Enhancing Collaboration Among Undergraduates in Informatics: A Teaching and Learning Process Based on Crowdsourcing

William Simão de Deus*, Heydi Miura Machado†, Renata Marques Barros‡, José Augusto Fabri§ and Alexandre L'Erario§
*Postgraduate Program in Informatics
†Undergraduate Program in System Analysis and Development
‡Undergraduate Program in System Information
§Computation Department
Federal University of Technology - Paraná
Cornélio Procópio – PR – Brasil
williamd@alunos.utfpr.edu.br, heydimiuramachado@gmail.com, renata_mbarros@outlook.com, {fabri,alerario}@utfpr.edu.br

***Abstract*—Crowdsourcing (CS) is a development model in which small activities are carried out through the collaboration of participants. Many organizations are developing their products through CS to parallel activities, reduce costs, and employ specialists. These factors have boosted the area of informatics, creating a new paradigm for the development of software. However, despite the widespread application of CS in informatics, the literature on teaching and learning CS for undergraduates is still very incipient, and the informatics courses aimed to maximize only the teaching of traditional development processes (Agile, distributed, etc). With this in mind, this study was developed to present the following contributions: (i) a personalized process of teaching and learning CS to the undergraduates in computing courses, and (ii) demonstrates the configuration that a classroom and/or computer lab should possess to generate a CS teaching and learning environment. To accomplishment this study, experimental trials were conducted with undergraduates in different courses in informatics. In conducting the trials, classrooms and computer labs at a university were set up simulating barriers found in CS. As results, all undergraduates have accomplished and reached the ultimate goal through CS, and the process of teaching and learning allowed for the enhancement of several factors, such as teamwork, integration, and support.**

I. Introduction

The term crowdsourcing (CS) was coined by Howe [1] after observing a change in the business model of several organizations. The problems and activities of these organizations were exposed to groups of people who contributed their own innovative solutions without being characterized as outsourcing, but a distributed model of contribution [2]. This model was driven by the development of new technologies, enabling the use of CS in several areas such as text translation, image labeling, software development, and marketing [3]. Along these lines, CS improved engagements and established a collaborative link between the public and organizations [4].

Presently, CS has become massive and has created a new trend in the software development model based on collaboration and innovation [5]. Thus, this new model encouraged the adoption of CS in software projects; however, educational institutions are still not preparing undergraduates for this type of development, using collaboration and the exchanging of experience [6]. Educational institutions focus on teaching the concepts of traditional models, such as Agile or distributed development to educate undergraduate students, minimizing their attention on the actuality of CS in undergraduate courses. According to Karataev and Zadorozhny [7], this prerogative can be sustained by the fact that CS is still a development model considered new, complex, and extremely difficult to be applied to undergraduate learning.

According to Bosnic et al. [8], another costly factor in teaching and learning CS is the complexity of configuring the participants in the classroom. The authors state that a distributed teaching and learning process can become costly or even impossible to configure because of the dispersion. In addition, CS also demands content creation and feedback from undergraduates to check progress, increasing the teacher's workload [9].

Nowadays, contributions to the teaching and learning of CS can be found in literature through technological means, such as on-line tools and tutors [7], [6]. Yet, these works present a set of limitations on the setting of CS for students, representing a gap for applying CS in the classroom. Collaboration is exploited minimally, besides not demonstrating the configuration processes for the execution of teaching and learning CS. With this scenario in mind, this paper seeks to exhibit a substantial gain for the configuration, teaching, and the learning process of CS for students. The contributions of this study encompass two domains: i) a process for the environment and presentation of CS for students and ii) present the configuration that a classroom must support to simulate a CS environment. The validation method adopted for this research was the use of experimental trials with a group of undergraduates with backgrounds in System Analysis, Software Engineering, and Computer Engineering courses. The results demonstrated that CS can be implemented for teaching and learning of students

978-1-5090-5920-1/17/$31.00 ©2017 IEEE

through the application of a process organized in 3 levels. Among the main highlights of the CS process, communication, teamwork, and collaboration were praised.

The rest of this study is organized in the following sections: Section II, introduces the literature review about CS. Section III, presents the works that provide synergistic results to this study. Section IV presents the original process and its modeling for the CS environment. Section V, describes the validation methods and procedures. Section VI, demonstrates the results obtained. Finally, Section VII, presents the final conclusion and limiting factors of this study.

II. Crowdsourcing

The term CS, presented by Howe [1] in 2006, defined a new business model. This model referred to the technological advances that allow activities from organizations to be carried out by a crowd of people. However, without characterizing as a type of outsourcing, but using an open call to perform a service or task for a heterogeneity network of participants [10].

