

Semiotic Inspection Method in the Context of Educational Simulation Games

Daniela C. C. Peixoto

Raquel O. Prates

Rodolfo F. Resende

Departamento de Ciência da Computação (DCC) – Universidade Federal de Minas Gerais (UFMG)

Av. Antônio Carlos, 6627 – Pampulha – Belo Horizonte – MG – Brasil

{cascini, rprates, rodolfo}@dcc.ufmg.br

ABSTRACT

Software process simulation is increasingly being used as an approach for analyzing complex business, for supporting management planning, for helping with software process training and learning and for supporting the software process improvement. In addition to providing to users a simulation tool that supports all these aspects, it is also important to consider some other requirements during the tool's design, such as efficient and effective communication of the designer's message to the user. In this way, we show how semiotic concepts can be used in the analysis and generation of knowledge through the application of the Semiotic Inspection Method (SIM), a semiotic engineering evaluation method. In this paper we present a scientific application of SIM to a Software Engineering simulation game focusing the analysis on feedback aspects and issues. The results go beyond the system analyzed and point to considerations regarding simulation games used in educational contexts.

Categories and Subject Descriptors

D.2.2 [Design Tool and Techniques]: User Interfaces

General Terms

Design and Human Factors

Keywords

Simulation, Game, Semiotic Inspection Method, communicability

1. INTRODUCTION

For an adequate Software Engineer training and formation, the Software Engineering courses should reduce the gap between learning by studying and learning by doing [9]. The academic community has recognized the industrial dissatisfaction with student preparation [1], and in response to it, has created some practices that try to be closer to what happens in real organizations [11]. These practices include focusing on case study analysis of large projects, training in integrated industry-classes and using simulation games [9].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'10, March 22-26, 2010, Sierre, Switzerland.

Copyright 2010 ACM 978-1-60558-638-0/10/03...\$10.00.

There are some empirical evidence showing that simulation games can be an effective tool for enhancing learning and understanding of complex subjects [3], [16]. In this context, simulation can be used as a powerful tool to practice Software Engineering concepts that are difficult to practice in classes [2].

Simulation games are more enjoyable and fun when they provide sufficient challenge for the player [10]. Games should engage student's interest and should be usable to permit results that are more clear-cut and enjoyable. Therefore, important aspects of the games must be evaluated, in particular its communicability. Communicability is a concept proposed by Semiotic Engineering Theory [4] and is defined as "the property of software that efficiently and effectively conveys to the users its underlying design intent and interactive principles" [4], [15].

In this paper we describe our work about how the communicability of a simulation game can be improved. The goal is to achieve efficient and effective communicability through the game interface. Our main result is a list of aspects that should be observed during the design of educational simulation games.

There are two distinct methods to evaluate the communicability of an interface: Semiotic Inspection Method (SIM) [4], [5], [6], [7] and Communicability Evaluation Method (CEM) [4], [15]. Both are qualitative and interpretative methods. The use of CEM requires expensive experiments since it requires specialists to observe users in a tightly controlled environment. In our work we decided to use SIM since it is an inspection method and not so expensive that provides results comparable to CEM. As described by de Souza et al. [7] SIM has a technical and a scientific application. Technical application of SIM focuses on how a method can improve and inform interaction design in the context of a specific system development; whereas scientific application focuses on how a method can broaden the knowledge of Human Computer Interaction (HCI). In our work we perform a scientific application of SIM in order to evaluate the communicability of a Software Engineering educational simulation game, specifically regarding its feedback. As a result we identify feedback aspects and issues relevant not only to the system evaluated, but for simulation games used in educational contexts in general.

The remainder of this paper is structured as follows. Section 2 presents a brief overview of Semiotic Inspection Method. Section 3 describes the case study. Section 4 concludes and suggests future work.

2. A SEMIOTIC INSPECTION METHOD OVERVIEW

The Semiotic Inspection Method has been proposed within the Semiotic Engineering theoretical framework. Semiotic

Engineering perceives HCI as a designer to user metacommunication. That is, a twofold communicative process, because the designer to user communication is achieved through user-system communication.

