Transparently Teaching in the Context of Game-based Learning: the Case of SimulES-W

Elizabeth Suescún Monsalve, Julio Cesar Sampaio do Prado Leite
Departamento de Informática
Pontifícia Universidade Católica do Rio de Janeiro
Rio de Janeiro, Brasil
emonsalve@inf.puc-rio.br, http://www-di.inf.puc-rio.br/~julio/

Vera Maria B. Werneck
Departamento de Informática e Ciência da Computação
Universidade do Estado do Rio de Janeiro
Rio de Janeiro, Brasil
vera@ime.uerj.br

***Abstract*— This work presents a pedagogical proposal, in the context of game-based learning (GBL), that uses the concept of Transparency Pedagogy. As such, it aims to improve the quality of teaching, and the relationship between student, teacher and teaching methods. Transparency is anchored in the principle of information disclosure. In pedagogy, transparency emerges as an important issue that proposes to raise student awareness about the educational processes. Using GBL as an educational strategy we managed to make the game, a software, transparent. That is we made the inner processes of the game known to the students. As such, besides learning by playing, students had access to the game design, through intentional modeling. We collected evidence that, by disclosure of the information about the design, students better performed on learning software engineering.**

***Index Terms*— Transparency, Games-based Learning, SimulES-W, Pedagogy.**

I. Introduction

Theories of learning have been developed for a long time. The educational process is very complex and cannot support drastic solutions as has been demonstrated throughout history. However, in recent decades our society has been exposed to a rapid transformation as a result of the introduction of new technology. This change is being reflected in social and personal scopes. In the same way, this impact has been felt in the way students learn and expect from classes. As such, education has evolved from the traditional teaching model for more dynamic models, which are more attractive and effective. In addition, the "traditional teaching method is based on memorizing theoretical concepts presented by the teacher in an abstract manner, dissociated from practical reality" [8]. For that reason, methods based only on the accumulation of knowledge have become obsolete, since, with introduction of new technologies, there is an expectation on learning by doing and experimenting.

In [16] the authors present an analysis about traditional teaching based on the Galperin Approach. They argue that traditional teaching gives the teacher tasks as: explain, demonstrate and elaborate concepts, and if possible, he/she evolves the concepts to a level of education. In the same way, the learners must follow the rationale of the teacher. In addition, learners must present his/her doubts, memorize information and learn to use formulas that explain the use of concepts in certain situations. Galperin [8] emphasizes that this process compromises the quality of education because the concepts are presented in an abstract way and it generally does not reflect reality. The author also mentions that the teacher could use examples to show the practical application of concepts, however learners will remain in the status of "observers". In addition, the format of tasks is usually done in an automated way. In this perspective, the traditional teaching remains a slow, exhausting and, usually, without motivation process.

On the other hand, dynamic models are being proposed such that teaching is performed in a more practical way, investing in "the development of the ability by means of discovery" [8]. Thus, the teacher has the responsibility to encourage learners to observe, arouse their curiosity and challenge them to investigate and find examples. So, a learner starts his/her own experience and interferes in his/her own learning. In a similar manner, familiarity with problem-situations can develop the ability to recognize certain situations and how to act on them. For that reason, learning by doing is seen as a possible solution [16]. Our work departs from the idea that learning by doing is a positive strategy. As such, we also believe this strategy contributes to a leaner´s motivation. Moreover, a situation-problem approach allows the usage of these concepts and allows that their influence on the action context where they are implanted be observed.

In practice, it is possible to show how concepts of software engineering are taught and it can be demonstrated by the use of a mix of lectures and small practical projects [28, 29 and 30]. However, these projects do not simulate situations of big and complex systems [24]. Software engineering education is challenged to produce highly qualified people, with mental flexibility to adapt to changes caused by the introduction of new technologies. One of the problems of traditional software engineering education is that it gives a lot the importance to the theoretical content, and this is often problematic, since it does not necessarily imply that the learner is able to apply this content to real life situation. One possible strategy [33, 34 and 36] to mitigate this problem is to use real software projects, and as such implementing “learning by doing”. However, providing the proper scenario for this type of strategy is costly and sometimes hard to implement.

A possible strategy to replace real software projects is the use of Game-Based Learning. According to [38] the term *games-based learning* in general “refers to activity to engage and hold learners in focus by encouraging them to participate during the lesson through game-play”. Actually, game-based learning (GBL) addresses teaching in a dynamic way and has been successfully used as a support tool in several areas, Connolly et al in [39] presents evidence, by a large systematic literature review, of the positive results in using computer games and serious games, including software engineering [31 and 32].

GBL has been shown to be an alternative to software projects in providing a situation-problem environment without the costs and difficulties of conducting a real software project [35 and 37]. One of these alternatives is SimulES-W [17], a collaborative platform that implements a simulation of a card base game where the goal is to produce specific software, with given characteristics. However, an approach to teaching software engineering should not forget, in any case, the importance of knowledge, but it should be addressed in a dynamic way, bringing at the same time knowledge, skills, abilities, and values and raising awareness in the learner about their own learning process., Taking this in consideration we have added the principles of transparency to the GBL (Simules-W) strategy as to inform the students about Simules-W design [13].

II. Transparency

Adopting the ideas of Galperin in [8], we explored a way of teaching software engineering in a more active, engaging and participative manner. According to Galperin teaching with this approach should take into account: i) the knowledge or skill to be taught should be in the form of situation-problem and its assessment should be the beginning and not the end as in traditional methods. ii) Activities should be selected and organized according to potential learners. iii) Activities should have a sequential presentation and must follow a mapping that enables the learner to achieve the solution of the problem immediately before the processing of learning be completed satisfied. All of this is to provide to the learner the opportunity to have a situation-problem experience and to learn about the operational logic of the solution. Lastly, iv) problem-situations should be correlated, allowing the learner to investigate general aspects. This approach is named formative-conceptual.

Based on this, we believe that a GBL strategy will meet the requirements of the formative-conceptual approach. Ebner and Holzingerb [5] suggest that there is evidence which shows that the learning results of using games is at least equivalent to the results from learning using the traditional methods. Besides, GBL also allow learners to have access to the concepts in dynamic and operational way. In [13] we show how GBL is preferred from the point of view of students, which points to the motivation students have in using games as a supporting tool for learning. Other important evidence reported in [1] is that related to long-term learning. Those concepts that are taught using situation-problem allow the learner to deduce and apply this concept in different situations.

However, to be more effective in achieving the requirements of the formative-conceptual approach, we believe that GBL is not enough. We understand that to be more effective GBL must consider how to improve the awareness of the learner. Complementing GBL will involve the learner as an active participant in their learning. As such, the learner should know how he/she is taught [14] by a transparent process. According to [16] awareness is the ability to interpret between each of the specific situations and its context of occurrence. That means that the learner gains knowledge by means of his/her own perceptions or by means of information and process knowledge. Accordingly, the learner assumes an active role, being involved on his/her own leaning.