According to Howe [11], the amount of knowledge and talent dispersed between a set of people, through CS, exceeds individual capacity, evidencing the use of knowledge as a way to create, disseminate, validate, and innovate ideas [12].

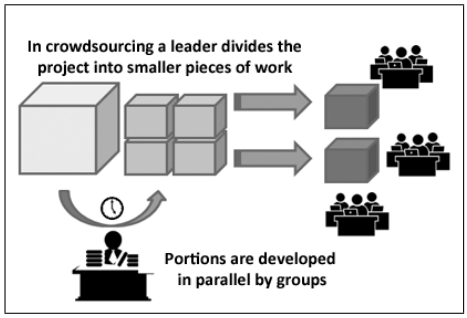

Fig. 1. Crowdsourcing Process Inpired by Suganthy and Chithralekha [5]

In order to facilitate the understanding of the CS process, the approach proposed by Suganthy and Chithralekha [5] was used in the application of CS for software development (Figure 1). The process of CS employs a leader, who is responsible for selecting the work that will be developed by CS. This leader divides the workload into small portions and distributes them to groups of people interested in its development. At this point, CS is innovative compared to other traditional models. Rather than the leader selecting a group, online workers, who show an interest in the development of this activity (whether by financial reasons, status, or professional appeal), send contributions to help develop this portion of work [10].

III. Similar Works

The literature has a set of works with objectives and synergistic results related to this research, so, three works that present similar contributions were selected. The first similar work (SW1) was presented by Karataev and Zadorozhny [7], introducing a model that allows the creation of educational content in small lessons. These lessons are adaptive and social, similar to the CS concept. As a result, the authors demonstrated that collective learning experiences can be efficiently shared with the use of small lessons.

Challenges related to the efficiency of undergraduate learning were addressed by Anderson [13] in a second similar work (SW2). In this study, the author used mechanical social learning by combining communication between students and mentors. With this, a CS concept, capable of connecting and making stored information available for student use, was generated.

CS was also analyzed jointly by researchers affiliated with universities in the United States, Australia, and China [14] in the third similar work (SW3). The authors presented a self-taught learning paradigm that uses the complementary knowledge of each participant to add to the knowledge of the crowd. The results showed that the self-taught learning process in CS minimized error and costs.

As it was presented, CS has been studied for application for the teaching and learning. However, the works analyzed present a set of limitations on CS configuration. To demonstrate the comparison of similar works, Table I presents the study referenced, CS method used, and limitations in each approach:

TABLE I
Analysis between similar works

Ref	Crowdsourcing Concept	Limitation
(SW1)	Teaching based in small lessons	Setting up the CS environment
(SW2)	Communication in group	Unavailability of mentors
(SW3)	Knowledge of the crowd	Setting up the CS environment

The use of CS through small activities for a crowd was discussed in the work of Karataev and Zadorozhny [7]. However, the configuration of the teaching and learning environment was not presented in the work. This limitation was also found in the Fang et al. [14] approach. Finally, the unavailability of knowledge-bearing mentors was found by Anderson [13]. The teaching and learning process proposed in this study supplies the perceived limitations. Furthermore, it discards the availability of mentors since the public is dynamic and it also offers the configuration contribution of a CS environment.

IV. Background

The creation of the CS teaching and learning process was developed from the traditional process of teaching and learning. Therefore, the original process was adapted for CS reality, with the configuration, dispersion, and necessaries tools.

A. Original Process

The original process was developed by Fabri et al. [15] to present to students, who were enrolled in computing courses, teachings related to software process and management of software projects. This method was selected to base the current process on the following characteristic:

i) The process was evaluated in an academic way (using graduate students) and in an industrial way (using professionals in the software development field). With these factors, the results obtained demonstrated high success.

According to Fabri et al. [15], the process was conceived in three multidisciplinary stages carried out to assist understanding a particular topic:

- In the first stage, the students are acclimated with the theme through theoretical presentations, being the teacher's role to select and build the support material to present to them.
- In the second stage motivation is given to the students for the practical application of the theory previously presented. At this stage, the teacher selects and composes the objectives that should be evaluated.
- Finally, the validation process is verified through the development of students' skills. For the validation process evaluation tests should be conducted.

The original process was configured in interconnected and sequential steps, as shown in Figure 2.

