Using Computer Games to Teach Design Patterns and Computer Graphics in CS and IT Undergraduate Courses: Some Case Studies

Ismar Frango Silveira[1], Pollyana Notargiacomo Mustaro[1], Luciano Silva[1]

[1]Faculdade de Computação e Informática, Universidade Presbiteriana Mackenzie (UPM)

{ismar, polly, lucianosilva}@mackenzie.br

Abstract. *The acquisition of skills related to creativity and abstraction, crucial to any Computer Science (CS) and Information Technology (IT) curriculum, constitutes unquestionably a didactical challenge. From the educational point of view, the use of games in such process constitutes a motivational element that could help to make the knowledge building process more personalized. To show the possibilities of this educational scenario, two case studies were conducted in CS and IT undergraduate courses. The results obtained in these processes demonstrate the effectiveness of using instructional design elements to elaborate a proposal of applying computer games in undergraduate courses in order to motivate students to learn non-trivial subjects in a ludic way.*

1. Introduction

From the educational viewpoint, the use of games in teaching-learning process constitutes a motivational element that can help to make knowledge building process more individualized. It is also feasible to consider students' learning styles, thus setting up an adaptive and flexible environment where any skill, topic or even concept could be effectively learned (Prensky, 2007).

Besides, according to Tapscott (1998), nowadays, undergraduate students belong to net generation and usually play different kinds of games in consoles, computer, and Internet. Such so-called "N-geners" could be characterized by having autonomy sense, intellectual overture, technology inclusion (or facility to use technological elements even though never having any previous contact with them), freedom of expression, curiosity, immediacy, and mainly trust on themselves. This scenario perfectly fits into a game universe-based educational proposal, since nowadays it is required to institute mechanisms that take advantage of technological culture over where they are steeped and transform it in learning resources.

Metaphorically speaking, professors could be considered similar to game masters that guide and encourage players (the students) into a game (the educational process itself) to play the game, face the challenges and go through the next level (in other words, succeed in academic life). By exceeding curriculum demands, students could be able to enjoy a lifelong, meaningful learning experience (Ausubel, 1962).

However, teachers and students barely consider games as something detached from entertainment. The sole tentative of introduce “serious”, non-entertaining games in curriculum often causes the inverse effect, since this sort of games tend to be tedious, since they do not prime for the entertainment-related aspects that are responsible for retaining students’ attention.

It must be remembered, although obvious, that the act of learning does not have to be a boring, unexciting situation to what students are exposed for a significant part of their lives. Oppositely, it must be a stimulating and – why not? – funny, entertaining activity to be performed by students. Thus, recovering the ludic side of learning is primordial to motivate students to learn the issues curricula tell that they have to.

Specifically in Computer Science and Information Technology area, students often are already gamers, so that they are completely aware to game strategies, terminologies and playing. The introduction of game-related situations in their curricula has being a smooth and well-accepted operation, since games belong to their cognitive comfort zone. Thus, given such familiarity with the pedagogical instrument – the game – even uncomfortable, hard to be taught syllabi could make use of it in order to approximate curriculum subjects to students’ social context.

This paper is organized as follows: next section discusses the theoretical foundation of the research; third section presents two case studies that were taken in the context of undergraduate courses in Universidade Presbiteriana Mackenzie. Some conclusions and directions for further work are provided at the end of the paper.

2. Theoretical Background

The approach applied in the conducting of both case studies is mainly based on Problem-Based Learning (Barrows and Tamblyn, 1980; Barr and Tagg, 1995; Wilkerson and Gijselaers, 1996) which is an instructional theory that motivates students to work in a collaborative way to propose solutions to real-world problems. In both cases, students were faced to different sorts of game-related problems.

Using games with an educational proposal requires a completely different instructional design approach. In this case it is necessary to consider a large range of aspects: learning objectives, objects and styles, as well as some structural course elements, as methodology, evaluation, outcomes and feedback (Mustaro et al., 2007). Having in mind the game development process itself, a proper usage of games in education would deal with distinct elements like narratology, storyboarding and creativity. More than being just useful for learning, game-based education presupposes that the whole process ought to be motivating, challenging, enjoyable, in order to engage learner in a process where learn and play are tangled into a sole action.