The system's interface is a complete and complex message sent from designers¹ to users. This message is formed by signs. According to Peirce, signs are anything that stand for something (else) in someone's perspective [14]. The message conveys to users the designers understanding of whom the users are, what problems they want or need to solve and how to interact with the system in order to achieve their goals. This designer-to-user communication is indirect, since users understand the intended message as they interact with the interface. When users do not understand aspects of this message, a communication breakdown takes place [4].

Semiotic Engineering theory classifies the signs in an interactive system into three classes of signs: metalinguistic, static and dynamic [6], [7]. *Metalinguistic signs* are signs that refer to other interface signs. They are instructions, tips, online help, error and informative messages, warnings and system documentation. They are signs that the designer uses to explicitly communicate to users the meanings encoded in the systems and how they can be used. *Static signs* express and mean the system's state, they are motionless and persistent when no interaction is taking place. They can be perceived (and interpreted) in snapshots of the system's interface before or after interaction occurs. For instance, buttons, text areas and check boxes at a given moment. *Dynamic signs* express and mean the system behavior. Their representations unfold and transform themselves in response to an interactive turn. For example, if we click on the search button the behavior will present the results of a search. This behavior is a dynamic sign.

The Semiotic Inspection Method examines the designer's metacommunication that users are exposed to as they interact. The goal is to identify communication breakdowns that may take place during the user-system interaction and to reconstruct the designers' metamessage being conveyed by the system. To do so the evaluator analyzes the message being transmitted by signs at each level: metalinguistic, static and dynamic.

This method is carried out in five distinct steps [5], [6], [7]: (1) an inspection of metalinguistic signs; (2) an inspection of static interface signs; (3) an inspection of dynamic interaction signs; (4) a contrastive comparison of designer-to-user metacommunications identified in steps (1), (2), (3); and finally (5) a conclusive appreciation of the quality of the overall designer-to-user metacommunication. At the end of steps (1), (2) and (3), the evaluator reconstructs the metamessage being conveyed by signs at that level, filling out the template of the designer to user communication. The evaluator also identifies potential communicability problems that may take place at that level. In step (4) the evaluator contrasts the metamessage generated in the previous steps and checks for inconsistencies or ambiguities among them. It is not expected that the messages generated by each sign level to be identical², but they should be consistent.

¹ In this paper designers should be interpreted as whoever speaks for the design team.

² Signs at each level have distinct expressive possibilities. For instance, natural text used at the help system – metalinguistic

Finally, in step (5), the inspector reconstructs a unified metacommunication message, judging the costs and benefits of communicative strategies identified in previous steps and generates the evaluation report.

Like other methods, SIM requires a preparation step. In this step, the evaluator defines the purpose of the inspection, does an informal inspection in order to define the focus of the evaluation, navigates through the system and finally elaborates the inspection scenarios. If the method is being applied scientifically, this step should also produce a clear statement of the research goal and an examination of how it could be achieved by using SIM [7].

The scientific application of SIM helps researchers in diagnosing the system's problematic situations and it also provides theoretical concepts that support the formulation of problems as research questions [5], [7]. In a scientific context, SIM also includes a final triangulation step to validate the results [7]. The validation step corresponds to a triangulation of results against evidences obtained with other methods or procedures. The triangulation can be carried out with empirical evidences provided by endogenous and exogenous sources. The endogenous triangulation is obtained with reference to different aspects of the same system or systems in the same domain. In this triangulation, the evaluators search for interpretations that are consistent with their analysis. The exogenous is carried out with reference to systems in other domains.

The scientific application of SIM is recent and it was firstly applied to investigate the feedback of a system. In this study [7], the research question addressed was: "how the system's feedback is communicated to users during interaction". The case studies were carried out with a simple editor of cascading style sheets (Simple CSS[®]) and Google Groups[™]. The results of the Simple CSS study were endogenously triangulated with findings of web material (FAQ, users' opinions, etc) about CSS editors of the same kind and exogenously triangulated with an analysis of system's feedback to users' actions in creating and configuring a group in Google Groups[™]. The results presented in de Souza et al. [7] show that it is possible to achieve a more powerful metacommunication using different sign types (namely, icons, indices and symbols) that simultaneously and consistently mean the same feedback. One important value of this scientific application of SIM is to associate feedback issues with end-users specification, designing and programming tasks.

Feedback is a critical element during interaction; because it allows players to interpret the software and the designers' message. In an educational game, it provides information for novice players during the learning process as well as supporting players' decisions throughout the game [13]. Thus, the results presented in de Souza et al. [7] and the importance of simulation games to educational contexts [13] has motivated our investigation of feedback issues specific to this context.