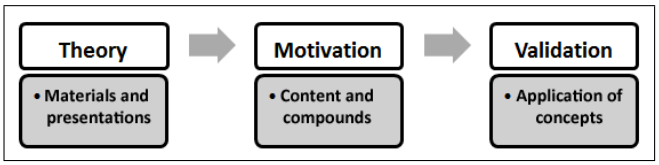

Fig. 2. Teaching and Learning Process Proposed by Fabri et al. [15]

The presented process was based on the implementation of the three steps described above. Much of the work to be done is directed at the teacher, such as the preparation of the material and the evaluation procedure. To adapt to CS reality, a new step for the environment setup was created and with this, the teacher's workload was reduced.

B. Teaching and Learning in Crowdsourcing

To apply the teaching and learning process in CS, the original process of Fabri et al. [15] was remodeled as follow:

- 1. The first step has not been modified, but the teachers can apply successful examples of CS to solve challenges related to the focus of their teaching. These examples are found in literary works, such as the use of CS in social networks [16], geographic mapping [17], or marketing [18], etc.
- 2. Configuration: This step has been added to allow students to experience an environment that simulates the CS reality. In this way, variables capable of controlling the communication and the dispersion were added. To organize the configuration of the CS environment, three levels were created:
  - **Level I:** Students are in the same room but in different groups. The groups can only communicate via electronic chat. It should be noted that at this level, there is no temporal dispersion, and therefore the CS simulation for students is low. Also, different groups can have the same activity. In order to improve the CS simulation and the collaboration, the tasks must be dependent. A practical example is a creation task (group A) and the second task is the validation of the first task (group B).
  - **Level II:** At this level, the undergraduate students are separated into two or more groups. Each group is allocated in a classroom. Thus, personal communication between students of different groups is impaired and will only happen via electronic chat, but the communication between the students of the same group can be personal or via electronic chat. In this scenario, CS is considered intermediate. This level provides a much closer proximity to CS compared to the previous level. Thus, it is possible to distribute the same activity to both separate groups in separate rooms and to conduct a dispute over which one finishes the task first.
  - **Level III:** At the last level, students are dispersed both temporarily and physically. Such an approach ensures that everything is developed outside the classroom, and communication takes place only via electronic chat. As students are dispersed, each one of them becomes a unique entity. Therefore, several activities can be created and distributed randomly among themselves.
    This is the most complex level and the one that most closely approximates to the reality of CS since the students face real problems that CS poses, such as communication problems, asynchrony, and deadline. Collaboration is necessary for the execution of the activities, and students should understand and apply this concept to conclude them. Until the previous levels, each group formed a unique entity operating for the development of the task. However, with the physical and temporal disparity, each student becomes a unique entity and the results depend entirely on the contribution of all.

  In all levels, a platform equipped with communication chat should be adopted to provide communication amongst students. This way, CS platforms, electronic chats, or information bases can be used. It should also be noted that information and communication logs are important for the validation process of level II and III, as the teacher is not able to be present with groups/students during the execution.
- 3. Motivation: The theoretical motivation was treated to select an exercise capable of developing CS activities. As presented in the first stage of this process, there are several examples that can be simulated and/or adapted to be conducted with students. The activities should enable students to use group knowledge, collaboration, and communication to replicate CS characteristics.
- 4. Validation of teaching: To verify if the students were able to capture and apply the knowledge of CS, the

results of the activities should be evaluated. A simple evaluation method verifies whether the goal outlined has been successfully achieved or not. If the students reach a solution using their combined knowledge coupled with the group collaboration it's likely that students have learned the CS work method. It is also proposed that the log chat communication is examined to verify if there was the collaboration between the students.

The main modifications in the original process deal with the insertion of a new stage: the configuration and the remodeling of the Motivation stage to CS reality and the teacher's workload reduction. The composition of the teaching and learning process for CS is presented in Figure 3.

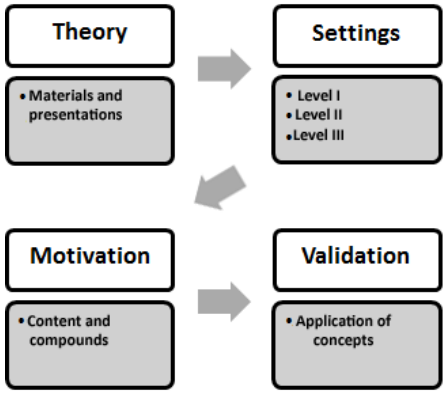

Fig. 3. Teaching and Learning Crowdsourcing Process

After reshaping the process to the reality of CS, the validation process was started. In this way, three experiment trials were conducted to verify the performance of the process on the CS reality. Each trial was designed to test the process configuration levels.