Jesper Juul (2005) points out that games present essentially two forms for providing challenges for players: first is based on combination/variation of a reduced set of rules (called *emergence*); second is related to level-oriented game structure and a well-defined sequence of events (denominated *progression*). However, the act of bringing the game universe to education leads to the need of considering also previous knowledge and skills students have in order to present them challenges that match the requirements present in the definition of Zone of proximal development (ZPD) (Vygotsky, 1978).

More than just providing games to N-geners students, it is relevant to motivate them to develop their own games, particularly in Computer Science (CS) and Information Technology (IT)-related undergraduate courses as. This approach means to merge real-world problem solving, as described by Shaffer (2007) with previously described PBL. From this point of view, it is relevant to stand out proposals as *Digital Zoo* epistemic games where expertise development and community of practice are constantly stimulated in an environment that permits to design creatures as mechanical and biomechanical engineers (Shaffer, 2007).

According to Norman and Schmidt (2000) the use of PBL in curricula could improve knowledge retention, students' motivation, self-learning skills, besides it helps students to develop the skill of transferring concepts and procedures learned to new scenarios. Thus, the approach followed in this research uses games to motivate a PBL-based strategy to increase students' intrinsic motivation, allowing them to experience some significant learning situations.

3. Case Studies

The first case study was carried through years 2005-2006 with three different groups of students from 4th year of a CS undergraduate course, involving approximately 120 students. The second case study was experienced with nine different groups of IT undergraduate students, during the same years, involving around 550 students.

In the first case, students were meant to have a Software Engineering-driven, object-oriented vision about games, having to propose different solutions for the problem of modeling games, using broadly General Responsibility Assignment Software Patterns – GRASP (Larman, 2006) and GoF Design Patterns (Gamma et al., 1995).

The main purpose behind patterns in general, and Design Patterns (Gamma et al., 1995) in the specific case, is to provide to software development community a set of problem-solution pairs in order that they are supposed to be generic enough to serve as guidelines to solve recurrent problems in different domains. However, Design Patterns' learning process demands students to have a high level of abstract reasoning, besides a certain degree of maturity on Software Engineering issues, which makes this task a non-trivial effort to be performed.

The first step to motivate students to use of GRASP concepts was based on the proposal of choosing a game and building a class diagram to represent it, besides applying two object-oriented metrics: LCOM and CBO. These metrics were chosen from the well-known set of metrics proposed by the classical work of Chidamber and Kemerer (1994) and they are meant to support the application of two GRASP patterns, respectively High Cohesion and Low Coupling (Larman, 2006).

One of the students choose a classic game called Tetris (from Russian Тетрис), which pieces are composed of four square blocks, called *tretaminoes*, that fall down on the playing field. One result devised from the proposed experiment is depicted on the next figure:

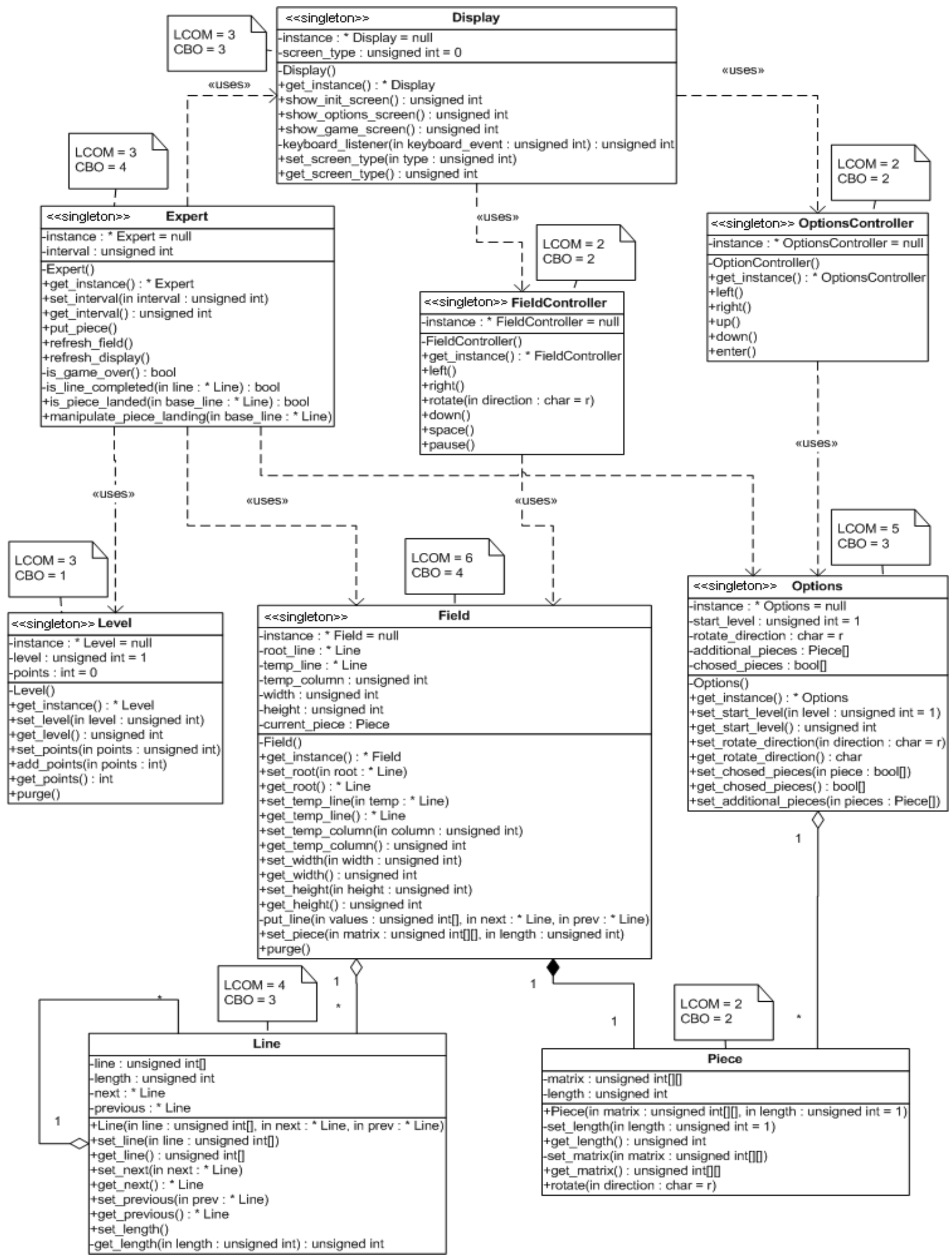


Figure1 – A class diagram with GRASP and object-oriented metrics

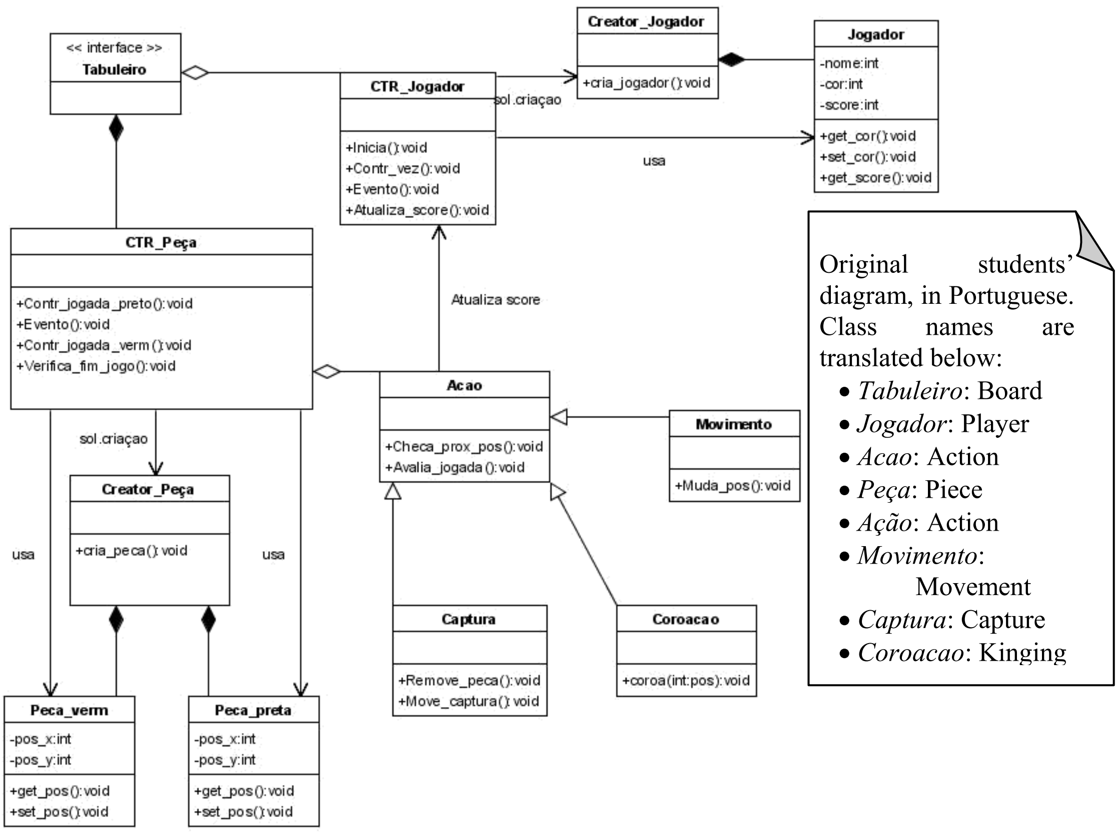
The proper application of GRASP could lead students to a satisfactory game project. The applying of PBL and Collaborative Learning lead students to build these solutions in a self-directed basis. Other significant element was criticism, supported by LCOM and CBO metrics, which permit students to decide if they had made the correct choices during design phases.

