Applying a Distributed CSCL Activity for Teaching Software Architecture

Fáber D. Giraldo
System and Computer Engineering
University of Quindío, Armenia, Colombia
e-mail: fdgiraldo@uniquindio.edu.co

Sergio F. Ochoa
Computer Science Department
University of Chile, Santiago, Chile
e-mail: sochoa@dcc.uchile.cl

Myriam Herrera
Informatics Institute
National University of San Juan, San Juan, Argentina
e-mail: mherrera@iinfo.unsj.edu.ar

Clifton Clunie
Computer Systems Engineering
Technological University of Panama, Panama
e-mail: clifton.clunie@utp.ac.pa

Andrés Neyem
Computer Science Department
Pontifical Catholic University of Chile, Santiago, Chile
e-mail: aneyem@ing.puc.cl

Sergio Zapata
Informatics Institute
National University of San Juan, San Juan, Argentina
e-mail: szapata@iinfo.unsj.edu.ar

José Luis Arciniegas
IDIS, University of Cauca, Popayan, Colombia
e-mail: jlarci@unicauca.edu.co

Fulvio Lizano
Informatics School
National Autonomous University of Costa Rica, Costa Rica
e-mail: flizano@una.ac.cr

Abstract— **Teaching and learning software engineering have been recognized as important challenges for computer science students and instructors. These challenges become stronger if we consider the distributed software development scenario that is gaining space everyday into the software industry. In order to help address this challenge, this article introduces a Collaborative and Distributed Learning Activity (CODILA). This collaborative activity helps students to reach the professional skills required in the professional work, and instructors to perform such transfer process. The paper also presents and discusses the obtained results from a particular CODILA, which address the challenge of designing a software architecture of a communication infrastructure in a distributed way. Six Latin American universities participated in the activity. The obtained results were satisfactory and indicate these types of experiences can be used to address the stated challenges.**

Keywords-CSCL, CODILA, Collaborative Educational Experience, Teamwork, Distributed Collaboration, Software Architecture.

I. Introduction

Many of Software Engineering areas present different challenges in their teaching process. Among the factors that influence these challenges are the software scalability and complexity, the software design and construction under the cost and quality constraints, and substantial human aspects that affect all areas/activities of Software Engineering practices. The complexity of software construction is well known. This complexity includes technical aspects (such as, the use of IDEs and programming languages), cognitive (e.g., understanding key practices) and social (e.g. issues related to teamwork). As result of this multifaceted nature, Universities delivering Software Engineering courses need to put special attention to the tasks performed by students in order to: (1) improve their capability to analyze and solving problems, (2) expose them to a real world scenario of software development, and (3) teach them to work geographical distributed.

Those requirements generate particular challenges to the Software Engineering teaching/learning process, since it is difficult that students of a regular academic program can work real situations in order to get the skills required for their professional lives.

Software Engineering is largely based on team work, which can be co-located or distributed. The growing outsourcing tendency bases on keeping distributed teams and supporting their activities through strong capabilities for teamwork; therefore students need to develop such skills [1]. Teamwork is a mechanism that allows students to gain experience and understand the dynamism of the professional work.

This paper presents the results obtained in a Collaborative and Distributed Learning Activity (CODILA) [2], in which participated six Latin American universities. This Computer Supported Collaborative Learning (CSCL) activity involves student teams that work in a collaborative and distributed way to solve a problem; in this case it was the design of a software architecture for a secure communication system. The activity execution was monitored by the authors in order to understand the individuals' and teams' performance.

Next section briefly introduces principles of CSCL and team work. Section 3 describes the distributed CSCL and experimental software engineering principles. That section also introduces the template to specify a CODILA. Section 4 presents collaborative distributed experience performed on software architecture. Finally, section 5 concludes and presents the further work.

II. CSCL And Teamwork

CSCL is a pedagogical approach that is supported on the principles of cognitive theories; for example the psychogenetic approach of Jean Piaget, Vygotsky's sociocognitive theory, the theory of activity of Leontiev and Galperin, and the Theory of Cognitive Modifiability of Feuerstein. These theories share the cognitive and metacognitive development of students in real, symbolic and dynamic contexts that allow them to recreate their knowledge and potentiate them.

