ISSN No. 0976-5697

Volume 2, No. 2, Mar-Apr 2011

International Journal of Advanced Research in Computer Science

CASE STUDY AND REPORTS

Available Online at www.ijarcs.info

Software Engineering • Role-Playing Game: an interactive game for Software Engineering education

Fabiane Barreto Vavassori Benitti
Universidade do Vale do Itajaí, Itajaí, Univali
Itajaí, Brazil
fabiane.benitti@univali.br

Abstract: The Education on Software Engineering is based on the use of models that provide assistance to the management and development of software project. SE•RPG appears as a pedagogical proposal that intents to support the learning process through the simulation of a software company environment, challenging the students with the activities and problems from the development process management. A classroom evaluation proved SE•RPG to be a tool capable to minimize the gap between theory and practice in the learning process, and also a stimulating resource.

Keywords: Software Engineering Education; project management; life cycle models, game based learning, computer-assisted instruction.

I. INTRODUCTION

Few areas of education face the challenges in the same way as Software Engineering does when it comes to select the curriculum contents that will be valuable in the students professional careers. Not only because the Software Engineering is a relatively new area, but also because of the changes that quickly occur in terms of the technology fundamentals, which take place in the methods and tools to support Software Engineering.

The teaching in the software development process and project management at classroom is limited to a succession of theorist lectures and projects of didactic representation that compose an insufficient learning of the Software Engineering domains [1] [2].

Baker, Navarro and Hoek [3], reinforcing that the root of the problem seems to be found in the way that Software Engineering is generally learned: throughout a series of theories and concepts demonstrated in classes, the students must develop a small project in classroom as an attempt to use the knowledge previously obtained. Despite the theory and practice in Software Engineering are needed for the education of future professionals, this limited practice does not comply correctly to the critical actions involved in the engineering process. The teacher may explain most of these actions in expositive classes, but the students will not have the opportunity to be placed in a complete Software Engineering process and will not practice cases based on different life cycle models.

Research in the fields of training and education suggests that using games in the training process can engage students, reinforce concepts through practice, and achieve deep learning, supporting the retention of the content [4].

From the difficulties experienced in class and based on research carried out ([1][2][3][4]), the SE•RPG (Software Engineering • Role Playing Game) has been developed. SE•RPG provides assistance to the learning process through the RPG, placing players in 'real-life' situations that are comparable to ones they might encounter and from which they learn the consequences of the reactions they might have. As learning tools, RPG aims at providing players with knowledge of a given situation [5].

This paper is structured as follows: Section II presents background information on some educational features that compose SE•RPG such as software development process and project management. Section III explains the learning aspects in Software Engineering domain. In Section IV, a guide for the SE•RPG is presented, reporting its functionalities. In Section V, the experiment to evaluate the game is discussed and the results are presented. Finally, conclusions and comments on future work are given in Section VI.

II. SOFTWARE DEVELOPMENT PROCESS AND PROJECT MANAGEMENT

Software development process is a series of combined methods and practices that lead to the production of a software product [6] and it is composed by some stages, or phases, each one executing specific defined tasks that result the project conclusion [7]. Despite there are so many different development processes, some phases are fundamental to all of them, referred in a particular manner by each author. Based on the models described for [6], [8] and RUP (Rational Unified Process) process [9], SE•RPG adopts as standard the phases as follows: (i) requirements specification; (ii) system analysis and design; (iii) implementation; and (iv) test.

There is not an 'ideal' software development process, but there are many ways to improve existing processes, as the standardization of the processes. Software development process models are abstracts representations of the process, which can be used to explain different approaches of the software development [6]. SE•RPG considers the models as follow: (i) waterfall model; (ii) iterative model and (iii) prototyping.

According to Sommerville [6], the waterfall model has this name due to the systematic sequence of a phase to another one, where the results of each stage are the beginning of the following phase. SE•RPG allows the student “to try” the development based on the waterfall model approaching the phases of specification as illustrated in Figure 1.