We understand transparency as a concept related to information disclosure, which is been used in different settings, mostly related to the empowering of citizens with regard to their rights to know. In particular we are interested in a process view of transparency, a general quality, which is implemented by a set of policies, practices and procedures that allow citizens to have: accessibility, usability, informativeness, understandability and auditability of processes held by centers of authority. In our case the center of authority is the educator.

The use of GBL empowered by a transparent process was enacted in a software engineering class, in which we collected facts of how this has affected students’ performance.

III. Transparently Teaching

A. Transparency Pedagogy

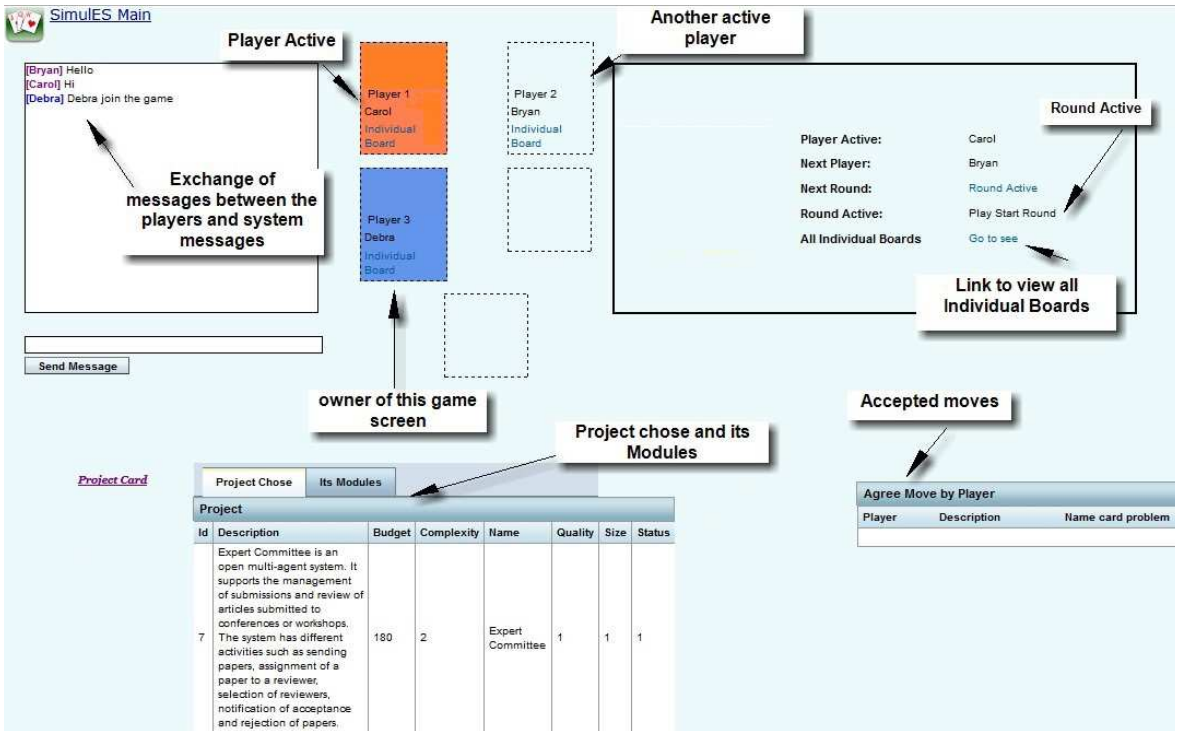
Our approach to teaching uses a vision anchored in the principle of transparency as information disclosure [11]. Transparency in pedagogy emerges as an important issue, which aims to make the learner aware about his/her teaching-learning process and content production [13].

This concept was instantiated defining pedagogy as a discipline that examines teaching methods to be better suited in promoting learning of increasingly complex new concepts, which are considered important for the development of thought [16]. Teaching-learning is a process that begins when the teacher creates opportunities for the production and construction of knowledge [8]. "Equip minds with skills to understand, feel and act in the society..." says Bruner in [4]. Thus teaching-learning promotes the formation and development of both subjects of the process [15], and for that reason, teaching without learning is nonexistent and vice versa Freire in [7]. So, learners take ownership and become aware of knowledge, because they learn to act conceptually (the conceptual practice) Puentes and Longarezi in [15]. That means, the process is finished when the learner has access to the meaning of the concept and when he/she have awareness of this process.

For that reason, we explore transparency. However, from the perspective of pedagogy, transparency seeks an environment where goals are open and teaching methods aims consensus by focusing on learner participation and feedback arising from his/her participation. Our aim in this paper is to show how we have instantiated this concept using GBL for software engineering.

B. SimulES-W

SimulES-W is the digital version of SimulES [6], an educational board and card game. SimulES is an evolution of the ideas of the Problems and Programmers (PnP) game [18] [38]. Different from PnP, SimulES-W does not have any specific development process and the development process can be explored pedagogically during the game; for instance one player can use an agile approach whereas the other can use a waterfall one. As such the learner may explore different development processes.



SimulES Main
Player Active
Another active player
Round Active
Exchange of messages between the players and system messages
Link to view all Individual Boards
owner of this game screen
Accepted moves
Project chose and its Modules
Player 1 Carol Individual Board
Player 2 Bryan Individual Board
Player 3 Debra Individual Board
Send Message
Project Card
Project Chose
Its Modules

[Bryan] Hello
[Carol] Hi
[Debra] Debra join the game

Player Active:	Carol
Next Player:	Bryan
Next Round:	Round Active
Round Active:	Play Start Round
All Individual Boards	Go to see

Project

Id	Description	Budget	Complexity	Name	Quality	Size	Status
7	Expert Committee is an open multi-agent system. It supports the management of submissions and review of articles submitted to conferences or workshops. The system has different activities such as sending papers, assignment of a paper to a reviewer, selection of reviewers, notification of acceptance and rejection of papers.	180	2	Expert Committee	1	1	1

Agree Move by Player

Player	Description	Name card problem

Fig. 1. Main Board SimulES-W [22]

The main precondition to play SimulES-W is being either a software engineering student or a person with basic knowledge and involved in software engineering. As described in [24], SimulES-W is used in both general and specific software engineering knowledge with an educational component that allows real practice to be simulated. SimulES-W is a multiplayer game and the player who wins the game is the one which first completes the software product with quality and budget defined in the project card. Figure 1 illustrates the main screen named Main Board. SimulES-W has been used as a teaching tool in other experiences with learners in software engineering. Some experiences have been described in [6, 17, 23, 24 and 25]. Learners exercise different roles where each role has to deal with the project budget and the hiring and firing of software engineers. Additionally, the game has a set of concepts and problem cards that are used to improve the one´s game or block other players' movement in their games. These cards display theoretical software engineering concepts that must be analyzed and applied by the learners. The knowledge of software engineering in these cards can be used either as obstacle, or stimulus for the game players´. Moreover, the game also has an activity for building the software product that makes the players exercise on the intrinsic software engineering concepts of the cards as the dynamics of building and audit, given by embedded bugs. The learners begin the construction of software artifacts required by the project, make inspections, and if a defect (bug) appear in some artifact, they should fix them, otherwise, the delivered product may fail acceptance test. So, players must take in consideration the risk of failure, if inspection and debugging are neglected. The first student, who can construct the software without having any problems, wins.