V. Validation

To validate this work an experimental approach was adopted. For this, the protocol, proposed by Wohlin et al. [19], for experimental trial was selected.

A. Characteristics of Problems Studied

The CS teaching and learning process has three levels of configuration. With this, three experimental trials were conducted to validate the process. For level I and II, a CS initiative belonging to Federal Technological University - Parana was selected. This initiative uses Legos Mindstorms robots for teaching and learning programming and project management for students in Systems Analysis, Software Engineering, and Computer Engineering courses. For Level III, a Systems Analysis group was selected to perform the Requirements Engineering stage of a software project using CS.

Presentations were given to the participants of each experimental trial and in the end of them, a CS challenge was exposed. Each challenge was set up to run according to the stages of the CS teaching and learning process. To validate and detect collaboration, the analysis of the communication logs and observation of the participant's behavior were analyzed.

- H_0 Participants did not reach the goal of each experimental test, demonstrating that CS cannot be systematized in a process of teaching and learning.
- H_1 Refutation of H_0.

The hypothesis test was based on the following assumption: if the participants reached the goal of the challenge using collaboration, the teaching and learning of CS would have been achieved.

The challenge of level I was to build a Lego Mindstorms robot using group knowledge and collaboration. The Level II challenge was to drive a Lego Mindstorms robot with separate teams in two distinct environments. The challenge of level III was to carry out the specification and documentation of the software project requirements using CS.

B. Methods and Procedures

Level I: To validate the first level, the CS initiative based on LEGO Mindstorms was used. The Mindstorms have a widespread use in educational environments as a robotic kit that allows teaching and learning in areas such as physics, electronics, robotics, and computing [20]. In the first experimental trial, undergraduate students from Systems Analysis and Computer Engineering courses were selected to verify the CS teaching and learning process.

In the theoretical presentation, the students were introduced to the LEGO Mindstorms initiative. There were presentations about the concept, use, and programming of robots. About 20 minutes was spent introducing all the basic knowledge.

The students were organized into two groups. The groups were independent and could not exchange information. Each group received a kit containing the parts of the Lego Mindstorms robot.

As motivation, the following objective was adopted: to verify if the students were able to use collaboration between the group participants to build a robot capable of moving autonomously.

In the evaluation process, the authors of this study verified if the two groups reached the goal, evidencing collaboration. Both groups employed less than 50 minutes to assemble the robot.

To synthesize the protocol of the experiment applied to the process of teaching and learning CS, the steps and their compositions are presented in Table II:

TABLE II
Evaluation Procedure in the Level I

Step	Composition
1. Theoretical Presentation	Worked during a traditional lesson with slide show and explanation
2. Configuration (Level I)	Same classroom, but the students were organized into two groups
3. Motivation	Build a Lego Mindstorms robot capable of moving autonomously
4. Validation	Check whether mounted robots could move

Level II: In the second level the undergraduates were also organized into four groups. Two groups stayed inside the classroom and the two others, in a computer lab. It is important to note that no participant had previous knowledge of Lego Mindstorms. Thus, the time constraint and lack of knowledge in Lego made it impossible to attempt any Internet searches to solve the challenge quickly.

The students were also from Systems Analysis, Software Engineering, and Computer Engineering courses. The groups were formed randomly, uniting students from different courses.

Groups A and B were allocated in a laboratory equipped with computers and internet access. Group C and D were placed in a classroom without internet and computers. The objective of this experiment was to verify if the teaching of CS could be applied in two different environments achieving similar results.

Both groups received the same task: to program the Lego software to drive the robot in a circuit. But only the laboratory had the physical circuit for testing. In the normal classroom, there was only one image of the circuit. In this mode, each group should collaborate with its participants to drive a Lego robot through the circuit.

The total time of the experiment was about 1 hour and 45 minutes, and both groups reached the same objective. Table III shows the contextualization of the second experiment trial:

TABLE III
EVALUATION PROCEDURE IN THE LEVEL II

Step	Composition
1. Theoretical Presentation	Worked during a traditional lesson with slide show and explanation
2. Configuration (Level 2)	Four groups in different rooms: tow groups allocated in a computer lab with computers, internet access and scenario. Anothers groups allocated in a classroom without internet and computers
3. Motivation	Make the Lego robot complete a circuit
4. Validation	Check order in Lego robot

Level III: To test this level the undergraduates of System Analysis formed a single group that worked collaboratively using the group's knowledge to elaborate and document the requirements of a software project.

