named AulaNet [15]. For the development of activities various services were enabled in this environment.

Students performed the evaluation process in teams following the dynamic established by the type of group they belong to (i.e. collaborative or ad hoc). At the end of the activity, each group presented a report with the results of the evaluation done to the UTP Web site. Mediator instructors in each participating university contributed to this process resolving some questions generated during activity. Finally, as part of the third stage (evaluation), students answered a survey about the performed experience, which was evaluated again by instructors to gain feedback and adjusting the process. In this phase expert also evaluated quantitatively the reports of the

student groups.

It is important to emphasize that collaborative groups had, by default, a strategy to work together provided they were using the CODILA model. Each team member had assigned a particular technique for evaluating the usability of user interfaces. Therefore, each member had to study individually the assigned technique, becoming a specialist in it, and apply it to evaluate the usability of the Web site. Then, the students specialists in the same usability evaluation technique are meet and invited to share their expertise. Finally, collaborative groups of students returned to meet with their teammates, but now they count on a specialist in each technique. Then, they had to produce a single technical report with the Web site usability results, which was agreed between them. In each instance students used typical communication tools, such as discussion forums, chat, email, IP videoconference applications and instant messengers.

The ad hoc groups served as control groups in this experiment. Unlike collaborative groups, they were free to organize and coordinate work at their convenience. These groups received the same initial class and they were asked for the same final technical report than collaborative groups. Specific activities were not required to execute the work; each of these groups self-organizes its activities to obtain the final result. Mediators' instructors followed up students' activities through the whole process.

D. Results of the Second Experience

At the end of experimentation process, instructors evaluated the results obtained by the groups. They found that collaborative groups' performance was better than ad hoc groups. However, the difference was not significant in terms of the skills of group reports. The average score in the collaborative groups was 9.3 versus 9.2 of ad hoc groups (in a scale of 0-10). In this experience we use decimals to represent the grades because the tool used to support the experience (i.e. AulaNet) imposes such feature. Table 2 shows the grades obtained by each participating group.

After analyzing the interaction record of students participating in collaborative groups, and talking with students participating in ad hoc groups, the instructors concluded that counts on an initial working strategy improves the students' performance. Therefore, collaborative groups (in which the strategy was imposed by instructors) had a good overall performance.

Supporting this similar perception, ad hoc groups that performed well are those who have defined themselves an early working strategy. Hence, it is possible to assume that if group works with an ad hoc or collaborative approach has no direct implication on the work result. Rather, it is possible to conclude that groups that were organized early, did better than those who did not.

Table 2. Grades of the Groups in the Second Experience

Collaborative	Ad hoc
9.2	---
9.2	9.2
9.8	8.8
9.6	9.6
9	8.7
9.2	9.6
9.6	9
9.2	8.8
9.2	9.6
8.7	9.2
9.7	8.8
9.6	8.8
9	9.6
9.8	9.4
9.5	9.6
9.2	8.6
9.2	9.6
Average: 9.3	**Average: 9.2**

The ad hoc groups had more messages exchanges than collaborative groups. This is understandable because possibly collaborative groups, who had defined the strategy work, do not need discuss this topic. However, ad hoc groups need it. The most important issue is count on a common element to define an appropriate strategy, commitment and a common goal [7].

In order to obtain good results it is necessary to structure the activity in a way that really conveys an effective collaboration. CODILA defines a set of stages to guarantee this collaboration. We can observe that groups which individual work was effective, it is where there was more participation of team members, the final group score was better. It implies that in a collaborative activity, individual accountability it is as important as cooperation.

At the end of activity students fill out a questionnaire where they answer questions related with user satisfaction about the experience, and aspects they consider the most significant. Their level of satisfaction, represented in a scale from 1 (poor) to 5 (high), can be seen in Figure 4.

This experimentation showed us that CODILA helps students to obtain good performance in terms of the students' grades, but also the competences in terms of abilities to collaborate; it is the skills to work in groups and interact with

people dispersed geographically.