This first approach constitutes a basic level that prepares students to deal with more sophisticated concepts, like Design Patterns. Actually, the study and use of Design Patterns, as defined by Gamma et al. (1995) and Alur et al. (2003), among others, has been considered as standard in the core of undergraduate courses (Astrachan et al., 1998; Wick, 2005). In Brazil, for instance, the official Curricular Guidelines barely mention Design Patterns (Menezes et al., 2001).

To satisfactorily understand and apply design patterns involves a set of previous required skills of students. Many of them were developed in previous phase, for instance, the awareness of cohesion and coupling issues, separation of concerns, among others. Nonetheless, the results produced by students are not yet satisfactory if compared with professional expectances of complex software.

More than just modeling, students were also invited to criticize colleagues' works. Again, the fact of using popular games as subjects of modeling was decisive, as their domains are widely well-known by everyone. By means of criticism and controlled intervention, students were able to discover by themselves, in the application domain of the game, some of the patterns taught.

An example of criticism and controlled intervention mentioned above can be seen in Figure 2, showing a class diagram elaborated by group of students, evolving from another diagram previously presented in classroom. The lower part of the figure shows an intervention made by the teacher through virtual environment.



(Professor): This new version for the diagram is better than previously presented in classroom. However, there are still some issues to be addressed:

- *What's the need of having a class for black pieces and another one for reds?*
- *There is no relationship among Player, Action and Piece?*
- *Coupling of CTR_Piece is over diagram's average*

Figure 2 – An early class diagram for Checkers game, followed by teacher's intervention

Using students criticism and some punctual interventions of professors, students were able to absorb the main concepts involving such set of patterns and to discover by themselves some of these patterns. For instance, Figure 3 shows part of a diagram that models the Checkers game, made by the same students that created the diagram Figure 2, after many iterations of the process. In this figure, it is shown how students were able to identify and apply the Command pattern (Gamma et al., 1995).