3. CASE STUDY

In order to investigate feedback issues on educational games a Software Engineering game, SimSE³ [12], [13], was chosen and

sign – and an icon at the toolbar – static sign – usually do not express the same content equally well.


³<http://www.ics.uci.edu/~emilyo/SimSE/>

SIM was scientifically applied to this system. SimSE was chosen because it is a free tool and, according to its authors, it is an enjoyable educational game and relatively easy to play.

It is a customizable game that allows instructors to build different types of software process models. SimSE is a single-player game in which the player takes the role of a project manager of a team of developers. The player's goal is to develop a software project within the established budget, schedule and with reduced number of defects. SimSE was designed in such a way that the software project it simulates can have several different subjacent processes, models or approaches: Waterfall, Incremental, Code Inspection, Rapid Prototyping, Rational Unified Process and Extreme Programming. Unfortunately, the model choice is not part of the simulation and must be made in advance before the simulation starts [12], [13].

At the beginning of the game, the player can see a description of the Software Engineering tasks, the goal of the game, how much time or money is available and some directions on score penalties and power-ups. The player can assign tasks, hire or fire the employees, monitor the progress and purchase tools. In this work, we focus on the educational software engineering simulation environment of SimSE (Figure 1), and we do not consider the Model Builder, a tool that allows the specification of software engineering process models [13].

We will evaluate two kinds of feedbacks provided by the tool: informative feedback and performance feedback [10]. Informative feedback presents information regarding the effects of the player's action on the project or team and it aims at maintaining player's curiosity and interest in the game. Performance feedback informs players how close to achieving their goal they are.

Before we proceed with the Semiotic Inspection Method, it is important to present briefly some semiotic concepts that will be used in our analysis. Namely, a semiotic classification of signs types [4], [14] – *icons*, *indices* and *symbols*, and the semiotic phenomenological categories: *firstness*, *secondness* and *thirdness*. *Icons* are signs where representation evokes the *firstness* of its referent. A phenomenon of the first category brings up a unary quality experience of sensing the referent. So when we represent a cow with a sign like this , we evoke the visual experience we have when encountering cows in general. *Indices* are signs where the representation brings out the *secondness* of its referent. When we take a smoke as a representation of fire, this evokes a certain kind of association (smoke indicates fire). The idea of *secondness* presents some casual association between concepts. Finally, *symbols* are signs which representation evokes the *thirdness* of its representation. Symbols result from a regulating convention that relates them to their referent by virtue of an established meaning. For example, '®' conventionally refers to a registered trademark.

3.1 Preparation

As mentioned in Section 1, in this work we evaluate how feedback is communicated to SimSE players. The goal of this analysis is to identify, in a broader context, feedback issues on educational simulation games.

In the preparation for SIM application, we elaborated a scenario for SimSE evaluation. This inspection scenario describes an undergraduate student that is coursing a Software Engineering discipline and is motivated to use SimSE as part of an extra activity assigned by the professor:

Michael is a student of a Software Engineering discipline. This discipline is offered to undergraduate computer science students at the Federal Knollo University. At Knollo University, the duration of each discipline is 4 months, totalizing 60 hours. At this moment, Michael's Professor is presenting the last topic of the discipline: software test. They already have seen requirements, designing, implementation, software project, software configuration and software quality. Michael and his colleagues have some doubts about the real application of the concepts learned in class, so their Professor decided to use a software simulation tool that can reduce the gap between academic and industrial world. The Professor has found a free simulation tool available on the Internet, called SimSE. He hopes that the Software Engineering discipline can be improved by using this tool, which will bring "realism" to classes. SimSE allows simulation with different kinds of models, for example, Waterfall model, Incremental model, Rapid Prototyping model and Rational Unified Process model. It has already been tested and used in classes in some universities. The Professor gave a 10 open question questionnaire, as an optional-exercise, to his students about the Waterfall model represented in SimSE. As Michael didn't get good grades in the previous exams, he decided to answer this questionnaire. Michael visited the web-link provided by the Professor and downloaded the software. This is the first time that Michael uses this software.

In our scenario, Michael, the central character, is motivated to understand how this game works and to answer correctly the questionnaire provided by his Professor.