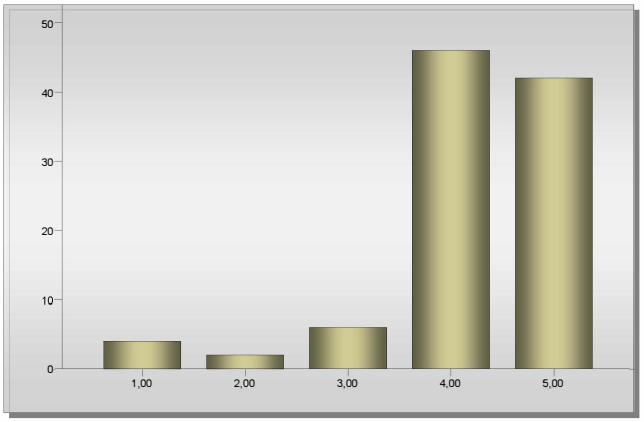

Fig. 4. Satisfaction Level of the Students

The group of students who performed the activity using CODILA were the students who the most considered the practice as important, mentioning for example that due to this experience they realized their capacities to work with person of different cultures, their ability to communicate with people they do not know and their capacity to apply the strategy "learned" during the practice in different experiences when they work in groups.

Instructors involved in this experience think these opportunities for collaborative work will allow students to improve their abilities to engage in projects with people from another country, without fear of failure, reducing uncertainty, making them actors in a global software development scenario.

VI. Conclusions and Future Work

Most students graduated of computer science programs are going to work professionally in areas related to software engineering. Software engineering not only is a discipline highly required by the industry, but also it is an expertise area that is evolving constantly. This pushes universities to prepare professionals able to manage technical and human aspect of the software development process. Therefore, these institutions must constantly refine curriculum trying to cope the new challenges proposed by the industry. Team work seems to be a medium that allow students to gain experience in building software inside a traditional or a global scenario.

Experience in team work should be complemented with contemporary trends imposed to software, since it becomes one of the sectors most affected by the phenomenon of globalization and market opening. Moreover, the geographical distribution of customers also push professional to the same direction. Therefore, it is necessary to strengthen the software engineering education by providing collective experiences in software development [22].

This paper proposed a computer supported collaborative learning activity named CODILA. This activity was initially conceived to support software engineering education in distributed scenarios; however it could also be applied in contexts where interaction between geographically dispersed participants can enhance the knowledge in a particular area, such as history, sociology, or learning of foreign languages.

CODILA was applied to support two experiences in Latin America. Students and instructors participating in these experiences feel comfortable and motivated with the activity. They think that CODILA allows undergraduate students to acquire not only particular knowledge about software engineering, but also specific skills required to collaborate in a distributed development team.

Evidence from the experiences presented in this paper must be corroborated and expanded by new applications of the proposed collaborative learning activity. For future experiments we will be considering several software engineering topics suggested by students in the anonymous surveys.

Nowadays, we are also implementing the Latin American Co-laboratory on eXperimental Software Engineering Research (LACXSER), that count on the support of several organizations, such as LACCIR (Latin American and Caribbean Collaborative ICT Research federation), Colciencias (Colombia) and Education Ministry of Argentina. The next steps in this initiative are to continue learning from these experiences and also to increase the number of participating universities.

Acknowledgements

This work is partially funded by the project entitled "Fortalecimiento de la Red de Investigación Aplicada en Ingeniería de Software Experimental", in the II Call of "Proyectos de Fortalecimiento a Redes Interuniversitarias", Ministry of Education, Argentina. Authors thank to CINTEL-Colciencias (Colombia) for partial funding of this work through the Project entitled "Red Latinoamericana de Investigación Aplicada en Ingeniería de Software Experimental", Grant IF-007-09 (Call Colciencias 487 - RENATA 2009), and also LACCIR Grant R1209LAC003.

References

[1] Gabran, A., Moore J. W. Guide to the Software Engineering Body of Knowledge (SWEBOK). IEEE Press, 2004.

[2] Adams, D., Hamm, M. Cooperative Learning, Critical Thinking and Collaboration Across The Curriculum. Second Edition, Charles Thomas Publisher, 1996.

[3] Alavi, M. Computer-Mediated Collaborative Learning: An Empirical Evaluation. MIS Quarterly, 18(2), pp. 150-174. 1994.

[4] Bagert, D., Hilburn, T., Hislop, G., Lutz, M., McCracken, M., Mengel, S. Guidelines for Software Engineering Education, Version 1.0. Technical Report CMU/SEI-99-TR-032, 1999.