All participants were allocated on a CS platform. This platform had an electronic chat for communication and a historical basis for project documentation. Participants were invited to register requirements on the platform. The requirements should have an identifier and its description. After registering the requirement, other participants could assist in the elaboration by adding details and/or developing the documentation. As all students were working on the same project, they should merge and update the documentation, as well as make constant communication with the participants to disseminate the progress of the project.

Every participant was enrolled in at least one requirement on the platform. All requirements were documented in a standardized way through an identifier and its description. We invited students to build test cases to verify if the requirements could be implemented from the participants' perspective. A test case represents a set of descriptions used to perform the test of a requirement, and it can only be developed if the participants understand its use and expected results. The contextualization of the experimental trial is presented in Table IV.

TABLE IV
EVALUATION PROCEDURE IN THE LEVEL III

Step	Composition
1. Theoretical Presentation	Worked during a traditional lesson with slide show and explanation
2. Configuration (Level III)	A group with physically distant participants. The participants used the communication through an electronic chat and only met in person twice a week.
3. Motivation	Elicitation of requirements
4. Validation	Test Cases

VI. RESULTS AND DISCUSSIONS

A. Level I:

The two groups (named group A and group B) succeed in completing the project, both of them completed the challenge of assembling a robot with autonomous movement using collective intelligence through CS. In group A, the assembly was completed in less than 40 minutes of starting, and after 167 interactions in electronic chat. Group B, spent about 44 minutes performing the experiment and used 103 interactions.

In the individual analysis of the electronic chat, it was realized that each group adopted the same set of practices for the experimental trial execution. The crowd, after giving some suggestions on how to mount, they decided whether the assembly was correct or not and if it would be better to adopt a new strategy.

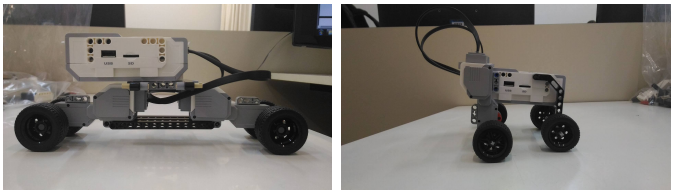

Fig. 4. Final structure of the Group A robot (left) and Group B (right)

The final product obtained by performing the experiment to validate the level I process was presented in Figure 4. In addition to the groups successfully completing the assembly through CS, characteristics such as communication and collaboration were evidenced in this level. Finally, both groups concluded their projects in less than 1 hour.

B. Level II

In the second experiment, the students were separated into four groups. Two groups remained in a common classroom (no computers and internet). The other two groups were allocated in a computer lab (with computers and internet access). The two groups that were located in a classroom were tasked with programming a Lego Mindstorms robot using only the

students' knowledge, without using the internet for research. While the two groups allocated in the computer lab had Internet access and the Lego Mindstorms robot to test. The robot had a pre-determined circuit (which can be seen in Figure 5) with a set of challenges.

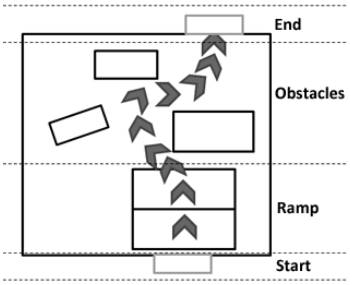

Fig. 5. The Challenge Circuit

The circuit was only available for groups in the computer lab. In this way, it was tested whether collective collaboration among group participants could accomplish the same task in different environments.

The assembled circuit had a start line, in which the robot should start its course. The first obstacle to be overcome was a ramp, and there were three other obstacles that the robot would have to deflect. When identifying the finish line, the robot should stop. It was observed that groups located in the traditional classroom completed the activities before the groups in the laboratory. This led to the believe that CS collaboration has become even more effective because it represents human collaboration.

It was also realized that the scenario in the classroom and lab influenced the way groups worked. The groups that were working in the lab spent a lot of time looking for solutions on the internet, while groups from the traditional classroom have employed their time debating ideas and solutions. The number of interactions is detailed in Figure 6:

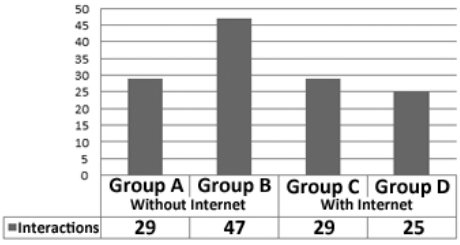

Fig. 6. Number of Interacions per Group At Level II

CS approach could be adopted in different scenarios, however, the internet was a distraction that influenced the groups' results. Nevertheless, as it can be verified, in both scenarios the process of teaching and learning CS made possible to achieve the final objective

C. Level III

In the last trial, the participants were invited to analyze and develop a portion of software production through CS. In this activity, undergraduates were grouped into a single entity that should work collectively.