Teamwork is an example of the importance of preparing students for practice. As an engineering discipline, Software Engineering has a strong emphasis on application of knowledge and preparation for the professional practice. Software Engineering also addresses a variety of human aspects in the process of building software, which go beyond of teamwork. The combination of technical foundations and practice case studies makes the teaching of Software Engineering particularly challenging. It implies that the instructors' profile has to include professional experience in Software Engineering. Since the pool of candidates with academic and professional experiences is relatively limited, establishing academic cooperation in this area with other universities can help address this challenge.

A. Distributed CSCL applied to Software Engineering

Next we present a list of key elements for collaborative learning, which guides the teaching/learning process of software engineering topics in a distributed way.

- *Positive interdependence*: This is the central element that covers the organizational and operational conditions that must occur within the team. Team members should be supported by each other, and trust in the understanding and success of the teammates. Interdependence must be considered when assigning goals, tasks, resources or roles.
- *Interaction*: The ways of interaction and verbal exchanges between members of a team, driven by positive interdependence, are those that affect learning results. The interaction allows realizing the tracing and exchanging between different team members. Student learns from each partner whom they interact with. The team can enrich itself if it has different interaction modes, enhancing thus its reinforcements and feedback.
- *Individual contribution*: Each member must assume the assigned functions. They must also have the space for sharing with the teammates and receive their contributions.
- *Personal and team skills*: The experience of the team must allow each member to develop or enhance his/er personal skills. Such skills must also allow the team's growth and skills acquisition, such as listening, leadership, evaluation, participation, coordination of activities and monitoring.

The goals of CSCL activities for teaching/learning software engineering in a distributed way are the following ones: promoting interaction among teammates, assessing the individual contributions, developing personal and group skills (e.g. negotiate, debate, question, make decisions and coordinate).

Students need to be continually questioned about their personal and group performance. It helps to reach some advantages of collaborative learning, such as:

1. *Concerning the execution of group tasks*:

- It promotes the achievement of goals and the quality of the work, since the team elaborates its proposal based on its members' proposals and solutions.
- It helps to disseminate and validate the knowledge of group members.
- It promotes the commitment of each member with the team.

2. *Concerning the interpersonal relations*:

- It encourages the feeling of solidarity and mutual respect, based on the results of teamwork.
- It strengthens social skills and communication.

3. *Concerning the cognitive skills*:

- It promotes the knowledge generation due to each student is involved in the development of research.
- It promotes the development of critical thinking and meta-cognitive development.

4. *Concerning the self-development*:

- It empowers the self-confidence.
- It evidences an intrinsic motivation in individual and group work.
- It contributes to decrease the feelings of isolation.
- It decreases the level of frustration to criticism and fear of failure.

The main obstacles for collaborative learning are usually three: (1) the resistance to the paradigm change required by students and instructors, (2) counting on an appropriate design of the collaborative activity to apply, and (3) having a good technological solution to support the activity (especially in distributed environments). Therefore, when we design a CODILA, we should clarify the educational principles, stakeholders and the teaching/learning processes that allow students to achieve particular knowledge and skills.

III. Distributed CSCL And Experimental Software Engineering Principles

Software industry becomes globally distributed, mainly driven by the development of trends, such Open Source software and developments outsourcing. Therefore software practitioners require significant changes into their formative stages; i.e. in the knowledge and mainly the skills considered by the educational institutions in undergraduate programs.

A model of distributed multi-faceted software development is emerging and demands the application of engineering principles in new and unfamiliar contexts. For ensuring the successful development of software using distributed resources, software engineers need to count on a solid foundation of project and product management, engineering of organizational processes, and multi-cultural communication skills [3].

The adoption of collaborative technologies in a software development organization involves the introduction of new technologies and work practices within the comprehensive and highly variable activity of software development. Productivity and deliverables of software projects are factors significantly variable. Therefore, it is difficult to assess the impact of the introduction of new technologies in a project [4].

Provided the distributed collaborative work in software development processes and the geographical distribution of software team members, it is mandatory to formulate experiments to validate the application of collaborative techniques in current software engineering teaching/learning scenarios [5]. Research and experimentation in collaborative work applied to software development should focus on factors such as negotiation among participants and stakeholders, distributed communication and coordination, and use of new engineering processes and strategies.

Experimentation scenarios should consider the distance among members as a critical factor influencing the collaborative learning/work [5]. Differences in the physical contexts (i.e. distributed and co-localized), time zones, culture and language will persist despite the use of new technologies. Remote work is enhanced when it is supported by high speed networks, such as RENATA [6] academic network in Colombia and RedCLARA[7] in Latin America. However, some distances aspects will remain present explicitly without possibility of support it even in the future.