As described in [17] SimulES-W was developed for teaching software engineering in general. Alternatively, it can be configured to focus on a particular subject of knowledge as well, as was shown in [24], where it was used to teaching risk management. Then, since cards can be edited, we can use problem and concept cards tuned to the interest topic. We can also configure project cards to deal with specific artifacts.

Experiences with students using SimulES-W [17, 24 and 25] have shown that this game has the necessary elements to enact particulars of the software process as per concept and problem cards, making SimulES-W a powerful and useful tool, but also a fun way to teach [17].

C. Teaching with Transparently SimulES-W

With GBL, as a pedagogical alternative, there is a balance between entertainment and dissemination of knowledge, motivating learners to learn while they play [1]. GBL is being inserted to complement classes [1, 2, 3, 10, 18, 19, 20, 21 and 39], due to increasingly accessible tools and more realistic environments. Thus GBL allows the use of elements that are not provided by traditional methods [17]. They are also potential enhancers of the process in which they are used, by being in constant evolution of targeted improvements and suggestions of those who use these tools.

Our approach has three main activities (Figure 3). They are designed to provide transparency of the GBL process to students. We want to show how through the use of GBL it is possible to instantiate the concept of transparency in pedagogy. Consequently, it is necessary not only that the process be transparent but also the software (the game) itself. Software evaluation can be done with a set of heuristics available in [26] which have already been partly used in building SimulES-W and were described in [23]. The activities of Figure 3 are: i) **Plan** activity, it is responsible for the organization of information about the teaching material, thus creating the content to be used. The intentional model written in i* [27] is available and used to control the activity, since learners have access to it. In this activity the learners are informed about the open process strategy, and motivated to provide feedback. In the planning learners will be asked to fill a pre-test which assess motivation, preferences, level of knowledge, expectations, engagement to class, among others; ii) **Apply** activity, previous activity allows us to have sufficient knowledge about learners and know how to address the present activity. Learners are instructed to use SimulES-W, with the material for that specific class. On the completion of the activity, that is playing the game, a post-test is applied. It must assess how learners felt about participation, benefits and feedback items. Questions about: context of usage, clear instructions, didactic strategy, activity, consistency with the objectives and/or the contents. Finally, learners are called for an exam. The exam is designed to measure acquisition of knowledge and learners performance; iii) **Evaluate** activity, the information collected is evaluated from different perspectives and decisions are made with relation to the group and future activities. To close, as our perspective is addressed to transparency, students are informed about the results and measures to be taken.

As mentioned above, intentional models with i* were used to inform the students how SimulES-W works: strategies and activities dynamic. Figure 2 is an illustration of this kind of model, and this one in particular shows the main board of the game. However, the models were also pretty-printed (as text) and given to students as to enhance model readability.

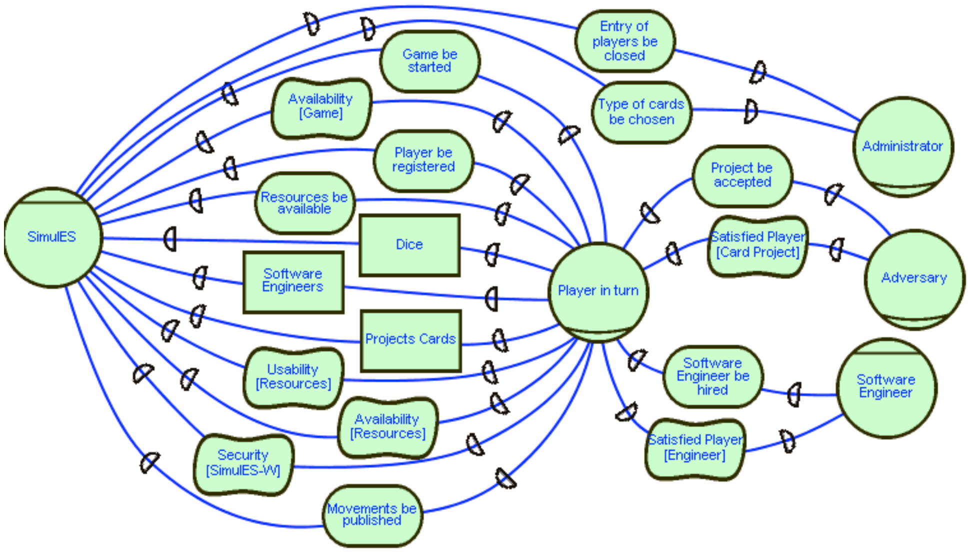


Fig. 2. SDsituation: Play round to start [13].

IV. Evaluation Approach

In order to introduce our approach an experiment was designed to verify the following hypotheses: *GBL together with intentional models (i*) could contribute to the transparently teaching of software engineering.*

Highlighting, the definition of transparency as a principle of disclosure information was introduced in [11]. Also, there are the lists of quality attributes [26] such as accessibility, usability, informativeness, understandability and auditability that are related to transparency. These quality attributes are further refined and their composition contribute to transparency [12] in the context of intentional models. In [9] there is a set of guidelines for assessing transparency attributes. For that reason we choose this kind of models to in our approach, trying to achieve transparency through this mechanism.

During the Planning stage, in the last half of 2013, an experiment was performed with a 26 student's class in the undergraduate software engineering course of the Computer Science program at the State University of Rio de Janeiro. This experiment was designed to study how SimulES-W as GBL with intentional models could influence pedagogy transparency. So we prepared the knowledge to be learned by creating specific concept and problems cards about requirements, design, coding, software quality and project management as described in [23,40]. The cards were also edited for the specific features of the class. Afterwards, the contents were added to the SimulES-W database. Finally, some tests were conducted to check the contents.