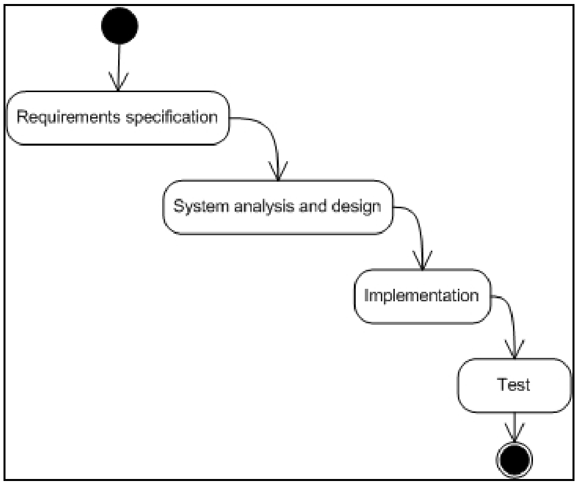


Figure 1. Waterfall model

The iterative model is often more effective than the waterfall model, because it develops the software incrementally, using the knowledge previously acquired in each version available. The SE•RPG uses the iterative model as described by [7] and shown in Figure 2.

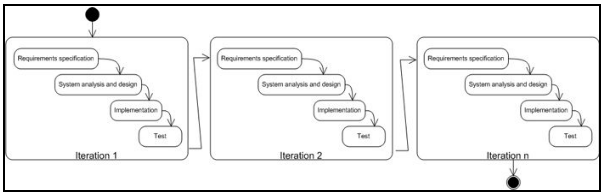


Figure 2. Iterative model

The basic idea of the prototyping is that instead of freezing the requirements before a design or coding proceed, a throwaway prototype is built to understand the requirements. For the development of the prototype, design, coding and testing are performed. However each of these phases is not performed very formally. By using this prototype, the client can get a "view" of the system, once the interactions with prototype can help the client understanding the requirements of the desired system. Figure 3 shows how SE•RPG approaches the prototyping, based in model presented by [7].

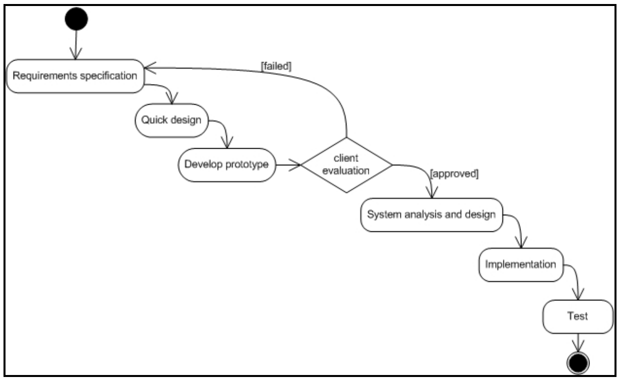


Figure 3. Prototyping

Project Management composes an essential concern in the software development process and covers a series of concerns, such as lack of time to execute a task, the complexity of the project or an inadequate budget [8]. Fewings [10] recommends the study of three fundamental dimensions of project management and its inter-relations to guide the procedures in these situations: (i) time; (ii) tasks; and (iii) resources. These features, as shown in Figure 4, interact continuously complying with the progress of the project. These three fundamental dimensions constitutes the objectives to be reached in the SE•RPG.

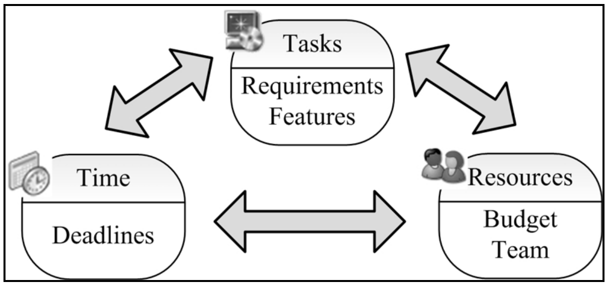


Figure 4. Dimensions of project management

III. GAME AND LEARNING IN SOFTWARE ENGINEERING

Some educators consider game-based learning to be a powerful instructional method. This technique deals with game applications that have defined learning outcomes [11]. Prensky [11] defines a game as "any contest (play) among adversaries (players) operating under constraints (rules) for an objective (winning, victory, or pay-off)." Educational games (or serious games) are designed to teach people about a certain subject, expand concepts, reinforce development or help them developing a skill. Recently, instructors have begun using games for Software Engineering education.