It is well-known that software development processes require the participation, expertise and ability of many people working together to achieve the project goals. Therefore, it is necessary to establish new mechanisms that let students to acquire skills of teamwork and effective communication [8] to address these new working scenarios. It is relevant not only from a technical perspective, but also to develop competencies and skills required by students to act as members of a distributed software development team [9].

From this perspective, the consortium of Latin American universities involved in the initiative LACXER[3] has proposed a collaborative model to support the distributed teaching-learning process for software engineering named CODILA (COlaborative and Distributed Learning Activity) [2]. This instructional process promotes the distributed learning through collaborative work. The structure and dynamics of the model involve participants geographically dispersed and enhance the active participation of team members. The CODILA model contains five stages: Preparation, Theoretical Class (lecture), Local Practice, Distributed Practice and Evaluation. However the local practice is optional.

During *preparation* stage the instructors will collaborate to define the parameters of the activity, such as: the topic to teach, the activity main goal, the instructor who will deliver the lecture, the group size and composition, the technologies that will support the collaboration process, and schedule of the activities.

During the *lecture* the instructor playing the role of lecturer delivers the knowledge about the select topic to all students participating in the activity. It is a synchronous activity in which participate all the students involved in the experience.

The *local practice* is performed during the session after the lecture and its goal is to help students to assimilate the knowledge delivered by the expert during the lecture. Typically it is a collaborative solving problem activity that involves 90-120 minutes and it is related to the topic tough by the lecturer.

The *distributed practice* involves at least one week and also team of distributed students with a common goal. This practice has three sequential phases: (1) individual work, (2) collaborative session and (3) integrative work. During the first phase the work is divided in as much parts as members has the team. Each member is responsible to complete his/her assignment. During the second phase the team performs a collaborative session where each student explains his/her work to the teammates. Thus the knowledge flows inside the team. Finally, the individual works are integrated and adjusted to form a single product that will represent the outcome for each team. Such outcome is delivered to the instructors.

During the *evaluation* stage such outcome is evaluated and also the collaboration process and supporting technologies used into the experience.

The authors have designed a template to specify a CODILA, which has been useful as mechanism for synchronization, feedback and managing the activity design (see Table I). The template has also helped to socialize, validate and adjust the experience. Instructors from the participating universities are free to contribute with their suggestions and comments. Table I shows the structure of the template to specify these distributed collaborative experiences.

IV. Teaching Software Architecture From A Collaborative Distributed Perspective

In 2010, we have applied the CODILA model in a collaborative experience for teaching Software Architecture topic. In this experience participated six Latin American universities from four countries. These universities were grouped in two subsets as follow:

Track 1 (50 students)

- Pontifical Catholic University of Chile, Chile (PUC).

TABLE I. TEMPLATE FOR DESIGNING CODILA EXPERIENCES

SECTION	PURPOSE
Topic	It refers to the topic that will be used in the distributed collaborative activity.
Short description of collaborative experimental activity	It briefly describes the learning activity to be performed. Using this description an instructor must determine whether the activity is adequate for your course.
University leader / name of instructor(s) responsible(s)	University proposing the experience will guide the implementation of the activity. it should be specified the name and contact information of the expert instructor that will be delivering the lecture.
Participating universities	It is the list of participating universities that will be performing the activity. The list also includes instructors and students.
Activity goal	Pedagogical purpose pursued by the distributed collaborative activity. This activity should try to verify a Software Engineering and/or instructional hypothesis.
Experience expectations	It refers to the potential benefits that academy and industry will obtain with the regular execution of this activity.
Hypothesis	The designer of the collaborative experience can define assumptions associated with the activity. It can be of two types: *1)* Interesting for the software industry or hypotheses that attempt to produce results (indications, findings, etc.) that are useful for software companies. *2)* Interesting for teaching software engineering. These are hypotheses that intend to identify how to produce useful results for participants in the teaching-learning process. If the designer defines a hypothesis the following section must be defined.
Experimental design activity	The activity should be adjusted within a rigorous experimental design, since it intends to obtain scientific findings in addition to the educational results.
Instructional model	For each stage of the collaborative teaching-learning model, it must be included: the duration, the responsible, the documentation to be distributed to the participating universities, the documentation that participating universities will deliver to the leading university, deliverables for /of students, etc.
Final evaluation	Define the method used to perform the final evaluation of the collaborative experience.
Analysis of results	Define the procedure that will be used for analyzing the results of the activity. These procedures may involve statistical analysis, observations, etc.
Requirements for students	This section presents the pre-requirements that the students must have to participate in the activity. Usually these pre-requirements are related to previous knowledge or academic level.
Requirements for communication	These are the requirements that must be addressed by the technological infrastructure supporting the distributed experience (e.g., communications platform, tools for supporting collaboration, software development tools and specific software).
Material support	It represents the papers, slides, assignments of the local and distributed practices, and any other supporting document required to perform the activity. This material should be provided by the leading university.
Performing date and time	The most probable date and time to perform the collaborative activities must be established. Such instance must be agreed between all participating universities. The time schedule for the activity should considering the time zone of every university.
Universities and institutional responsible	It specifies the acronyms of participating universities and the responsible of each one.
Expected results	The CODILA specification should include a list of products the team wants to obtain when completing the activity.
Annexes	The CODILA designer must specify the documents related to the experiment, such as: notes, practical exercises, charts and slides. He/she must also include references, in which s/he based the experience, or any bibliographic reference that can be appropriate for students and other instructors.