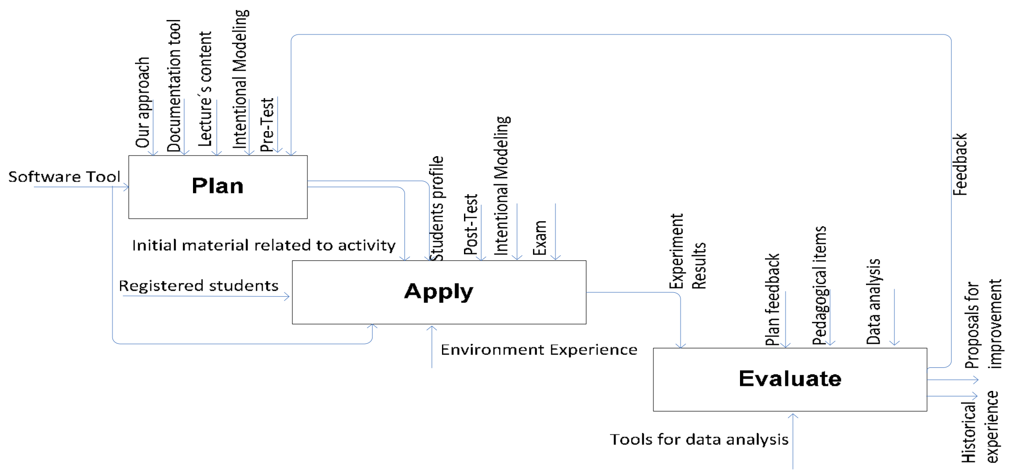


Fig. 3. SADT Diagram which shows activities related to our approach.

The experiment was designed to be applied on three different groups: (i) Lecture: This group was taught the concepts and problems in a traditional lecture. (ii) SimulES-W: This group was taught the concepts and problems using SimulES-W. (iii) SimulES-W with models: This group was taught the concepts and problems with SimulES-W however they had in advance the information how SimulES-W works by receiving (reading) i* models of SimulES-W.

First, a class was given to all groups and all the students received the information about experiment and the instructions about the division into three different groups and their activities. After that they all filled the pre-test and lastly, learners were randomly separated into the three groups. The 26 learners in the class were divided according to the fundamental principle of randomization to ensure the comparability of the groups and highlighting that they participated as volunteers to this case study. Each group with its activities was scheduled by days, participating in specific activities.

In Lecture (Group 1), learners attended a class, specifically on concepts and problems typical in software engineering. When the class had finished, the learners filled post-test related to contents and class perception.

In SimulES-W (Group 2), learners received information related to the activity and on SimulES-W. They also received instructions and details about the game: origin of the game, historical review, basic rules, dynamic rules, goals and main screens. They were then motivated to use the tool. The activity took place in a classroom with a teacher and two instructors who guided the learners and answered questions related to the activity. During this we emphasized the concepts and the problems related to subject. Instructions about the tool including navigation, interface features and execution of actions were explained.

In SimulES-W with models (Group 3), before class, learners received, by email, the documentation related to the game models and to the activity. This activity was the same as the previous one, except for the documentation provided, which learners could read and could use during the activity.

The students who used SimulES-W (Group 2 and Group 3) received previous training about the tool. It happened before each class through one presentation with the instructions and monitoring in place. This training took about half an hour

Finally, in another day the students of all groups took an exam. The activity **Evaluate** (Figure 3) was performed in 5 days. On the first day, 26 students were given the pre-test and received instructions about the activity. They were separated into three groups, each group of 12 students. On the second day, 8 students (66.6%) attended the class (Group 1) and filled the post-test. On the third day, 9 students (75%) participated in SimulES-W (Group 2) activity and filled the post-test. On the fourth day, 12 (100%) students participated in SimulES-W with i* activity and filled the post-test. On the fifth day, 22 students (84.6%) took the exam.

A. Pre-test

The pre-test was quantitative and had 7 closed questions; each of the questions had a basic description.

The first question: What kind of dynamic learning do you prefer? Competitive, cooperative or individual. This type of question can help teachers as to customize course topics. Figure 4 shows that students in that activity prefer cooperative learning (81%), after competitive (15%) and finally individual (4%).

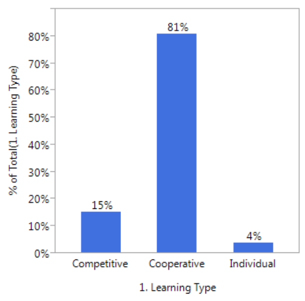


Fig. 4. Question 1. What kind of dynamic learning do you prefer?.

The second question, How do you rate your Software Engineering knowledge? As answers we listed: Very, Enough, Insufficient, Don´t not, Neither. We identified that students had some knowledge but not enough. Figure 5 shows how students rated themselves: as insufficient knowledge in SE (58%), enough (23%), Don´t not (12%) and neither (8%).

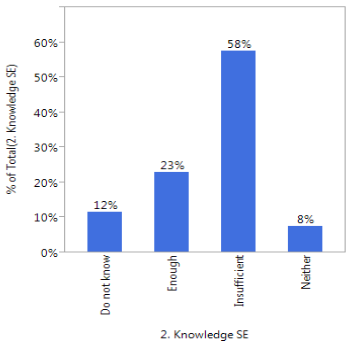
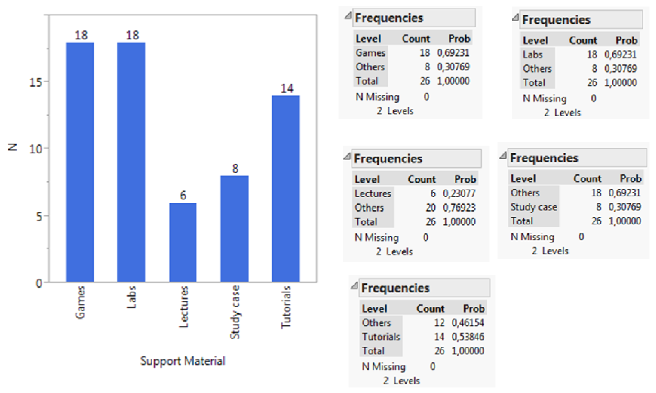


Fig. 5. Question 2. How do you rate your Software Engineering knowledge?

The third question, What deliverable strategy would you prefer for the course contents? This question is related to interest and preference about class material is delivered and was a multi-selection question. Figure 6 shows: 18 students chose Labs (69%); 18 students (69%) chose games; 6 students (23%) chose Lectures; 8 students (30%) chose study case based on papers; and 14 students (53%) chose tutorials.