SIMSE inspection using SIM was conducted based on the scenario described. We next present the main results from each one of the methods' steps.

3.2 Analyzing the designers' message

The analysis of the **metalinguistic signs** allows us to identify important elements used in the designers' discourse. The metalinguistic signs are used to provide a better understanding of the static and dynamic signs, and the intended relation among them. To analyze the metalinguistic signs the downloading site, manual, dialogs, messages and tutorials were inspected in regard to what they conveyed about the system's feedback (focus of investigation). The gist of what SimSE's designers intend to communicate to players focusing on feedback can be summarized in 5 messages, where 'we' means 'we, the designers' and 'you' means 'you, user, or in this case, Michael, the player' :

- a) If 'you' want more information about the functionalities of the game, 'you' should search in the Players Manuals. Since, there is no online help or tooltips available.
- b) 'You' can see some informative feedback through speech bubbles over the employees' head. These bubbles convey important information about the effects of the actions and also provide some guidance.
- c) 'You' can see some informative feedback of the employees and the resources (artifacts, customers, employees, projects and tools). Employees' information appears in the 'current activities' area. The status of the resources appears on the bottom table (Figure 1) or in a pop-up window.
- d) 'You' can only see 'your' performance feedback (score) at the end of the game.

- e) 'You' can learn how to improve 'your' score reading the rules used in the game through the Explanatory Tool. This tool allows 'you' to generate graphs of various attributes and discover insights of the rules underlying the game.

During the analysis of the **static signs** (step 2), important signs were identified at the main interface of the game (Figure 1). We can organize parts of designers' message content these signs convey regarding feedback aspects as:

- a) 'We' simulate an office environment, and 'you' can notice that 'you' are in an office with doors, walls, furniture and employees (avatars).
- b) 'We' provide, as in real offices, information about the current status of the resources and the employees on tables.
- c) 'We' provide some important information related to the simulation. At the main interface, 'we' provide information and action related to the simulation time as seen on the lower right corner. In the Explanatory Tool, accessed through the Analyze menu item, 'you' can generate graphs and evaluate the rules of the game.
- d) 'We' consider that 'you' will be able to infer that 'i' stands for a short narrative about the goal of the game and that 'R' stands for simulation Reset, as appears on the left side of the SimSE logo.

Static signs focus on the presentation of the metaphor of a physical office and beyond that some restricted functionality related to the simulation time control (next event, advance clock, time elapsed).

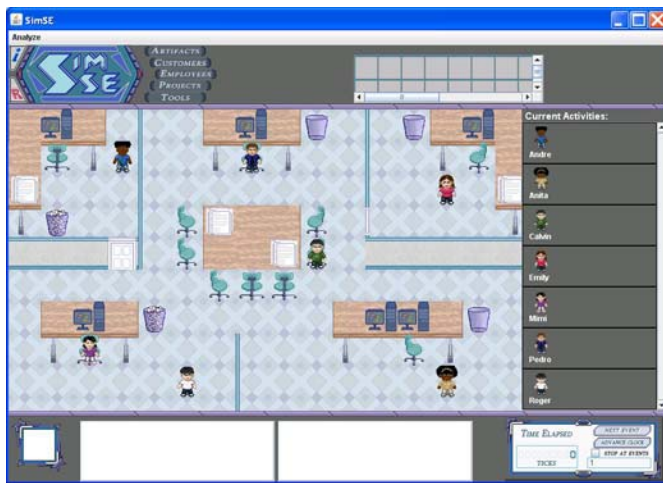


Figure 1. SimSE main interface.

The analysis of **dynamic signs** yields more interesting and informative results for our evaluation. With this analysis, it is possible to evaluate the dynamic aspects of the game and it is also possible to see the effects of some of the simulation parameters on the attributes. The gist of what SimSE's designers intend to communicate to players focusing on feedback through dynamic signs can be summarized in:

- a) 'We' use a metaphor of a physical office to determine the interaction 'you' may have with the game.
- b) 'We' consider that 'you' have some knowledge about Software Engineering, because of the signs 'we' used (e.g. the name of the activities: 'create the system test plan').