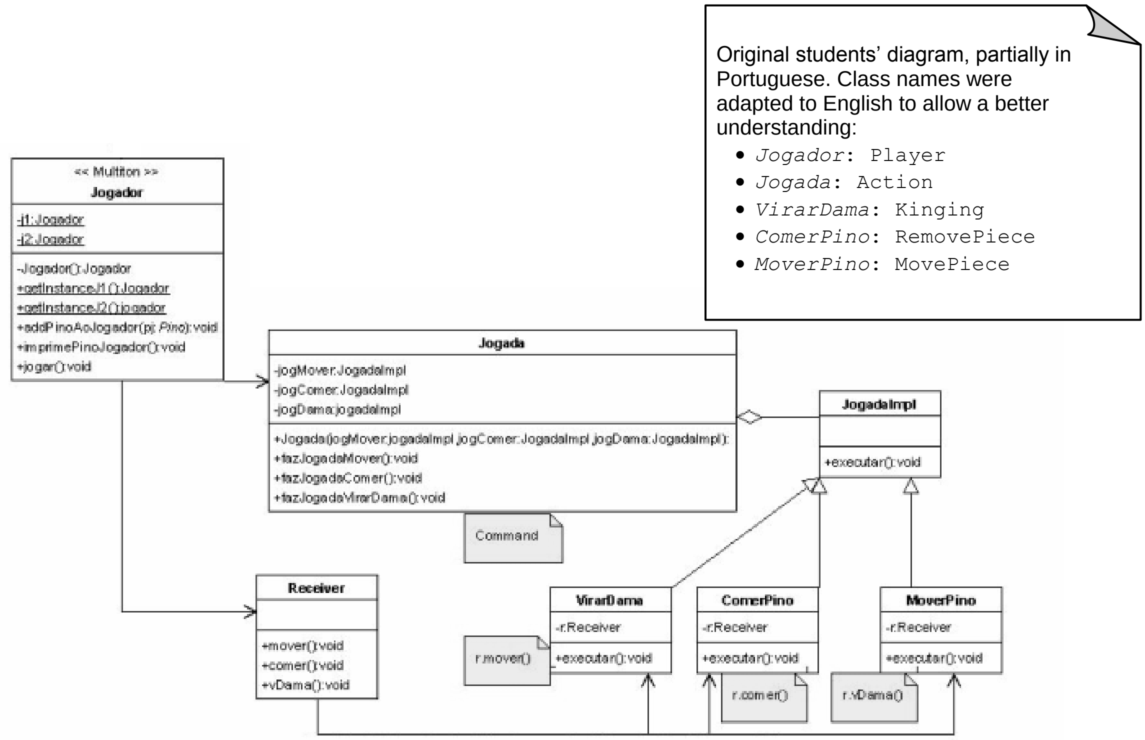


Figure 3 – Discovering the Command pattern in Checkers game

In the second case study, students were meant to face the problems related to Computer Graphics issues when creating a game scenario. Such modeling were supposed to evolve whilst students were presented to new Computer Graphics techniques. Games were used in this case study in order to improve students' sense of motivation, as long they were learning and applying sophisticated techniques to create a game scenario. Simple board games were chosen, since they usually relies on a family of accomplishable challenges for undergraduate IT students, regarding to their simple, but comprehensive, modeling process of pieces and the board itself, as well as they open a wide range of possibilities for studying rendering techniques.

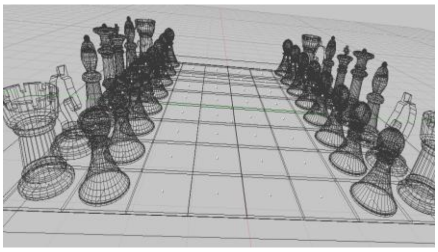
Teaching Computer Graphics to IT undergraduate course implies in analysis and challenges to institutes an approach to teach sophisticated algorithmic content to students of course more driven to application of technology and to find some elements to motivate than in this educational path.

This issue has been discussed since 1970's, and a large range of APIs and frameworks appeared in order to make Computer Graphics learning easier in different contexts (Knowlton, 1972; Towle and DeFanti, 1978). Recently works proposes alternatives to the omnipresent OpenGL through the use of widely spread languages that give support to 2D-3D basic modeling tools, like pure Java (Mukundan, 1999), Java plus Java3D API (Zhang and Liang, 2005; Tori et al., 2006) or even not so popular languages like Ada (Brown, 2004). Other experiences were made with modeling software, like Maya and Blender (van Gumster, 2003; Zhu and Owen, 2004). The present case study was carried out using Blender.

More than just using tools, the experience conduced with these classes was meant to stimulate students to use such tools. It is well-known that complex modeling tools have inherently also complex, user-unfriendly interfaces, besides being, most of the times, machine-consuming, proprietary and expensive. The solution found to some of these problems was using Blender, an open source, cross platform suite of tools for 3D modeling and rendering.

To develop this, it was proposed to Groups of a 4^{th} year IT course one same experiment: modeling and rendering of a board game. Initially, the groups had chosen which game wanted to work and, using concepts of modeling and rendering, had produced a final project of a game in two phases.