Frequencies

Level	Count	Prob
Games	18	0,69231
Others	8	0,30769
Total	26	1,00000
N Missing	0	
2 Levels		

Frequencies

Level	Count	Prob
Labs	18	0,69231
Others	8	0,30769
Total	26	1,00000
N Missing	0	
2 Levels		

Frequencies

Level	Count	Prob
Lectures	6	0,23077
Others	20	0,76923
Total	26	1,00000
N Missing	0	
2 Levels		

Frequencies

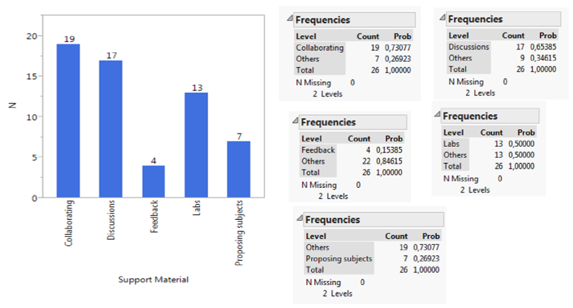
Level	Count	Prob
Others	18	0,69231
Study case	8	0,30769
Total	26	1,00000
N Missing	0	
2 Levels		

Frequencies

Level	Count	Prob
Others	12	0,46154
Tutorials	14	0,53846
Total	26	1,00000
N Missing	0	
2 Levels		

Fig. 6. Question 3.What deliverable strategy would you prefer for the course contents?

The fourth question, How would you like to participate in improving course quality? This question is addressed to identify how students would like to participate in improving the present course, this also was a multi-selection question; Figure 7 shows 19 students (73%) reported they would like to participate in collaborative activities; 17 students (65%) reported they would participate through discussion activities; 4 students (15%) chose feedback activities; 13 students (50%) chose labs; and finally, 7 students (26%) chose proposing topics.



Frequencies

Level	Count	Prob
Collaborating	19	0,73077
Others	7	0,26923
Total	26	1,00000
N Missing	0	
2 Levels		

Frequencies

Level	Count	Prob
Discussions	17	0,65385
Others	9	0,34615
Total	26	1,00000
N Missing	0	
2 Levels		

Frequencies

Level	Count	Prob
Feedback	4	0,15385
Others	22	0,84615
Total	26	1,00000
N Missing	0	
2 Levels		

Frequencies

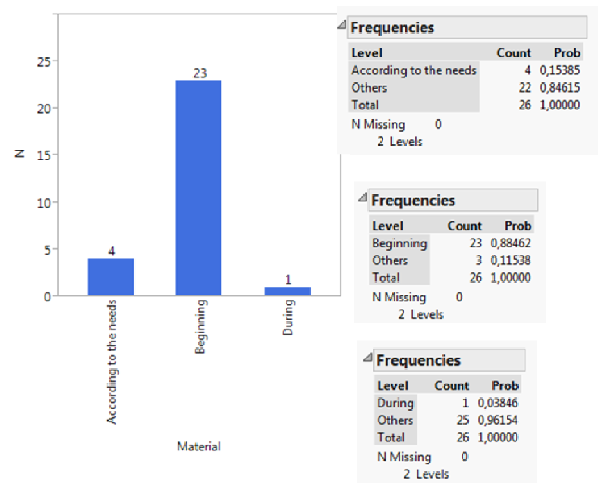
Level	Count	Prob
Labs	13	0,50000
Others	13	0,50000
Total	26	1,00000
N Missing	0	
2 Levels		

Frequencies

Level	Count	Prob
Others	19	0,73077
Proposing subjects	7	0,26923
Total	26	1,00000
N Missing	0	
2 Levels		

Fig. 7. Question 4. How would you like to participate in improving course quality?

The fifth question, When should the content and objectives of the course be provided? This question is related to preferences about foreknowledge of course information. Figure 8 displays: 4 students (15%) expressed that they should be provided according to the needs of the class; 23 students (88%) they report that the contents should be provided when the course begins; and finally, 1 student (3%) reported that the content should be provided throughout the course.



Frequencies

Level	Count	Prob
According to the needs	4	0,15385
Others	22	0,84615
Total	26	1,00000

N Missing 0
2 Levels

Frequencies

Level	Count	Prob
Beginning	23	0,88462
Others	3	0,11538
Total	26	1,00000

N Missing 0
2 Levels

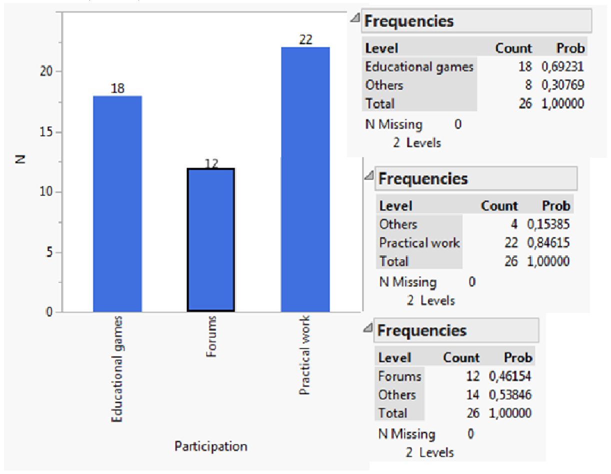
Frequencies

Level	Count	Prob
During	1	0,03846
Others	25	0,96154
Total	26	1,00000

N Missing 0
2 Levels

Fig. 8. Question 5. When should the content and objectives of the course be provided?

Question six was: What was your motivation to participate? This question is more general, and with that, we wanted to know what things are more motivating to the students, it also was a multi-selection question. Figure 9 portrays that 22 students (84%) reported they like especially practical work; next 18 students (69%) chose educational games; and finally, 12 students (43%) also chose forums.



Frequencies

Level	Count	Prob
Educational games	18	0,69231
Others	8	0,30769
Total	26	1,00000

N Missing 0
2 Levels

Frequencies

Level	Count	Prob
Others	4	0,15385
Practical work	22	0,84615
Total	26	1,00000

N Missing 0
2 Levels

Frequencies

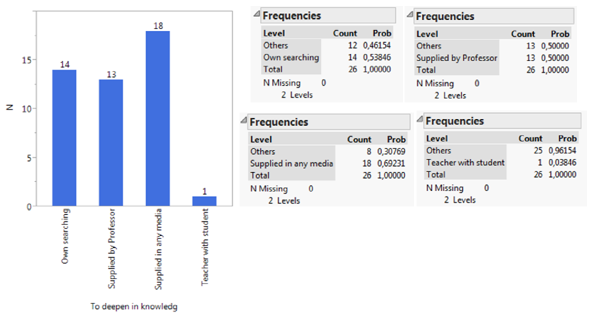
Level	Count	Prob
Forums	12	0,46154
Others	14	0,53846
Total	26	1,00000

N Missing 0
2 Levels

Fig. 9. Question 6. What was your motivation to participate?

The last question, How would you like to dig deeper into content class? This question aims to identify the level of participation and student preferences and as this should be done. Figure 10 shows: 18 students (69%) think that the information should be available in some media; 14 students (53%) think that they should look for their own information; 13 students (50%) consider that the teacher should provide the information; and finally 1 student (3%) reported that the teacher should assist the student in finding information.

We identify with the pre-test, preferences, motivations and knowledge level of the students; also this information was useful to identify how the class should be addressed. In addition, when a teacher knows about students' background knowledge, skills as well as their needs, the teacher can be better prepared to promote an effective learning.