- c) 'We' provide four important feedback mechanisms: the tables, the activity list, the bubbles over the employees head and the Explanatory Tool. The tables present information about the resources and employees. The activity list presents the actions that the employees are performing. The activities executed by each employee are also presented at the bubbles over their heads. The Explanatory Tool is accessed through the Analyze menu item.
- d) 'You' can interact with the project team directly on the main interface. The employees are static avatars that communicate through pop-up bubbles. They have the same appearance throughout the game, independently of what happens and they remain at the interface, unless they are fired (only then do they disappear).
- e) During the game 'you' can see information related to the results of 'your' actions (e.g. employees' mood and energy). 'Your' score can only be seen at the end of the game.
- f) 'We' would like 'you' to play the game immediately after reading the project goal, using a 'trial and error' strategy (exploratory interaction), since 'we' do not direct 'you' towards any other information.

In the communication described in 'c', the avatars overhead bubbles can confuse the players when acting as a project manager. When they assign more than one activity to the employees, the order of execution is not represented. And the message presented by the employees, sometimes just lists the current activity they are performing.

As we **collate and compare** the metacommunication conveyed by metalinguistic, static and dynamic signs, focusing on feedback, we notice that the most effective communication of design intent is expressed by means of dynamic signs. This can be observed since the resulting metacommunication message after the analysis of dynamic signs incorporates all the concepts conveyed during the static and metalinguistic evaluation. As pointed out, the metacommunication is mainly achieved through a 'trial and error' strategy and the static signs have limited expressivity in transmitting the overall message. Regarding the metalinguistic signs, the players manual is not so limited, however, the tool does not offer any help during the interaction. For example, in the Explanatory Tool accessed through the Analyze menu item, clicking with the right button of the mouse over the generated graphs, allows the player to create multiple branches of the game. However, since this is not explained to users, and they must interact with the system to discover it, they may never perceive this possibility that can be useful in comparing the effect of different strategies on the project. Metalinguistic signs could help motivating players with this game, by providing them with the necessary information they need to better understand the game.

Although dynamic signs in SimSE support the 'trial and error' strategy, miscommunication gets in the way of the player's learning process. When players start the game some rules may not be clear to them. For instance, why inspecting code activity requires at least 3 members of the project team, how the pay rate affects the productivity of each employee, or what are the consequences of skipping an activity. Players can see some of these effects using the Explanatory Tool. However, the use of some functionalities (e.g. create branches) in the Explanatory Tool is not simple, as explained above.

Malone has listed three aspects that make games fun to learn: they should provide a challenge, they should use fantasy and they

should evoke player curiosity [10]. Although it is possible to observe some of these characteristics in SimSE, if someone tries to play SimSE again, it may not seem so enjoyable anymore, mainly because there is no new information at the game's interface and there are no levels of difficulty for experienced players. Players can change just the order of the selected activities, trying to achieve a better score.

3.3 Communicability Evaluation

During the metalinguistic and static evaluation, we observed that the designers clearly stated that the players will play the role of a project manager, however it is not communicated that the players must have basic knowledge of Software Engineering and this can affect their interest and motivation. Therefore, the knowledge level required to play the game should be clearly stated to the players, whatever it is.

In relation to feedback, the tool should support students in acquiring a more complete and consistent knowledge [10]. The informative feedback shows some status of the project, but the player can get confused with what actions would have a positive or a negative impact. If players do not read the manual they will not understand the goal of the Explanatory Tool that shows the rules of the game and it would be safe to assume that players rarely read manuals before playing. One improvement would be to allow students to access their performance indicators before the end of the game, or bring to the game a virtual consultant that could give hints (for example as a virtual software quality manager) to act as a wizard.

The informative feedback, such as the number of defects in an artifact and the money spent, is given to the player during the game. But the players learn about their performance only at the end of the game. So, it may take several game plays to understand the impact of decisions on the overall performance and obtain good results. The performance feedback should be previously presented in the way to enhance challenge and to minimize self-esteem damage. This is in accordance with some learning theories used in the game [13] as: Discovery Learning and Learning through Failure (which also reinforces the designer 'trial and error' strategy) and Constructivism (that reinforces the previous needed knowledge of the players). Otherwise, as discussed above, the game could fail to be enjoyable after a while.