- Technological University of Panama, Panama (UTP).
- University of Cauca, Colombia (UCauca).

Track 2 (52 students)

- University of Quindío, Colombia (UQ).
- University of Chile, Chile (UChile).
- National Autonomous University of Costa Rica, Costa Rica (UNA).

All students were in the 8th or 9th semester of Computer Science or Informatics undergraduate programs. They grouped in distributed teams with three members each. The track 1 involved 18 teams and track 2 involved and 17 teams. Teams of track 1 were randomly formed, whereas in track 2 we used the students' psychological profile to form the teams.

The challenge to address by the team was the architectural design of a communication infrastructure for the government, were privacy and security are critical attributes. We also ask for software performance and maintainability which are contradictory requirements. The idea was to push the students to negotiate their individual solutions (that considered just one quality attribute) in order to obtain a balanced, integrated and robust design. Usually software quality attributes coexist in states of mutual tension. It demonstrates the need for high level of interaction, collaboration and negotiation between students designing the architectural specification. In the case of globalized projects, architectural decisions are largely made by distributed development teams.

The main goal of this collaborative and distributed architectural design was to simulate a real software distributed design process. The quality attributes addressed by the students were *Security*, *Performance* and *Mantenibility*. Each attribute is approached by one student, i.e., each team has a student who is expert in *Security*, other student in *Performance*, and so on.

During the collaborative exercise students realize the need to negotiate with their colleagues, to define a unified architecture that considers the three quality attributes mentioned above. Designs negotiation is not a trivial task due to each quality attribute possess a set of architectural tactics and drivers that may generate conflicts between quality attributes.

The lecture about design of software architecture was delivered synchronously from the University of Chile to the other participating universities using Microsoft LiveMeeting as a communication platform. Later, the students began working in a distributed way in order to specify the architecture for the assigned system. Distributed practices included three recordings in which the students shown their interactions. Each record was done for a specific deliverable and it had a specific purpose:

- *Recording 1*: First meeting of students, presentation of each student and the quality attributes to address by them.
- *Recording 2*: Each student exposed his/her qualities attribute assigned into the previous session. Each student exposed his/her proposal for the architectural design, based on the quality attribute assigned to him/her.
- *Recording 3*: In this session students negotiate to reach an integral design of the product. They also prioritized the features of the final architecture and discuss the technical characteristics of the solution (advantages and disadvantages).

Fig. 1 shows the lecture delivered from the University of Chile and Fig. 2 shows the participants in this theoretical session.

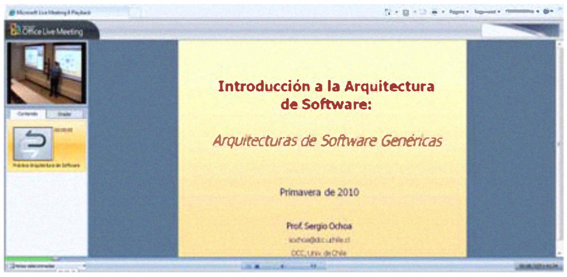

Figure 1. Software architecture lecture delivered from UChile

B. Obtained Results

The design created by the teams was evaluated and qualified with a score between 1 and 7. The minimal score for approval was 4. Analyzing the students' scores (Table II) we could hypothesize the use of an instrument (based on the psychological profile of the students) to form the teams had a positive impact on the teams' performance, since the Track 2 performed better than Track 1, although with a minimal difference. However, when reviewing the qualifications given by students for evaluating the work with colleagues, it is evident that teams formed randomly (i.e. Track 1) had better performance and interaction level than Track 2 teams. It occurred as result of a lack of interaction conflicts between students enrolled in 6 groups of Track 2. These conflicts were neutralized by the heuristic in charge of forming the teams, hypothesizing these conflicts were disruptive. However in the practice we saw that acted as triggered of discussions and negotiations among teammates.