Undergraduates were responsible for developing small portions of work on the same project. The project itself deals with a web system, responsible for an integration environment between clients and service providers.

Each student should submit a contribution that was evaluated by the researchers. After the researchers' approval, a second student should select the activity and develop a test case for it. If the activities were approved by the researchers and complemented with a test case, the contributions were counted in the experimental test.

The undergraduates should develop their activities at different times. To ensure this cohesion, the contributions had a log dating the day and time of submission. In addition, undergraduates were encouraged to submit their contributions in distinct environments to simulate CS with great similarity. The process executed at level III allowed the mirroring of activities. In general, some graduates undertook the project analysis stage, while others were responsible for a test. It was interesting to note that despite the physical and temporal dispersion, undergraduates operated as a single and rational entity, performing small portions of work in different environments. One of the contributions provided by the experiment trial at level III is presented in Figure 7. At Level III, there was a very important detection: communication. The participants talked and debated the project on a daily basis using electronics chat. Thus, there were submissions on almost every day of the experiment. It should be noted that this experiment was carried out within a period of a month and with about 20 participants.

D. Experimentation Highlights

With the execution of the three experimental trials and their respective analysis, a key CS factor defined as collaboration was evidenced in the results. The collaboration took place mainly in three areas: teamwork, integration, and support:

- **Teamwork:** It represents work focusing on the same goal. In the three experimental trials, the groups were able to solve and develop the proposed activities that required the union of the team. In the experimental tests of level I and III, it was impossible to reach the final goal without the support of the team due to the configuration. While at level II, teamwork was assessed as a facilitator. In the experiment at the level I, one of the participants wrote in the chat, *"also send me suggestions to get the job done faster"*. Other participant stated, *"Is everyone in the group understanding our strategy?"*, and this was answered with, *"yes, but return the current state of the strategy to identify how we can optimize"*.
- **Integration:** It represents the capacity of the group to operate with the interaction of all participants. In the experimental trial of level II, social integration allowed undergraduate student groups to solve the challenge quicker than groups that used the Internet. The researchers were allocated in the rooms and laboratories used during the

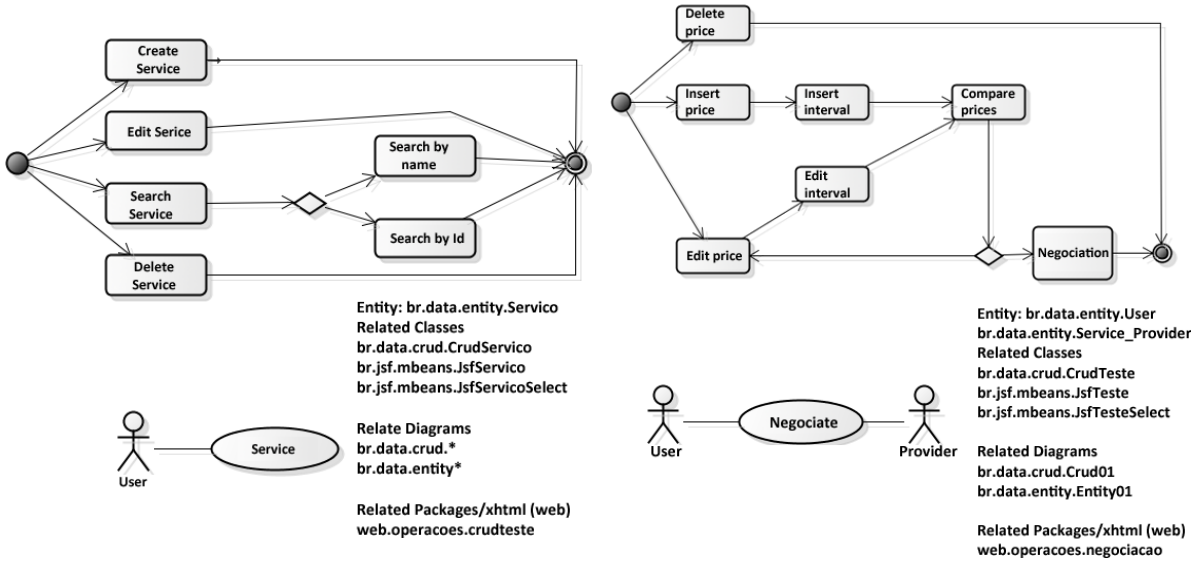

Fig. 7. The image represents use case diagrams and fluxograms generate by graduate students using crowdsourcing

level II trial, and it was observed that the students that were in the computer laboratories dispersed their attention with the Internet, without effectively contributing to the ultimate goal.