The objective of the game presented in this article is the learning of aspects related to the development life cycle and the software project management. Similar initiatives can be found in literature and are approached in this section.

A different approach to Software Engineering learning is cited by [3], introducing a card game simulating the software development process called Problems and Programmers. The game takes place in a competitive structure where gamers draw cards based in the waterfall model and their cards move between columns representing each phase of the process until its conclusion.

Another successful approach is SimSE, a desktop graphic software application that simulates the environment around the 'real-like' development process in a manner to represent the truly situations included in it [12]. SimSE consists in three main components: (i) a simulation module that contents the environment and elements in the process, such as developers and artifacts; (ii) a graphical interface to interact with the user; and (iii) game metric that rules the game [2].

Steele [13] presents a comparison of the 12 different project management simulation tools and evaluated against various desirable features. Steele [13] observed that a notable area is the lack of simulation customization (in particular the inability to customize, turns, projects, simulation duration, and life cycles.)

SE•RPG is a solution similar to the SimSE, but it runs in the web and has 3 different models in a single game. Moreover, its design is based on principles of the RPG (not the simulator, the focus of SimSE). Another feature that highlights the SE•RPG over the other initiatives mentioned above is the integration of strong aspects of project management, as it has been built to facilitate future customization by allowing all the settings (projects, tools, cost, duration and team) to be defined in an XML file (supplying the deficiency pointed out by [13]).

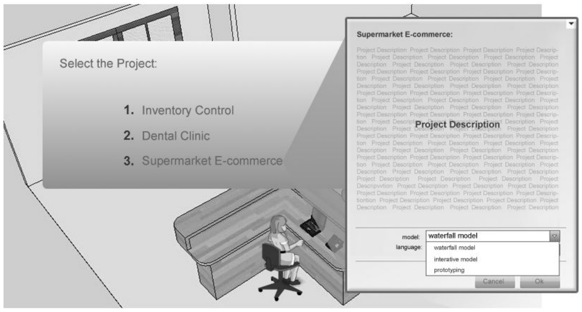
IV. SE•RPG

SE•RPG appears as a proposal to create a tool to support the software development learning process through the simulation of a 'real-like' environment, with rules to allow the student to confront the challenges and participate in the activities included in the software development. The game must enable the player (student) to take place in the process, playing the role of a project manager and making choices that will affect the game results.

The dynamics of the game have the following steps in sequence:

(i) Select the project (Figure 5): The player must choose between three project options (Software for inventory control of a shoe store, software for managing a dental clinic or e-commerce site for a supermarket). After getting to know the characteristics of the project (through a brief description of the projects, cost and time), the player sets the programming language (SE • RPG considers C + +, Java, Pascal and PHP) and select the life cycle (considering the three models described in Section II - Waterfall, Iterative or Prototyping).

(ii) Choose the project team (Figure 6): Each character (potential team member) has a description, such as "... he is good with the elicitation of requirements, but has little knowledge of design patterns; has a good knowledge in programming languages Java and Pascal ...", this description provides indicative of the abilities of the characters. The player must select the characters that will be part of the team, and for this he should consider the skills needed for each phase of the life cycle. The character of the example will have good performance in the phase of requirements specification and implementation (in Java and Pascal), he will not perform well in the phase of analysis and design though.



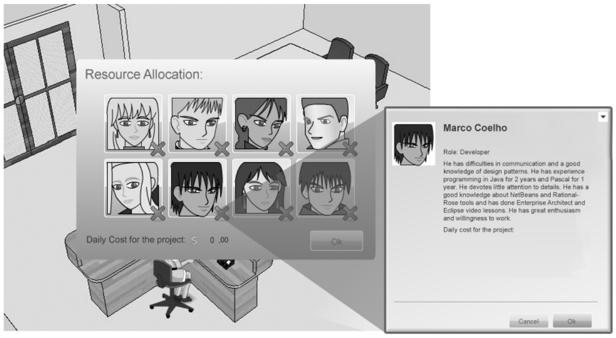
Supermarket E-commerce:

Project Description

model: waterfall model

language:

Figure 5. Selecting project and model



Daily Cost for the project: $ 0,00

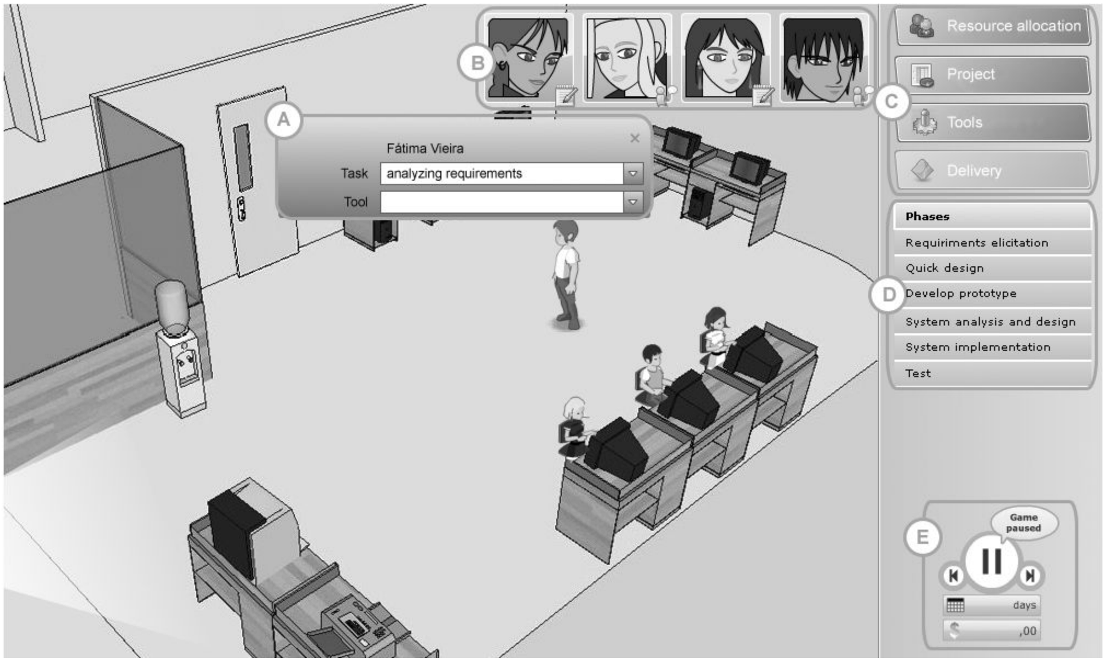
Marco Coelho

Role: Developer

He has difficulties in communication and a good knowledge of design patterns. He has experience programming in Java for 2 years and Pascal for 1 year. He devotes little attention to details. He has a good knowledge about NetBeans and Rational-Rose tools and has done Enterprise Architect and Eclipse video lessons. He has great enthusiasm and willingness to work.

Daily cost for the project:

Figure 6. Composing the development team



Fátima Vieira

Task: analyzing requirements

Tool:

Figure 7. Game screen: (a) delegating tasks; (b) characters illustration; (c) buttons to allocate team, view project information, purchase tools and delivery the project; (d) project progress; (e) time control.

(iii) Lead the development of the project (Figure 7): in the major step of the game, the player must assign tasks to team members (7A and 7B), and follow the life cycle selected (7D).The game does not allow the player to change the sequence of phases. Progress bars are displayed as the game runs(7D), and the player controls the turns of the game (7E). Also in this step, the player may fire team members, hire new ones (7C), buy tools (7C), determine which team member should use them and in which task (7A). The use of appropriate tool makes the task progress to go faster (if the character has the ability to use the tool). In order to help with the project management, it is presented the time of the project and the budget spent to date (7E).

(iv) Conclusion of the project and presentation of the results: after the player delivers the project (7C), the game shows an analysis of project progress, reporting on the suitability of the selected life cycle, budget analysis (% of profit or loss) and status of finished on time or late.

In addition, in order to approach the reality and impose further challenges to the player during step (iii), team members may briefly leave the project because they are sick or on holiday, for example.

Internally, the game mechanics consists in a system of rules that supply the simulation, generating probabilities close to reality. D20 System admitted the implementation of features as proposed above. The D20 System mechanics are based on a rule where the generation of a random value between 1 and 20 is added to relevant modifiers for the task, as the character ability and use of appropriate tool; and the result of this is compared to the value (level) of difficulty for this task, presenting the success degree or failure in the task execution [14]. It is not the objective of this paper to present the architecture or system of rules of the game based on the RPG mechanisms found in [14]. Adjustments were carried out to apply the system of rules to the game, however, this is not the focus of this article.