Analyzing the students' opinions concerning their teamwork in both tracks, it is evident that all students have a favorably perception of the work done by their peers (Table III: the score goes from 1 to 5). Some particular situations were identified in the Track 1, where students from UCauca reported that their peers in PUC and UTP showed some difficulties in asking for help when they were problems. In the Track 2, students from UQ felt that UChile students do not work as a team. Moreover UChile students said that students from UNA did not answer emails on time and were engaged with the project.

National Autonomous University of Costa Rica (25 Students)
University of Cauca (9 Students)
Pontifical Catholic University of Chile (23 Students)
National University of San Juan Argentina (observers)
University of Quindio (19 students)
Technological University of Panama (18 Students)

Figure 2. Participants in the lecture

TABLE II .SUMMARY OF QUALIFICATION BY TRACKS AND UNIVERSITIES

TRACK 1		***TRACK 2***	
PUC	**3,91**	*UQ*	**4,95**
UTP	**4**	*UChile*	**5,31**
UniCauca	**4,67**	*UNA*	**3,56**
Average: 4,08		**Average: 4,35**	

TABLE III. RESULTS OF STUDENTS CO-EVALUATIONS

PUC	***4,1***	*UQ*	***3,5161***
UTP	***4,72***	*UChile*	***4,2143***
UniCauca	***4,6667***	*UNA*	***4,2308***

For measuring the satisfaction of participating students, we built a survey and we give it to the students for its completion. It survey contained 35 questions grouped by several dimensions related to social skills (teamwork, leadership,

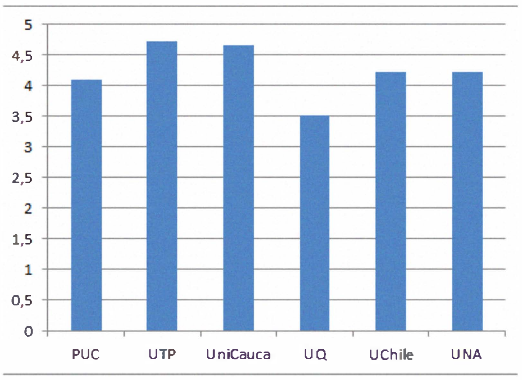

Figure 3. Values of co-evaluation of students in each university

negotiation and empathy) and experience (attitude and aptitude of the coordinators/contributors). Fig. 4 and Fig. 5 present the satisfaction level of each Track.

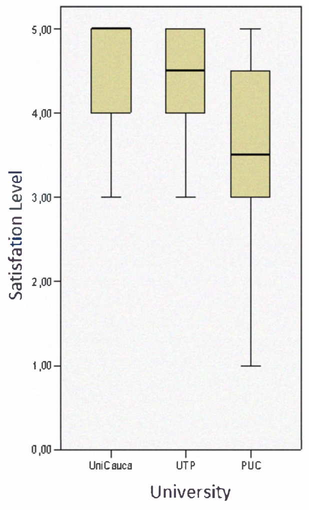

Figure 4. Satisfaction level in Track 1

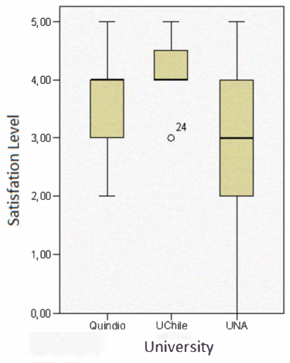

Figure 5. Satisfaction level in Track 2

In general, students from four of the six universities have expressed their satisfaction with the performed experience. In the other two cases, the researchers team found a lack of motivation in students, influenced by external factors to the experience, such parallel projects and low representativeness of the experience score into the final score for the course. In both cases, this was the first time that these universities were involved in this type of collaborative experiences.

C. *Learned Lessons*

Based on this architectural design experience and taking into account the collaborative results of previous experiments carried out since 2008 [10] [2] [11], the project team has identified some lessons, which are listed follow:

- *Team size*: The suggested number of members for a team is three. That number allows solving in case of conflicts and it also avoids free ridding attitudes because they are too visible in a small team.