Some results obtained by students in this phase are shown in Figure 4:



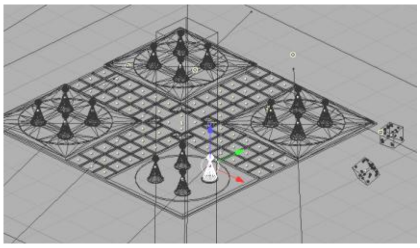


Figure 4 – Some results obtained in modeling phase

The results gained in variety of gotten proposals were sufficiently satisfactory and the motivation with the use of a game as subject was well accepted by IT students. With these results second phase focus is to add realistic effect to the models generated in the previous phase.

The challenge was to add realistic effect (illumination effects) to the models generated in the previous phase. Our current program of Computer Graphics includes two basic illumination models: Lambert and Phong. Although they represent the most basic models of illumination in Computer Graphics, they demand certain mathematical maturity to understand them.

Illumination Models, in Computer Graphics, allow the assignment of shades to points on a 3D model. They represent a simplification of the more general rendering equation, whose computation is a hard computational task. The discrete form of such equation can be seen in Figure 5.

$$I_p = k_a i_a + \sum_{\text{lights}} (k_d (L \cdot N) i_d + k_s (R \cdot V)^{\alpha} i_s).$$

Figure 5 – The general rendering equation

In this equation, the terms (k_a,k_d,k_s,α) represent the material coefficients to be bound to the 3D model and quantifies the amount of reflected light i_a, i_d, i_s (ambient, diffuse, specular).

Lambert illumination model is the most simplified model for illumination, including only the diffuse component. On the other hand, Phong is composed by three components: ambient, diffuse and specular. As the model of illumination of Phong is empirical, we also follow the same empirical strategy to teach the model to our students, whose outcomes can be seen in Figure 6.

Figure 6 – Some results obtained in rendering phase

As an initial strategy, groups had been motivated to experiment several lighting conditions in order to gain mathematical and perceptual skills related to photo-realistic parameters. Using these skills, they had tried some models to get realistic qualities for wood, plastic, amongst other materials that could appear in the scenes of the games. The results had been very interesting, mainly because the students had faced the mathematical barriers of the model and had been able to transform them into sufficiently convincing products of the point of view of realism.

4. Final considerations and further work

This paper showed two case studies carried by the authors in a Brazilian University, where games were effectively used to teach two different subjects: Design Patterns for a Computer Science course and Computer Graphics for an Information Technology course. These case studies, from educational point of view, also demonstrate the plan elements of instructional design used to elaborate a proposal of applying games exclusively in undergraduate courses as Computer Science and Information Technology.

These elements increase the proposal because students were challenged to study in a detailed way Design Patterns and Computer Graphics contents' to have opportunity to create their games architectures and interact with other colleagues to find solutions to real-world projects, not just in a hypothetic or theoretical manner, as usual.

The use of computer games as motivation factor for the presented concepts was extremely important not only for the ludic aspect, as well as for the wealth of exploration elements. In both cases, the exploration factor represents an important aspect for attainment of elegant, clear and cheap solutions for software, as well as three-dimensional realism in models.

One element that can be explored in future researches is to explore new possibilities to amplify the collaborative approach. To achieve this, one proposal could involve the use of massive multiplayer games to investigate how students work and structure autopoietic systems that allow the problem examination and solving in game universe.

Finally, it is relevant to point out that this kind of proposal requires effective changes of teachers' posture. These involve increasing the value of the ludic aspect in education and verify, by use of instructional design analysis, how different contents could be combined in new curricular structures (by inter and transdisciplinary way) to create game universes able to contextualize and offer meaningful learning experiences to students.

References

Alur, D., Crupi, J. & Malks, D. (2003) *Core J2EE Patterns: Best Practices and Design Strategies*.2nd ed. New Jersey: Prentice Hall.

Astrachan, O., Mitchener, G., Berry, G. & Cox, L. (1998). Design patterns: an essential component of CS curricula. In: *Proceedings of the 29th SIGCSE technical symposium on Computer science education* (pp.153-160). Atlanta, Georgia, USA, February.