Frequencies

Level	Count	Prob
Others	12	0,46154
Own searching	14	0,53846
Total	26	1,00000

N Missing 0
2 Levels

Frequencies

Level	Count	Prob
Others	13	0,50000
Supplied by Professor	13	0,50000
Total	26	1,00000

N Missing 0
2 Levels

Frequencies

Level	Count	Prob
Others	8	0,30769
Supplied in any media	18	0,69231
Total	26	1,00000

N Missing 0
2 Levels

Frequencies

Level	Count	Prob
Others	25	0,96154
Teacher with student	1	0,03846
Total	26	1,00000

N Missing 0
2 Levels

Fig. 10. Question 7. How would you like to dig deeper into content class?

As a matter of fact, we also identified motivations and preferences in general behavior of the group when we observed them, for example, we had more support among those groups where students were scheduled in activities with SimulES-W. One of the activities with SimulES-W was a Friday, early and with rain and all students attended the activity. The day that results were presented, we offered a completion activity, those who did not play or want play again could do that. To our surprise, all the students stayed for that latter activity. This shows that students are motivated to learn and to participate in didactic activities driven by GBL.

B. Post-test

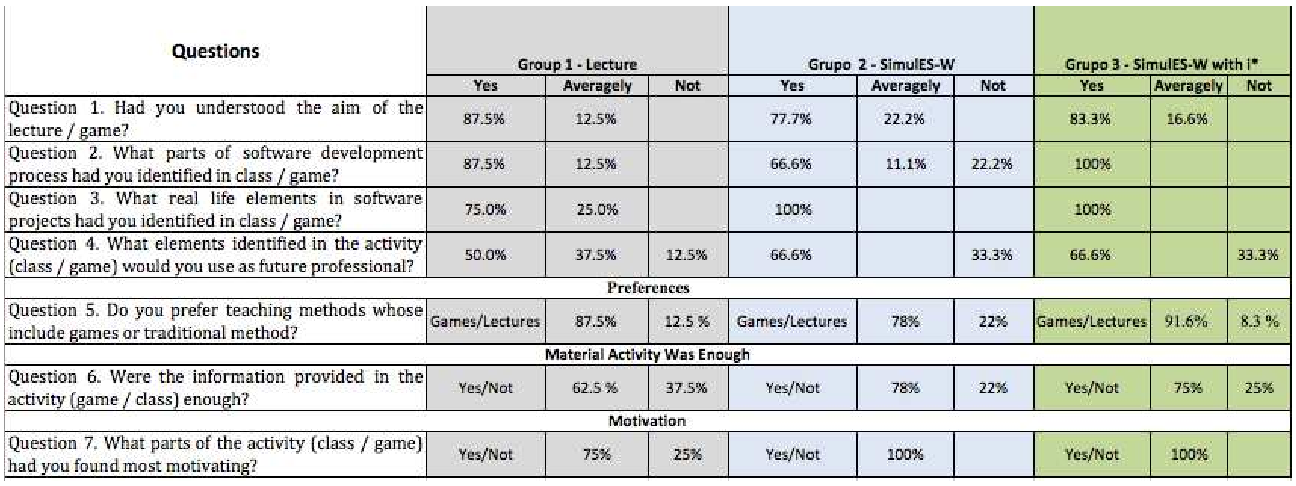
We designed a post-test for each of the experiences. Thus, The post-test for group 1 (lecture) had 9 questions. The post-test for group 2 (SimulES-W) had 13 questions. And the post-test for group 3 (SimulES-W with i*) had 14 questions. Some questions were closed and others open. Also, we created specific questions for each experience. We considered it necessary to do specific questions for each of the activities. Especially, for activities which used the game. To illustrate, we asked about software, interface, game dynamic, collaborative aspects, and competitive aspects, among others. In addition, we created similar questions for the three groups. And for reasons space in this paper, we will focus on these last.

In short, we identified more participation and motivation in those groups where the game was used. It was reported more negative aspects related to traditional class. Group 3, who used to SimulES-W with i* got better performance and that was evident in the response of the questions. All three groups showed preference for learning software engineering by means of GBL.

The Table 1 shows a summary of similar questions for the three groups. The questions 1, 2, 3 and 4 show that Group 3 (SimulES-W with i*) had the best percentage in effective responses, followed by Group 1 (Lecture) and Group 2 (SimulES-W). Next, all students expressed their preference for games, being that Group 3 (SimulES-W with i *) reported the highest percentage. Related to Question 6 in general students were satisfied with material of the activity. Finally, Question 7 All students found motivating elements in the three activities, however, Groups 3 (SimulES-W with i*) and Group 2

(SimulES-W) were who most elements found and were more motivated in the activity. As a whole, according to the responses and with what we observed in all experiences, the most unmotivated group was Group 1 (Lecture) being the group that more criticized the activity.

TABLE I. RESUME POST-TEST



Questions	Group 1 - Lecture			Grupo 2 - SimulES-W			Grupo 3 - SimulES-W with i*		
	Yes	Averagely	Not	Yes	Averagely	Not	Yes	Averagely	Not
Question 1. Had you understood the aim of the lecture / game?	87.5%	12.5%		77.7%	22.2%		83.3%	16.6%	
Question 2. What parts of software development process had you identified in class / game?	87.5%	12.5%		66.6%	11.1%	22.2%	100%		
Question 3. What real life elements in software projects had you identified in class / game?	75.0%	25.0%		100%			100%		
Question 4. What elements identified in the activity (class / game) would you use as future professional?	50.0%	37.5%	12.5%	66.6%		33.3%	66.6%		33.3%
Preferences									
Question 5. Do you prefer teaching methods whose include games or traditional method?	Games/Lectures	87.5%	12.5 %	Games/Lectures	78%	22%	Games/Lectures	91.6%	8.3 %
Material Activity Was Enough									
Question 6. Were the information provided in the activity (game / class) enough?	Yes/Not	62.5 %	37.5%	Yes/Not	78%	22%	Yes/Not	75%	25%
Motivation									
Question 7. What parts of the activity (class / game) had you found most motivating?	Yes/Not	75%	25%	Yes/Not	100%		Yes/Not	100%	

C. Exam

An exam was applied to all groups one week after we had finished the classes. The exam had 6 questions related to concepts and problems in software engineering that were presented during the activities. The exam aimed the identification of how the concepts that were understood by learners. In short, the exam was to verify if the process teaching-learning achieved its goals. The students were asked to explain: i) What is software engineering. ii) The different roles people canplay when they participate in a software project. iii) The artifacts generated in the software development process. iv) Some problems that can appear during the software construction process and describe why they happen. v) Types of concepts considered as good practices in software engineering and how to describe them. vi) Explain the importance of software quality assurance as a control mechanism in a software project.