[5] Bareiša, E., Karčiauskas, E., Mačikėnas, E., Motiejūnas, K. Research and Development of Teaching Software Engineering Processes. Proc. of the Int. Conf. on Computer Systems and Technologies. Bulgaria. 2007.

[6] Carver, J., Jaccheri, L., Morasca, S., Shull, F. Issues in Using Students in Empirical Studies in Software Engineering Education. 9th Int. Software Metrics Symposium (METRICS'03), 2003.

[7] Collazos, C., Guerrero, L., Pino, J., Ochoa, S. Improving the Use of Strategies in Computer-Supported Collaborative Processes. Proc. of the 9th Int. Workshop on Groupware (CRIWG'03). Grenoble, France. Springer Verlag LNCS, 2806, pp.356-370. 2003.

[8] Collazos, C., Ochoa, S., Mendoza, J. Improving the Schemes of Evaluation Through Collaboration Processes (In Spanish). Revista Educación y Educadores 10(1). 2007.
[9] Collazos, C., Guerrero, L., Pino, J., Ochoa, S.F. Evaluating Collaborative Learning Processes. Proc. of the 8th Int. Workshop on Groupware (CRIWG'02), Springer Verlag LNCS, 2440, September, 2002.
[10] Collazos, C., Giraldo, F., Ochoa, S., Aballay, L., Clunie, G., Zapata, S., Clunie, C., Lund, M. Teaching Usability form a Collaborative Perspective (In Spanish). V Congreso Colombiano de Computación, 2010.
[11] Cuneo, C., Mariño, M. Collaborative Environment in the Education of Expert Systems (In Spanish). Comunicaciones Científicas y Tecnológicas. 2005.
[12] Damian, D., Moitra, D. Global Software Development: How Far Have We Come? IEEE Software, 23(5) pp.17-19. 2006.
[13] Falchikov, N. Learning together: Peer tutoring in higher education. London: Routledge Falmer. 2001.
[14] Favela, J., Peña-Mora, F. An Experience in Collaborative Software Engineering Education. IEEE Software, 18(2), pp. 47-53. 2001.
[15] Fuks, H., Raja, L.H., Gerosa, M., Lucena, C. Competency Management for Group Formation on the AulaNet Learning Environment. Proc. of CRIWG'03. LNCS, Vol. 2806, pp. 183-190. 2003.
[16] Dick, B. Simmons, R. Software Engineering Education in the New Millennium. Proc. of the 30th Annual International Computer Software and Applications Conference (COMPSAC'06). IEEE Press.2006.
[17] Hawthorne, M., Dewayne, E. Software Engineering Education in the Era of Outsourcing, Distributed Development, and Open Source Software: Challenges and Opportunities. Proc. of the 27th Int. Conf. on Software Engineering (ICSE). St. Louis, USA. Pages: 643 - 644. 2005.
[18] Huang, S., Distante, D., On Practice-Oriented Software Engineering Education" Proc of the 19th Conference on Software Engineering Education and Training Workshops (CSEETW'06). Turtle Bay, Hawaii. pp.15. 2006.
[19] Johnson, D., Johnson, R., Holubec, E., Cooperation in the Classroom. Boston: Allyn and Bacon, 1998.
[20] Johnson, D.W., Holubec, E. Cooperation in the Classroom. Edina, MN: Interaction. 1993.
[21] Kagan, S. Cooperative Learning and Social-cultural Factors in Schooling. In California Department of Education, Beyond Language: Social and Cultural Factors in Schooling Language Minority Students. Los Angeles, CA: Evaluation, Dissemination And Assessment Center, California State University, Los Angeles. 1986.
[22] Lund, M., Zapata, S., Aballay, L., Herrera, M., Torres, E., Collazos, C., Giraldo, F., Ochoa, S., Evaluation of collaborative instructional process to software engineering in distributed learning environment (In Spanish). Revista Avances en Sistemas e Informática 6(2) 2009.
[23] Manjarres, A., Arias, M., Gaudioso, E. Transverse Competencies in software engineering teaching (In Spanish). VI Jornadas de Redes de Investigación en Docencia Universitaria, Universidad de Alicante. 2008.
[24] Mariño, S. Virtual Environment Design of Teaching-Learning for an Artificial Intelligence Course (In Spanish). Revista Electrónica No.3. 2008.
[25] Mesa, J., Alvarez, J., Villanueva, J., Cos, F. Update of Teaching-Learning Methods in Courses of Engineering Project Management (In Spanish). Formación Universitaria, 1(4), pp.23-28. 2008.
[26] Ochoa, S., Pino, J., Guerrero, L., Collazos, C. SSP: A Simple Software Process for Small-Size Software Development Projects. First IFIP International Workshop on Advanced Software Engineering, Santiago, Chile. Springer Science + Business Media. Vol. 219. pp. 94-107. August, 2006.
[27] Orvis, K., Lassiter, A. Computer-Supported Collaborative Learning: Best Practices and Principles for Instructors. Information Science Publishing, Hershey, New York. 2007.
[28] Redondo, M., Mendes, A., Ortega, M. Collaborative Planning of Design for Programming Learning (in Spanish). International Workshop in Educational Software, Chile. 2001.
[29] Shaw, M. Software Engineering Education: A Roadmap. Proceedings of the Conference on The Future of Software Engineering. Limerick, Ireland. ACM Press, pp. 371 - 380. 2000.
[30] Shim, C., Choi, M., and Kim, J. Promoting Collaborative Learning in Software Engineering by Adapting the PBL Strategy. World Academy of Science, Engineering and Technology, Vol. 53, pp. 1157-1160, 2009.
[31] Schümmer, T., Lukosch, S., Haake, J., Teaching distributed software development with the project method, Proceedings of the 2005 conference on Computer support for collaborative learning: learning 2005: the next 10 years!, pp. 577-586, 2005.
[32] RENATA. High Performance Academic Network. URL: http://www.renata.edu.co/, Last visit: July 2010.
[33] CLARA. High Performance Network to Support Latin American Cooperation. URL: http://www.redclara.net/, Last visit: July 2010.