At a closer examination, the problem is more complex. This game does not represent some signs of firstness that would be more effective to communicate and, in consequence, give direct feedback to the players. For example, when the energy of the employees reduces, they could gradually disappear from the interface, or if they were moody they could look different (for example, with a tired face). In all, some of these characteristics should be incorporated in this simulation to turn it into a more game-like tool. The feedback represented by secondness can also be improved if the cause-effect relations are better represented. We observed that the consequences of some actions are not directly represented in the elements and the player can get confused. One example is when an employee goes on a break, he/she does not disappear from the interface. Another point of confusion is the definition of the order of execution of some tasks assigned to an employee. The player, as project manager, should be able to change (or at least see) the priority of the tasks, when a project is not going well, but this is not supported by the game. Regarding thirdness, the designer uses some known signification

systems to communicate with users. For example, the names of the activities are those conventionally used tasks in software development process. This indicates the need for players to have previous knowledge of these signification systems, since the meaning of used signs is not available at the interface. However, the student who does not know any of these concepts could be frustrated with this software. Other important symbolic signs are the rules of the game. The rules can foster an important learning process, where players become skilled in anticipating the effects of choosing an action. If the players learned these rules, they would eventually be in a position to achieve a good score and understand the dynamics of a Software Engineering team. Therefore they should be well communicated to players within the context of the game, which does not happen in SimSE.

3.4 Triangulation

To validate our conclusion about the feedback communication in SimSE, a triangulation with empirical evidence provided by endogenous sources was performed. We found concrete evidence that players have missed the informative and performance feedback and that the prerequisites for playing the game are not clear [13].

Evidence 1: *"I still don't really understand what the score is based on."* and *"I'm not really sure exactly what the scoring criteria are."*

These are comments from students that do not use the Explanatory Tool. And for them, it was not possible to understand the rules that led to a certain score.

Evidence 2: *"Rules were a major help. The rules are really helpful-even if someone doesn't know anything about Software Engineering I think rules can teach you how to play the game."* and *"SimSE's explanatory tool is a useful resource for helping players understand their score, but its value lays primarily in its rule descriptions."*

Students that used the Explanatory Tool think that rules can be useful in learning how to play the game and understanding the score. But, yet it is not possible to know how the score is calculated.

Evidence 3: *"[SimSE is] a good way of putting concepts into practice."* and *"It didn't help so much compared to what I already know."*

These comments show that the player should have some knowledge of Software Engineering to play the game.

An important result, that reinforces the first scientific application of SIM [7], is that the consistent use of **icons**, **indices** and **symbols** that refer to the same feedback (redundancy) may improve the designer to user metacommunication. A comparison among the results of SimSE and those of Google Groups and SimpleCSS (described in [7]) reveals that the three evaluations have much in common. The first characteristic is the importance of the perceptible change of the **iconic** representation of the elements after the changing of their attributes (e.g. the employees' visual appearance). Another important characteristic is the correct visual effect of the **indices** (for example, after being fired the employee would disappear from the interface). In addition, an explicit and consistent symbolic representation of partial results and strategies used to calculate them could improve the feedback, considering that only through the use of icons and indices it would be virtually impossible to communicate such aspects.

4. CONCLUSION

This article presents the steps using SIM and the analysis of feedback in an educational simulation game. The goal was to identify issues regarding efficient and effective communicability conveyed through the game's feedback.

The analysis was performed on SimSE, a Software Engineering simulation game for software process education. In this analysis we observed that designers of SimSE focused on the informative feedback and use of 'trial and error' strategy. Dynamic signs are privileged in designer-to-user communication and convey most aspects of the whole message. However, it seems that designers should explore other playability characteristics [8] in the game, such as varying the levels of difficulty. This would require much more elaborated signs, and it would probably provide players a more productive experience.

We can conclude that some important aspects should be observed during the design of educational simulation games:

- Designers should represent some signs of firstness that would give direct feedback to players. In some situations, this feedback can be more effective to the learning process than using tips or on-line helps. For example, in SimSE, the modification of the avatars' state after some action (they disappear, move or look different).
- Secondness signs that can communicate cause-effect relations are also important, since players can quickly visualize the results of the selected actions.
- Designers should use thirdness signs, since rules and strategies are conventions defined by designers and cannot be fully communicated through firstness and secondness signs.
- The designers should provide an efficient and effective way to present the rules used in the game and their relation to the players' performance. In educational games, this would support the learning process, and players could become skilled in anticipating the effects of an action and correcting them when necessary.
- It is important to have both informative and performance feedback during the whole game. This provides challenge for the players motivating them.