With this set of game features, we believe that the game can help students understanding about the characteristics of each of life cycle model, as well as differences between them, helping to identifying the phases (nomenclature and sequence), and help identifying basic and essential activities in project management. We believe the possibility to "try on" the development of different projects with different life cycles may allow the student to reinforce the theoretical concepts discussed in class.

V. Evaluation

A. *Evaluation Process*

To ensure evaluation of SE•RPG as a learning tool to the software development process and project management, a test with two questionnaires was applied to a sample of twenty-four Computer Science students during Software Engineering class, in two different moments: before and after using the SE•RPG. The first questionnaire was to provide a comparison to the data collected in the second one.

Both pretesting and posttesting consisted of questions about life cycle models (waterfall model, prototyping and iterative model) and project management, asking students to correctly identify the characteristics of each model from a list with 12 questions about models provided, and one question asked to students to identify 3 project manager activities.

The evaluation process took place in four stages: (i) the teacher of Software Engineering gave a lecture (2 hours) presenting concepts related to software life cycle, characteristics of different models, their advantages and disadvantages. Before class, reading material was available to students; (ii) the pretest was applied to the students (15 minutes); (iii) a brief explanation on how to play SE • RPG was conducted to the students (this explanation was focused on how to operate the game's interface and not on game strategies, ie, without approach the concepts of models and management). Afterwards, the students were instructed to play at least one time each project (thus covering the three models), this activity was performed at 1:30 hours; (iv) finally, the posttest was applied again to all students.

B. *Hypothesis Evidence and Analysis of the Results*

The null hypothesis for the experiment was as follow:

H01: There will be no difference between the overall pretest and posttest scores about project management.

H02: There will be no difference between the overall pretest and posttest scores about software development process.

To obtain a tendency to sample, a distribution test was applied in order to compare collected data. In the collected sample, answers related to the project management were analyzed by a paired T-test, appropriated to a small sample.

When looking at achievements in project management for this group of students, before and after the playing, by using the T-test, we find a positive shift in the mean from 0.54 to 1.96 with p-value = 0.000 (which means significant at all significant levels) indicating better performances in project management for the group.

As shown in Figure 8, 20 students obtained a better performance in the posttest, while 4 students showed no improvement.

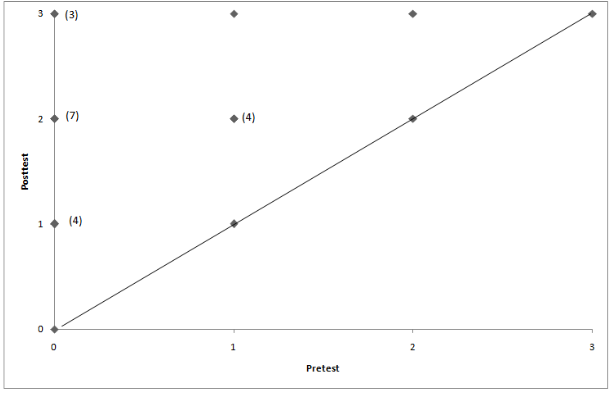


Figure 8. Scatterplot of pretest versus posttest scores which relates to questions about project management. *The number next to some points refers to the number of students who achieved the same scores.*

In the test, the question to the students was "Type 3 activities that must be performed by the manager of a software project." Thus, the correction of the question involved identifying how many correct activities were presented in the pretest and posttest. It is important to mention that the students did not receive explanation about project management. All the students were starting the course in Software Engineering. Thus, we can observe the aspects that were evidenced by the game in the perception of the students (Figure 9).