Cesar A. Collazos, received his degree in System Engineer at Universidad de los Andes (Bogota-Colombia). PhD in Computer Science at Universidad de Chile. Full Professor at University of Cauca (Colombia), head of the IDIS Research Group. His research areas are CSCL, CSCW and HCI.

Sergio F. Ochoa is assistant professor of Computer Science at the University of Chile. He received his Ph.D. in Computer Science from the Catholic University of Chile. His research interests include computer-supported collaborative work, educational technology and software engineering. Dr. Ochoa is a member of IEEE, ACM and the Chilean Computer Society and sits on the Steering Committee of the LACCIR (Latin American and Caribbean Collaborative ITC Research Initiative) and CLEI (Latin American Center for Studies in Informatics). His research interests are CSCW/L and Software Engineering.

Sergio G. Zapata currently has a position as full professor in Computer Science at the Informatics Institute, National University of San Juan, Argentina. His research focuses on topics related to Software Engineering: overall process of software development, software quality and learning of software engineering. He has over 20 years of experience as independent consultant in Software Process.

Fáber D. Giraldo received his degree in Systems and Computer Engineering from the University of Quindío, Colombia. He is a candidate for a M.Sc. degree in Engineering, major field in Informatics, from the EAFIT University in Colombia. He is involved with the research group SINFOCI (Information System and Industrial Control) from the University of Quindío. His major research interests are software engineering and model driven development.

M. Inés Lund received her degree in Informatics and obtained her Specialist degree in Information Systems at the National University of San Juan. She is a candidate for Master in Informatics at National University of La Matanza (Argentina). She is part of the Researchers staff of Informatics Institute at National University of San Juan. Her major research fields are software engineering and software design.

Laura N. Aballay is a Programmer form the National University of San Juan (Argentina). Currently she is Advanced Student in Informatics in such University. She is also Research Assistant at Informatics Institute of National University of San Juan. Her research areas are software engineering and software design.

Gisela Torres de Clunie received her degree in Informatics at Technological University of Panama (Panama City, Panama), M.Sc. and Ph.D. in Systems and Computer Engineering from COPPE – Federal University of Rio de Janeiro (Brazil). She is Full Professor at Technological University of Panama. Her major research fields are Software Engineering and Virtual Education.