With the undergraduates of the traditional classroom, the situation was different: they interacted with each other to solve the challenges. With this, all students focused on and offered contributions to the project rather than dispersing. In the level III experiment, the historical base used registered the contribution of all the participants for the submission of contributions.

- **Support:** It represents the help offered among the participants of each group. At Level I, several examples of mutual support among undergraduates were identified. An example occurred when the group wrote, *"look for the largest piece to assemble the base of the robot"*, and one participant complemented that with, *"the piece that we refer to is a single piece in the format of the letter 'H'"*. In the Level II, while one participant displayed nervousness to complete the goal, several participants in the group supported a positive message indicating that they would be able to finish on time. Level III, undergraduates with more experience in online repositories performed a merge operation between activities for the entire group to work with up-to-date activities.

In addition to the highlights previously cited, the results obtained through this study deepened CS teaching and learning and helped to solve the main gaps in the area. As has been seen, similar works on teaching and learning CS did not demonstrate the necessary configuration to apply CS among undergraduate students. Thus, three levels were identified and configured for simulating CS in a classroom or computer lab. Furthermore, another gap persisted regarding the unavailability of mentors or teachers, this was suppressed with the use of electronic chats or logs that were audited at the validation stage. Although our experimental tests focused on only three distinct courses of computing, there is no impediment to reapply the process to other undergraduate courses.

In addition to the previously mentioned areas of collaboration, it was also highlighted in the statements that the students approved the CS teaching and learning process. There were messages that the experimental trials *"served to bring all the students together"* and that *"classes and content became more didactic with the participation of all"*.

Due to the support received by the participants, it is believed that other courses in computing, such as graduation in Engineering or Computer Science, can apply this CS teaching and learning process to enhance collaboration.

Collaboration provided by CS can also be understood as a factor in avoiding teamwork conducted by a few students. One of the great challenges of teamwork is identifying the students who have effectively contributed to the group. With the CS teaching and learning process discussed in this study, the teacher can operate as an observer at the level I and use mechanisms, such as historical base records and log analysis of communication, to filter participants who have contributed effectively and identify the contribution.

Teaching and Learning Process Crowdsourcing				
		Collaboration		
		Teamwork	Integration	Support
Levels	I	Totally	Weakly	Totally
	II	Partial	Totally	Totally
	III	Totally	Totally	Totally

Fig. 8. Highlights obtained

As shown in Figure 8, all collaboration features were fully detected at level III. Level II has strongly demonstrated integration and support, while teamwork has been partial

because the groups in the room with access to the Internet have dispersed their attention. Finally, Integration was weak at level I, this is due to the fact that some students did not interact during the trial.

VII. Final Considerations

CS is still a relatively new model, but it has a variety of application in diverse areas. More incisively, CS has been explored as a model for software development in the IT field. However, higher education institutions still adopt traditional development models for the teaching and learning of these undergraduates.

The field of CS teaching and learning has not properly been explored, gaps in empirical studies and/or studies that demonstrate the configuration of a classroom needs to support CS. Supported by this scenario, a CS teaching and learning process was modeled. The process presented in this study has a set of steps and descriptions that must be configured to support CS in the progression of computer courses.

To validate the CS teaching and learning process, three experimental trials were performed with undergraduates of different computing fields: Systems Analysis, Computer Engineering, and Software Engineering. As a result, it was detected that the undergraduates were able to use CS to perform different activities, from the assembly of robots to the analysis and structure of web based projects.

The collaboration of CS was exalted among participants, highlighting teamwork, integration, and support. It was realized that the process adapted to the different environments and levels to which it was submitted, resulting in a great collaborative link between undergraduates. Therefore, it is believed that H_1 was considered true since the teaching and learning of CS were systematized and emphasized in the experimental tests conducted.

A. Limitations and Future Work

An empirical study demonstrates a set of limitations. In this study, it was realized that although using electronic chat, it is virtually impossible to block communication between scattered students in the same classroom. It was also perceived that the trials were performed in only three computing courses, but the computing area has a greater variety of courses. Finally, all experimental trials treated the students as crowd participants. In this way, the activities were defined or managed by the teacher who acted as project leader.