As future work, we plan to triangulate the results obtained with exogenous sources. We believe that this triangulation might expand the initial interpretations of the inspectors, because it would be carried out with reference to educational simulation games including some in other domains.

5. ACKNOWLEDGMENTS

Daniela Peixoto thanks Capes Foundation (grant n. BEX 1893/09-2) and Raquel Prates thanks Fapemig for the support to their research.

6. REFERENCES

- [1] Callahan, D. and Pedigo, B. *Education Experienced IT Professionals by Addressing Industry's Needs*. IEEE Software, 2002. 19 (5): p. 57-62.
- [2] Claypool, K. and Claypool, M. *Teaching software engineering through game design*. In Proceedings of the 10th Annual SIGCSE Conference on innovation and Technology in Computer Science Education. ITiCSE '05. Caparica, Portugal, June 27 - 29, 2005.
- [3] Cordova, D. I. and Lepper, M. R. *Intrinsic Motivation and the Process of Learning: Beneficial Effects of Contextualization, Personalization, and Choice*. Journal of Educational Psychology, 1996. v88, 715-730.
- [4] de Souza, C. S. *The semiotic engineering of human-computer interaction*. MIT Press, Cambridge, 2005.
- [5] de Souza, C.S., and Leitão, C.F. *Semiotic Engineering Methods for Scientific Research in HCI*. Synthesis Lectures Series, Morgan & Claypool, San Francisco, 2009.
- [6] de Souza, C. S., Leitão, C. F., Prates, R. O., and da Silva, E. J. *The Semiotic Inspection Method*. In: Proceeding of the 7th Brazilian Symposium of Human Factors on Computer Systems (IHC'2006). Porto Alegre, SBC, 1, 148-157, 2006.
- [7] de Souza, C. S., Leitão, C. F., Prates, R. O., Bim, S.A., and da Silva, E. J. *Can inspection methods generate valid new knowledge in HCI? The case of semiotic inspection*. International Journal of Human-Computer Studies. To appear (2009).
- [8] Desurvire, H., Caplan, M., and Toth, J. A. *Using heuristics to evaluate the playability of games*. In CHI '04 Extended Abstracts on Human Factors in Computing Systems (Vienna, Austria, April 24 - 29, 2004). CHI '04. ACM.
- [9] Ludi, S. and Collofello, J. *An analysis of the gap between the knowledge and skills learned in academic software engineering course projects and those required in real projects*. Frontiers in Education Conference, 2001. 31st Annual.
- [10] Malone, T. W. *What makes things fun to learn? heuristics for designing instructional computer games*. In Proceedings of the 3rd ACM SIGSMALL Symposium and the First SIGPC Symposium on Small Systems. Palo Alto, California, United States, September 18 - 19, 1980.
- [11] McMillan, W. W. and Rajaprabhakaran, S. *What Leading Practitioners Say Should Be Emphasized in Student's Software Engineering Projects*. In Proceedings of 12th Conference on Software Engineering Education and Training, 1999. IEEE Computer Society, p -177-185.
- [12] Navarro, E. O. and van der Hoek, A. *SimSE: an educational simulation game for teaching the Software engineering process*. In Proceedings of the 9th Annual SIGCSE Conference on innovation and Technology in Computer Science Education. ITiCSE '04. Leeds, United Kingdom, June 28 - 30, 2004.
- [13] Navarro, E. *SimSE: A Software Engineering Simulation Environment for Software Process Education*. Doctoral Dissertation, Donald Bren School of Information and Computer Sciences, University of California, Irvine, 2006.
- [14] Peirce, C. S. *The essential Peirce* (Vols.I and II). Edited by Nathan Houser and Christian Kloesel. Indiana University Press, Bloomington, 1992.
- [15] Prates, R. O., de Souza, C. S., and Barbosa, S. D. *Methods and tools: a method for evaluating the communicability of user interfaces*. Interactions 7, 1 (Jan. 2000), 31-38, 2000.
- [16] Ricci, K. E., Salas, E., and Cannon-Bowers, J.A. *Do Computer-Based Games Facilitate Knowledge Acquisition and Retention?* Military Psychology, v8 (4), 295-3, 1996.