Ausubel, D. P. (1962). A subsumption theory of meaningful verbal learning and retention. In *Journal of General Psychology*, 66, 213-224.

Barr, R. D. & Tagg, J. (1995). *From teaching to learning - a new paradigm for undergraduate education*. Change, Nov/Dec:13-25.

Barrows, H. S. & Tamblyn, R. M. (1980). *Problem-based Learning.* New York: Springer.

Brown, C. W. (2004). Teaching graphics using Ada. In: *Proceedings of the 2004 annual ACM SIG Ada international conference on Ada*, v. XXIV, 4. ACM Press.

Chidamber, S. R. & Kemerer, C. F. (1994). A Metrics Suite for Object Oriented Design, *IEEE Transactions on Software Engineering*, v.20 n.6, 476-493, June.

Gamma, E. Helm, R. Johnson, R. & Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software.* Reading, Massachusetts: Addison-Wesley.

Juul, J. (2005). *Half-real: Video Games between Real Rules and Fictional Worlds.* Massachusetts: MIT Press.

Knowlton, K. (1972). A report on the use of FORTRAN-coded EXPLOR for the teaching of computer graphics and computer art. In: *Proceedings of the symposium on Two-dimensional man-machine communication* (pp. 103-112). Los Alamos, New Mexico: ACM Press.

Larman, C. (2004). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*. New York: Prentice-Hall.

Menezes, P. B., Weber, R. F., Haeusler, E. H., Melo, A. C. V. e Camargo, M. S. "Proposta de Plano Pedagógico para Cursos de Ciência da Computação". In: Anais do Curso de Qualidade da SBC, 2001. Disponível em <http://www.sbc.org.br/>. Accesso em: 27/03/2006.

Mukundan, R. (1999). Teaching computer graphics using Java. In: *ACM SIGCSE Bulletin , Working group reports from ITiCSE on Innovation and technology in computer science education ITiCSE-WGR '99*, v, 31, 4. ACM Press.

Mustaro, P. N., Silveira, I. F., Omar, N. & Stump, S. M. D. (2007). Structure of storyboard for interactive learning objects development. In: A. Koohang & K. Harman (Eds.), *Learning objects and instructional design* (pp. 253-280). Santa Rosa, California, USA: Informing Science Press.

Norman, G. R. & Schmidt, H. G. (2000). Effectiveness of problem-based learning curricula: Theory, practice and paper darts. *Medical Education*, 34, 721-728.

Prensky, M. (2007). *Digital Game-Based Learning*. Minnesota: Paragon House.

Shaffer, D. W. (2007). *How computer games help children learn.* New York: Palgrave MacMillan.

Tapscott, D. (1998). *Growing up digital: the rise of net generation.* New York: McGraw Hill.

Tori, R., Bernardes, J. L., Nakamura, R. (2006). Teaching introductory computer graphics using java 3D, games and customized software: a Brazilian experience. In: *ACM SIGGRAPH 2006 Educators program*. ACM Press.

Towle, T. and DeFanti, T. (1978). GAIN: An interactive program for teaching interactive computer graphics programming. In: *Proceedings of the 5th annual conference on Computer graphics and interactive techniques SIGGRAPH '78*, v. 12, 3. ACM Press.

van Gumster, J. (2003). Blender as an educational tool. In: *ACM SIGGRAPH 2003 Educators Program*. ACM Press.

Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes.* Cambridge, MA: Harvard University Press.

Wick, M. R. (2005). Teaching Design Patterns in CS1: a Closed Laboratory Sequence based on the Game of Life. In *Proceedings of the 36th SIGCSE technical symposium on Computer science education* (pp. 487-491). St. Louis, Missouri, EUA, february.

Wilkerson, L. & Gijselaers, W. H. eds. (1996). Bringing Problem-based Learning to Higher Education. *New Directions for Teaching and Learning* 68. Jossey-Bass: San Francisco.

Zhang, H. and Liand, Y. D. (2005). Undergraduate computer graphics using Java 3D. In: *Proceedings of the 43rd annual Southeast regional conference - Volume 1.* ACM Press.

Zhu, Y and Owen, G. S. (2004). Teaching strategies: Integrating modeling and animation tools into an introductory computer science graphics course. In: *ACM SIGGRAPH 2004 Educators program*. ACM Press.