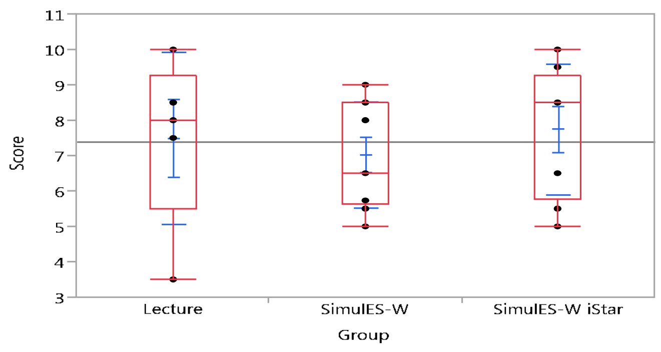


Fig. 11. Results of each group related to Exam Applied

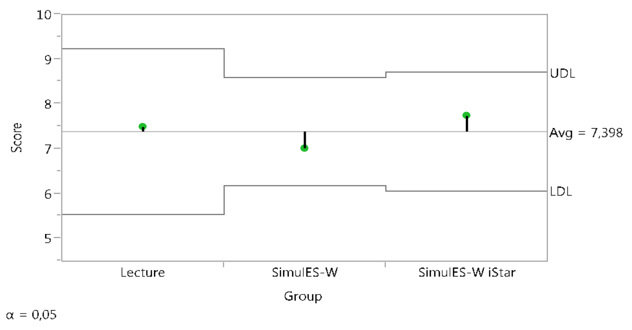


Fig. 12. Analysis of the Mean related to Exam Applied on Each Group

Figure 11 shows the scores range of each group, the highest and the low score. The results show that most of the students who attended the activity with SimulES-W got satisfactory score. Being that the mean for group as Group 1 (Lecture) 7.5, Group 2 (SimulES-W) 7.03 and Group 3 (SimulES-W with i*) 7.75. That means students in group 3 had a better performance on the exam. However, this difference is not significant.

Table 2 presents the results of the means using the ANOVA (Analysis of the variance). On this test we can conclude that it is impossible to reject the null hypothesis at 0.025 level so that is no considerable difference among the means of the three groups. This result is based on the fact that the F ratio (0.3312) is less than the value 4,51 of F 0.025, 2,19. Then we test for each group in separate and we had the same results, that the three groups have no difference relevant means. The Figure 12 shows the Analysis of the means of each groups and the average is 7.398 that is a good result. We also can see the tendency that Group 3 had achieved better performance than the other two groups and Group 2 had the lowest and Group 1 has been in the average of the two groups.

Analyzing these results we can conclude that the performance of the groups are equivalent, however there is a tendency that the group that used a GBL with transparency had a better performance and more motivation. The group that had traditional lecture did not lose the lesson content but was the least motivated.

TABLE II. ANALYSIS OF VARIANCE

Source	DF	Sum of Squares	Mean Square	F Ratio	Prob > F
Group	2	2.276831	1.13842	0.3312	0.7221
Error	19	65.305556	3.43713		
C. Total	21	67.582386			

V. CONCLUSION

Our work goal is to instantiate the concept of transparency in pedagogy. Transparency in pedagogy emerges as an important issue, which aims to make the learner aware about teaching-learning process and content [13]. Our approach is based on the concept of transparency as information disclosure [11] and on the formative-conceptual approach of Galperin in [8]. That approach also supports the idea that students better meet the learning goals, if they are motivated and the contents are related to real life situation.

Our approach used intentional models together with GBL to a Software Engineering teaching activity aiming at a more transparent class. We compared the results with a traditional teaching method as well as the use of GBL without intentional models. Our results, although applied to a small number of students who used the models, indicates that there is a positive outcome in the use of the transparency strategy with GBL (Group 3). This is a strong motivation to do more experiments focused in this direction that may lead to more conclusive statements.

In the practice related to teaching software engineering is necessary to understand and implement new and better teaching methodologies that aim better results in knowledge transfer. With this work we point out that pedagogy have provided quite important efforts to identify and disseminate methodologies, which seek to promote active participation of students in teaching-learning. For this reason, it is important that educators in SE thrive to use better practices in classrooms. The teaching-learning process is changing and we as educators must be prepared for these changes. In fact, the student should be more involved in this process. Armed with this, software engineering also has the advantage of having tools that should be taught and at the same time may be part of the practice in that process. That means, teaching with the same tools that will be used by students in their professional life is certainly a SE strength. Using GBL in the classroom, we are putting into practice what the new pedagogy is preaching and if we disclose the process to students, such as the case presented with the use of intentional models with i*, we will be offering a more transparent teaching-learning process.

Future work should continue evaluation activities as a feedback generation for improving our understanding on how to design better strategies for pedagogy transparency based on GBL. Also, it is necessary to devise a monitoring strategy to check if students retained the knowledge taught. A possibility is to strengthen the ties with related future courses, as to check retention.

ACKNOWLEDGMENT

Leite acknowledges the partial support of CNPq and Faperj, Cientista do Nosso Estado; Monsalve acknowledges the support of Capes; Leite and Monsalve also acknowledge the support of the CNPq 557.128/2009-9 and FAPERJ E-26/170028/2008 grants for the Brazilian Institute for Web Science Research.

REFERENCES

[1] A. Baker, E. Navarro, and A. van der Hoek, “Problems and Programmers: an educational software engineering card game”. In Proceedings 25th Intern. Conference on Software Engineering, IEEE Computer Society Press, 2003, pp 614-619.

[2] J. Beatty, and M. Alexander, “Games-Based Requirements Engineering Training: An Initial Experience Report”, International Requirements Engineering, RE '08. 16th IEEE, Catalunya, Spain. (Sept. 2008). Pp. 211-216.

[3] T. Birkhoelzer, E. Navarro and A. Van Der Hoek, “Teaching by Modeling instead of by Models”, Proceedings 6th International Workshop on Software Process Simulation and Modeling, St. Louis, 2005, MO, 4.

[4] J. Bruner, “A cultura da educação”. (2001) Porto Alegre: Artmed.

[5] M. Ebner and A. Holzinger, “Successful implementation of user-centered game based learning in higher education: an example from civil engineering”. Computers and Education, 49(3), (2007). pp. 873–890.

[6] E. M. L. Figueiredo, C. A. Lobato, K. L. Dias, J. C. S. P. Leite, C. J. P. Lucena, “Um Jogo para o Ensino de Engenharia de Software Centrado na Perspectiva de Evolução”, Workshop sobre Educação em Computação (WEI – 2007), pp. 37-46.

[7] P. Freire, “Pedagogia da autonomia: saberes necessários à prática docente”. (1996) São Paulo: Paz e Terra.

[8] P.I. Galperin, “Organization of mental activity and effectiveness of learning.” Soviet Psychology, Moscow, v. 27, n. 3, (may/june 1989) pp. 65-82.