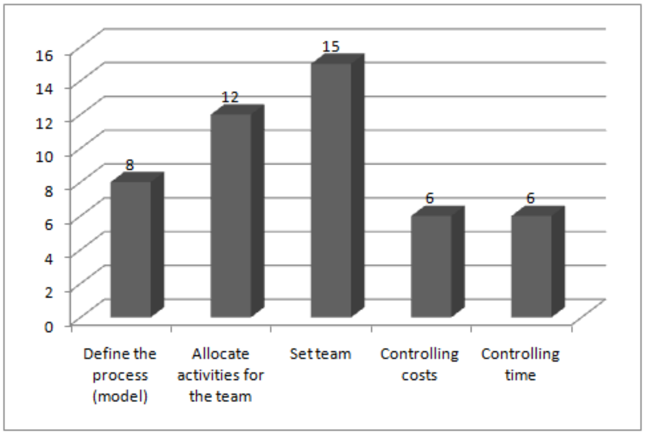


Figure 9. Aspects of project management evidenced by the game

In the collected sample, answers related to the life cycle models were analyzed by a paired T-test too. Statistical analysis indicates a significant difference on the life cycle models knowledge measured from pretest to posttest t(24) = 2,0687, p = 0,0654 (M_{pre} = 10.42; M_{post} = 10.79). That is, the SE•RPG had a statistically significant impact (significance level of 90% for a twenty four sample) on students' gains in life cycle models.

As shown in Figure 10, we can observe that 8 students obtained a better performance in the posttest, while 12 students showed no improvement and 4 students presented an inferior score in posttest.

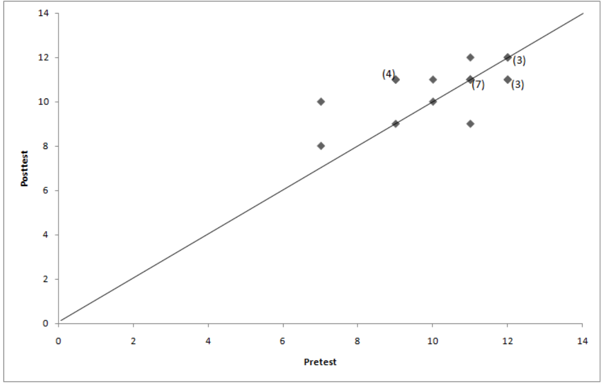


Figure 10. Scatterplot of pretest versus posttest scores which relates to questions about life cycle models. *The number next to some points refers to the number of students who achieved the same scores.*

In the test 12 characteristic of the different models were presented, and the student had to identify for each characteristic which was the model indicated. Differently the previous question, in this one the students had received a class explaining the models before the test. We believe that this fact was crucial to the difference presented in the evaluation of hypotheses H01 e H02.

The students completed a questionnaire stating their thoughts and feelings about the game in general (Figure 11) and their opinions about the pedagogical effectiveness of the game in teaching Software Engineering process and project management issues. In relation to the pedagogical effectiveness, 63% of students felt that the game is motivating and challenging. Moreover, 95% of students considered that the game provides practical insights of the concepts studied, and 22% of students indicated that the game has contributed significantly to the learning process.

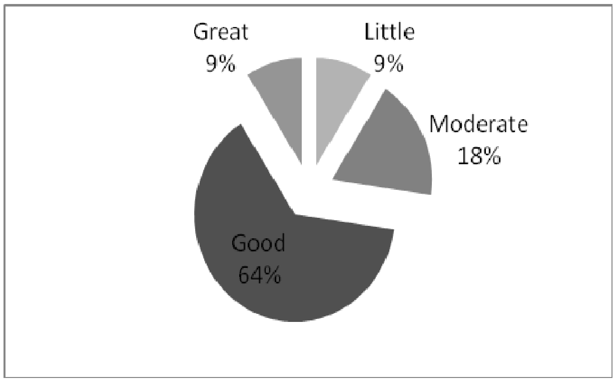


Figure 11. Results of the evaluation by students of the game

VI. DISCUSSION AND CONCLUSION

Evaluated tests and validations performed in classroom evidenced SE•RPG has potential for learning improvement, by the simulation of a virtual development environment to stimulate learning. The 'practical vision' of the differences between the models of life cycle presented in the game was mentioned by ninety-five percent of the students during the test, while the motivation and challenging aspect was identified by sixty-three percent of the sample, which confirms acceptance of the tool in the classroom context.