- *Supporting material for the problem solving*: The instructors must provide appropriate supporting material that helps the students during the activity. Supporting material, particularly examples, ensures a better preparation of the students to conduct the experience. This material includes: guides, related examples, practical activities and case studies.

- *Motivation*: In order to achieve better students' engagement and successful results of the experiment, instructors should motivate the students. Some of the strategies that can be used are the following ones:
    - The participating institutions must assign a significant relevance to the experiments' score.
    - The participation of the entire course is mandatory these experiences. Moreover, such participation cannot be optional for the students.
    - Disseminate and clarify the importance of participating in these experiences. UQ students suggested the preparation of videos in which students that participated in CODILAs can tell to other students about their experiences.

- *Subjects to teach/learn*: The topics selected to teach/learn in a CODILA must have the following characteristics: i) students must have prior knowledge in the subject, ii) participating institutions must count on infrastructure to perform distributed practical activities, iii) the subject should be self-contained (i.e. as independent as possible from other subjects). Instructors must ensure that the activity will be relevant for the professional lives of the students.

- *Monitoring:* Based on previous experiences, the authors recommend performing evaluations that allow instructors monitoring the teaching and learning process.

V. Conclusions And Further Work

This article presents the results of applying a collaborative and distributed learning activity (CODILA) to teach software engineering practices in several Latin American universities. These collaborative activities are framed within a Latin American initiative that tries to establish a co-laboratory to teach, experiment and research on Software Engineering.

A CODILA involving the architectural design of a software communication infrastructure was presented and discussed. The obtained results show complex interactions between students, due to the experimental scenario proposed by researches. It reflects that designing software architectures implies an important challenge for software developers.

The results obtained in the experience were encouraging. Since the topic is complex to address by undergraduate students, the score were not high. However most students felt comfortable with the activity and they think that CODILAs can help them to address the challenges of a distributed software development scenarios. Instructors participating in the experience also felt highly comfortable with the activity.

Evidence exposed in this paper must be corroborated with the application of new experiences in this topic, and evaluating results obtained in the anonymous surveys. For further works we will measure the maturity level of previous experiences and we will propose a distributed course in globalized Software Engineering.

ACKNOWLEDGMENT

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 also thank to CINTEL Colciencias RENATA (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), LACCIR through the grant R1209LAC003 and RedCLARA through ComCLARA 2010 call.

REFERENCES

[1] J.C. Ellis, A. Steven Demurjian, J. Fernando Naveda, Software Engineering: Effective Teaching and Learning Approaches and Practices. IGI Global, 2009.

[2] C. Collazos, S. F. Ochoa, S. Zapata, M.I. Lund, L. Aballay, F. Giraldo, G. Torres de Clunie, CODILA: A Collaborative and Distributed Learning Activity Applied to Software Engineering Courses in Latin American Universities. The 6th Intl. Conf. on Collaborative Computing: Networking, Applications and Worksharing.Chicago, USA. 2010.

[3] M. Hawthorne, D. Perry, Software engineering education in the era of outsourcing, distributed development, and open source software: challenges and opportunities. In Proc. of the 27th Intl. Conf. on Software Engineering (ICSE'05). ACM Press, pp. 643-644. 2005.

[4] J. Whitehead, Collaboration in Software Engineering: A Roadmap. Proc. of Future of Software Engineering (FOSE'07). IEEE Press, pp. 214-225. 2007.

[5] G. Olson, J. Olson, Distance matters. Hum.-Comput. Interact. 15, 2, pp. 139-178. Sept. 2000.

[6] www.renata.edu.co.

[7] www.redclara.net.

[8] M. Hawthorne, E. Dewayne, 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.

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

[10] F. Giraldo, C. Collazos, S. F. Ochoa, S. Zapata, G. Torres de Clunie, Teaching Software Engineering from a Collaborative Perspective: Some Latin-American Experiences, Proc. of Workshops on Database and Expert Systems Applications, pp. 97-101. 2010.

[11] C. Collazos, S. Zapata, M. I. Lund, L. Aballay, S. F. Ochoa, F. Giraldo, C. Clunie, G. Torres de Clunie, R. Anaya, A Collaborative Model to Teach and Learn Software Engineering (In Spanish). Iberoamerican Conference on Higher Education. Asunción, Paraguay. 2010