[9] GTS. Wiki of Software Transparency Group. At http://transparencia.inf.puc-rio.br/wiki/index.php/Transparência.

[10] A. Jain and B. Boehm, “SimVBSE: Developing a Game for Value-Based Software Engineering”. Proceedings 19th Conference on Software Engineering Education and Training, 2006, pp. 103 -114.

[11] J. C. S. P. Leite and C. Cappelli, “Software Transparency”. Business & Information Systems Engineering: Vol. 2, 2010, Iss. 3, 127-139. At: http://aisel.aisnet.org/bise/vol2/iss3/3.

[12] J. C. S. P. Leite and C. Cappelli, (2008). Exploring i* Characteristics that Support Software Transparency. In iStar (pp. 51-54).

[13] Monsalve, E. S. ; Werneck, Vera M. B. ; Leite, Julio Cesar Sampaio do Prado . A Case Study to Evaluate the Use of i* for Transparent Pedagogy. In: iStar 2014 Seventh International i* Workshop, 2014, Thessaloniki.

[14] E. Monsalve, V. Werneck and J. C. S. P. Leite, "Incorporando Transparência na Pedagogia através do Uso de Jogos para Ensino". 2013. WTRANS, SBES, Brasília.

[15] R. Puentes and A. Longarezi, "Escola e didática desenvolvimental: seu campo conceitual na tradição da teoria histórico-cultural" Educação em Revista Belo Horizonte. 2013, in Portuguese.

[16] A. Resende and H. Valdes, "Galperin: implicações educacionais da teoria de formação das ações mentais por estágios." Educação & Sociedade 27.97 (2006). pp. 1205-1232, in Portuguese.

[17] E. Monsalve, V. Werneck and J. C. S. P. Leite, (2011). Teaching Software "Engineering with SimulES¬W". Proceedings of XXIV Conference on Software Engineering Education and Training (CSEE&T 2011), Hawaii, USA. (pp. 31–40).

[18] PnP Problems and Programmers. (April.2012) at http://www.problemsandprogrammers.com/.

[19] G. Regev, D. Gause and A. Wegmann, "Requirements Engineering Education in the 21st Century, an Experiential Learning Approach", The 16th International Requirements Engineering Conference (RE'08), 2008, pp. 85-94.

[20] R. Smith and O. Gotel, "Using a Game to Introduce Lightweight Requirements Engineering", in Proceedings of the 15th IEEE International Requirements Engineering Conference, 2007, pp. 379-380.

[21] R. Smith and O. Gotel, "Gameplay to Introduce and Reinforce Requirements Engineering Practices", in Proceedings of the 16th IEEE International Requirements Engineering Conference, 2008, pp. 95-104.

[22] Software Engineering Simulation by Animated Models (SESAM) – Stuttgart–Germany. (April.2010) <http://www.iste.uni-stuttgart.de/se/research/sesam/overview/index_e.html>.

[23] E. Monsalve, (2014). "Uma Abordagem para Transparência Pedagógica usando Aprendizagem Baseada em Jogos". PhD Thesis in Portuguese, PUC–Rio, April 2014.

[24] E. Monsalve, A. Pereira and V. Werneck, Chapter: Software Engineering Teaching Through Collaborative Game. Upcoming book, "Overcoming Challenges in Software Engineering Education: Delivering Non-Technical Knowledge and Skills". Igi-Global, 2013.

[25] E. Monsalve, V. Werneck and J. C. S. P. Leite, SimulES-W: "Retroalimentação Evolutiva num Jogo para Ensino na Engenharia de Software". 2013, FEES, SBES, Brasília.

[26] Software Transparency. (2013) Available: http://transparencia.inf.puc-rio.br/.

[27] Yu, E. "Modeling Strategic Relationships for Process Reengineering". Ph.D. Thesis, Graduate Dept. of Comp. Science, University of Toronto, (1995).

[28] E. Sweedyk and R.M. Keller, (2005). Fun and games: a new software engineering course. ACM SIGCSE Bulletin. ACM, 2005. 37(3), 138-142.

[29] M. Barros and R. Araújo, (2008). "Ensinando Construção de Software Aplicada a Sistemas de Informação do Mundo Real". In: Proceedings of the First Forum on Education in Software Engineering (FEES), Campinas, Brazil.

[30] S. Qin and C.H. Mooney, (2009). "Using game-oriented projects for teaching and learning software engineering". In: Proceedings of 20th Annual Conference for the Australasian Association for Engineering Education (pp. 49-54). The University of Adelaide. Engineers Australia.

[31] A. Bollin, E. Hochmuller and R.T. Mittermeir, (2011). "Teaching Software Project Management using Simulations". In: Proceedings of XXIV *Conference on Software Engineering Education and Training (CSEE&T*) (pp 81–90); Hawaii, USA.

[32] A. Jain and B. Boehm, (2006). "SimVBSE: Developing a game for value-based software engineering". In: Proceedings of 19th Conference on Software Engineering Education and Training (CSEET) (pp. 103–114). Turtle Bay Resort, Oahu, Hawaii.

[33] J. Burge, (2009). "Application and Appreciation: Changing Course Structure to Change Student Attitudes", Proc.22nd IEEE Conf. Soft. Eng. Education & Training, pp. 45-52.

[34] Q. B. Li and B. Boehm, (2011). "Making Winner for Both Education and Research: Verification and Validation Process Improvement Practice in a Software Engineering Course, Proc. 24th IEEE Conf. Soft. Eng. Education &Training, 2011, pp. 304-313.

[35] N. Martin, et al. (2011). "Teaching software engineering using globally distributed projects: the DOSE course", Proceedings of the 2011 Community Building Workshop on Collaborative Teaching of Globally Distributed Software Development. ACM.

[36] S. Jarzabek, (2013), "Teaching advanced software design in team-based project course". Software Engineering Education and Training (CSEE&T), 2013 IEEE 26th Conference on. IEEE.

[37] M. Daun, A. Salmon, B. Tenbergen, T. Weyer and K. Pohl, "Industrial case studies in graduate requirements engineering courses: The impact on student motivation." Software Engineering Education and Training (CSEE&T), 2014 IEEE 27th Conference on. IEEE, 2014.

[38] A. Baker, E. Oh Navarro and A. Van Der Hoek, (2005). An experimental card game for teaching software engineering processes. Journal of Systems and Software, 75(1), 3-16.

[39] Connolly, Thomas M., Elizabeth A. Boyle, Ewan MacArthur, Thomas Hainey, and James M. Boyle. A systematic literature review of empirical evidence on computer games and serious games. Computers & Education 59, no. 2 (2012): 661-686.

[40] E. Monsalve. "Uma Abordagem para Transparência Pedagógica usando Aprendizagem Baseada em Jogos". Thesis in Portuguese, Department of Informatics, PUC–Rio, April 2014.