Through a Statistic test it was possible to determine a tendency in the paired samples of obtained data, that concludes the proposed hypothesis that the game has a significant contribution to the learning. The results showed a greater contribution to the learning in project management than the life cycle models. This difference is possibly related to the fact that students had taken classes related to the content models, while the management issue was not approached in the classroom at any time. In the experiment conducted without any explanation related to content management projects, we can identify which aspects are being evidenced in the game with regard to project management. Figure 8 shows that the issues highlighted by Fewings [10] as the key dimensions of project management were perceived by students. The dimension of the task was mentioned by 12 students (represented by "Allocate activities for the team"). The dimension of resources was mentioned by 21 students (considering "set team" and "controlling costs" as aspects of the resource dimension). The time dimension was mentioned by only 6 students ("controlling time"). In addition to the dimensions, 8 students considered the process definition as a function related to the project manager. This perception is correct and may indicate that students perceived the concept referring to a process.

It is important to consider that the level of the concepts approached in the game is still preliminary. The game is recommended as a complementary tool to the teaching of life cycle models and basic concepts of project management. However, based on the results, we believe that the game can help students having a more practical view on the issues discussed, and also retain the concepts involved.

The actual result is considered encouraging; however, new experiments with a larger sample should be performed. We consider important to conduct further experiments containing control group for comparison of results. In addition, it is important to improve the evaluation questionnaire including more questions.

This validation encourages continuous development of the tool to comprehend the contents inherent to the models of life cycle and project management, which are proposed as future works, as follow: (i) include another process models, such as spiral model, not considered at first; (ii) a tool to edit game modules to aid the professor in developing adequate challenges to the teaching methodology used (for now the professor must edit an XML file); and (iii) enhance the feedback provided to students at the conclusion of the game, trying to analyze the strategy adopted and comment on their performance.

VII. ACKNOWLEDGMENT

The author wishes to thank all the involved in the development at SE•RPG - Jefferson Mólleri and Felipe Ambrosio.

VIII. REFERENCES

[1] L. Ohlsson, and C. Johansson, “A Practice Driven Approach to Software Engineering Education”, IEEE Transactions on Education, vol. 38, no. 3, 1995, pp. 291-295. doi: 10.1109/13.406508.

[2] E.O Navarro, and A. Hoek, “Towards Game-Based Simulation as a Method of Teaching Software Engineering”, Proc 32nd ASEE/IEEE Frontiers in Education Conference (FIE 2002), Boston, MA, Nov. 2002, doi: 10.1109/FIE.2002.1158674.

[3] A. Baker, E. O. Navarro, and A. Hoek, “An experimental card game for teaching software engineering”, Proc. Conference on Software Engineering Education and Training (CSEE&T 2003), Spain, Madri, March 2003.

[4] S. El-Shamy, “Training games: Everything you need to know about using games to reinforce learning”,. Sterling: Stylus Publishing, 2001.

[5] O. Barreteau, C. Le Page and P. D'Aquino, “The joint use of role-playing games and models regarding negotiation processes”, Journal of Artificial Societies and Social Simulation, vol. 6, no. 2, Mar. 2003.

[6] I. Sommerville, “Software engineering”, Eight Edition, Addison Wesley, 2006.

[7] P. Jalote, “An integrated approach to software engineering”. Third Edition, New York: Springer, 2005.

[8] R. S. Pressman, “Software engineering: A practitioner's approach”, Seventh Edition, McGraw-Hill, 2009.

[9] Rational. “Rational Unified Process: release notes (version 2002.05.00)”, Rational Software Corporation, 2002.

[10] P. Fewings, “Construction project management: an integrated approach”,Oxon: Taylor & Francis, 2005.

[11] M. Prensky, “Digital Game-Based Learning”, McGraw-Hill, 2001.

[12] E. Navarro and A. Hoek, “Multi-site evaluation of SimSE”, Proc. 40th ACM technical symposium on Computer science education (SIGCSE), pp. 326-330, NY, Mar. 2009, doi:10.1145/1539024.1508981

[13] A. Steele, “Towards Next Generation Project Management Simulation”, Proc. Twenty Second Annual Conference of the National Advisory Committee on Computing Qualifications, pp. 115-119, Napier, New Zealand, Jul. 2009.

[14] Wizards of the Coast. d20 Modern System Reference Document. Retrieved from http://www.wizards.com/default.asp?x=d20/article/srd35, Dec. 2010.