An important contribution to future work is to consider an approach focusing on small portions of work. Questions regarding how small portions can be used in other courses represent an area of strong academic interest.

References

[1] J. Howe, "The rise of crowdsourcing," Jun 2006. [Online]. Available: https://www.wired.com/

[2] L. Machado, J. Kroll, S. Marczak, and R. Prikladnicki, "Breaking collaboration barriers through communication practices in software crowdsourcing," in *2016 IEEE 11th International Conference on Global Software Engineering (ICGSE)*, Aug 2016, pp. 44–48.

[3] A. Baruch, A. May, and D. Yu, "The motivations, enablers and barriers for voluntary participation in an online crowdsourcing platform," *Computers in Human Behavior*, vol. 64, pp. 923 – 931, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0747563216305295

[4] C. E. Barbosa, V. J. Epelbaum, M. Antelio, J. Oliveira, J. A. Rodrigues, S. P. J. Medeiros, and J. M. de Souza, "Conceptual crowdsourcing models for e-learning," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2014, pp. 1121–1127.

[5] A. Suganthy and T. Chithralekha, "Application of crowdsourcing in software development," in *2016 International Conference on Recent Trends in Information Technology (ICRTIT)*, April 2016, pp. 1–6.

[6] A. Kulakli and S. Mahony, "Knowledge creation and sharing with web 2.0 tools for teaching and learning roles in so-called university 2.0," *Procedia - Social and Behavioral Sciences*, vol. 150, pp. 648 – 657, 2014, 10th International Strategic Management Conference 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877042814051337

[7] E. Karataev and V. Zadorozhny, "Adaptive social learning based on crowdsourcing," *IEEE Transactions on Learning Technologies*, vol. PP, no. 99, pp. 1–1, 2016.

[8] 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*, ser. CTGDSD '11. New York, NY, USA: ACM, 2011, pp. 26–30. [Online]. Available: http://doi.acm.org/10.1145/1984665.1984671

[9] D. S. Weld, E. Adar, L. Chilton, R. Hoffmann, E. Horvitz, M. Koch, J. Landay, C. H. Lin, and M. Mausam, "Personalized online education—a crowdsourcing challenge," in *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012, pp. 1–31.

[10] N. Archak, "Money, glory and cheap talk: Analyzing strategic behavior of contestants in simultaneous crowdsourcing contests on topcoder.com," in *Proceedings of the 19th International Conference on World Wide Web*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 21–30. [Online]. Available: http://doi.acm.org/10.1145/1772690.1772694

[11] J. Howe, *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*, 1st ed. New York, NY, USA: Crown Publishing Group, 2008.

[12] O. Alonso, D. E. Rose, and B. Stewart, "Crowdsourcing for relevance evaluation," *SIGIR Forum*, vol. 42, no. 2, pp. 9–15, Nov. 2008. [Online]. Available: http://doi.acm.org/10.1145/1480506.1480508

[13] M. Anderson, "Crowdsourcing higher education: A design proposal for distributed learning," *MERLOT Journal of Online Learning and Teaching*, vol. 7, no. 4, pp. 576–590, 2011.

[14] M. Fang, X. Zhu, B. Li, W. Ding, and X. Wu, "Self-taught active learning from crowds," in *2012 IEEE 12th International Conference on Data Mining*, Dec 2012, pp. 858–863.

[15] J. A. Fabri, A. L'Erario, R. H. C. Palácios, and W. Godoy, "Applying mindstorm in teaching and learning process and software project management," in *Frontiers in Education Conference (FIE), 2015. 32614 2015. IEEE*, Oct 2015, pp. 1–8.

[16] E. O. Svee and J. Zdravkovic, "A model-based approach for capturing consumer preferences from crowdsources: The case of twitter," in *2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*, June 2016, pp. 1–12.

[17] J. Ingensand, M. Nappez, S. Joost, I. Widmer, O. Ertz, and D. Rappo, "The urbangene project: Experience from a crowdsourced mapping campaign," in *2015 1st International Conference on Geographical Information Systems Theory, Applications and Management (GISTAM)*, April 2015, pp. 1–7.

[18] C. H. Chu, M. Wan, Y. Yang, J. Gao, and L. Deng, "Building on-demand marketing saas for crowdsourcing," in *Service Oriented System Engineering (SOSE), 2014 IEEE 8th International Symposium on*, April 2014, pp. 430–438.

[19] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[20] C. B. Nielsen and P. Adams, "Active learning via lego mindstorms in systems engineering education," in *Systems Engineering (ISSE), 2015 IEEE International Symposium on*, Sept 2015, pp. 489–495.