Innovate in Your Program Computer Class: An approach based on a serious game

Martinha Piteira
IPS-ESTSetúbal
Setúbal
Portugal
mpiteira@gmail.com

Samir R. Haddad
SENAC Minas Gerais - Faculdade
Belo Horizonte - Minas Gerais
Brazil
samir@pbh.gov.br

ABSTRACT

Computers programming is one of the computer science disciplines which normally present high rates of student's dropout and retention. Literature points to several factors which contribute to dropouts and retentions. The difficulty in understanding abstract concepts and the current teaching method based on traditional lectures with low interaction between students generate low motivation and consequently the lack of interest in learning computer programming.

Several approaches to overcome this problem have been used. Some of these approaches are based on the use and integration of technology in learning. The current generation has a natural acceptance for technology and the use of emerging technologies in education contributes significantly to an increased motivation in students.

In order to help the increasing of motivation and the interest in learning programming in this paper we present a Web-based educational game. Through the use of this game, student learns, applying his knowledge while having fun playing. The game will be integrated and aligned with content and activities of the discipline. To facilitate this alignment and integration it was developed a methodology that will be described. It will also be described the evaluation methodology of the game.

General Terms

Design, Experimentation, Human Factors, Languages.

Keywords

Computer programming, eGames, Online Educational Games.

1. INTRODUCTION

The interest in educational games is increasing in many areas of education. This type of learning tools has been prompted by several factors one of them as referred in literature, is the increasing student's lack of motivation related to current teaching.

OSDOC'11, July 11, 2011, Lisbon, Portugal.

In scientific literature there are several arguments which advocate the integration of games in the teaching process and how they can positively contribute to successful learning.

For instance, Valiathan & Anand [1] argues that games have two necessary elements to the learning process: understanding and motivation. Authors argue that using games on teaching process helps learners to practice essential formulas, facts and processes, motivating learning of less interesting topics by adding a fun element on the learning process, which contributes largely to be considered excellent tools.

Hirumi [2] in its paper argues that games are effective tools for learning because: (a)Use action instead of explanation; (b)Create personal motivation and satisfaction; (c)Accommodate multiple learning styles and skills; (d)Reinforce mastery skills and (e)provide interactive and decision making context. Clark [3] refers in his paper that we can have the best of both worlds: Games and Learning. The two are not mutually exclusive. He said that with right design and tools, they can be mutually supportive. He identified ten things that games have to offer to learners: Motivation, Learner-Centricity, Personalization, Incremental Learning, Contextualization, Rich Media Mix, Safe Failure, Immediate Feedback, Lots of Practice and Reinforcement and Lots of Collaboration.

Games can be powerful tools, provide great fun while motivate, facilitate learning and increase the capacity of retention from what is taught, exercising functions of mental and intellectual player. They have attributes that enhance learning: rules (provide structure), goals (produces motivation), interactivity (allow action), results and feedback (encourage learning). Games force students to decide, choose and prioritize, and bring all intellectual benefits which result from this fundamental virtue, learning to think and then decide.

Based on benefits referred in scientific literature, games have been gradually integrated in learning process. But there are many unresolved issues. For example, which is the correct way to integrate these tools in learning activities? Sung [4] presents in his work some approaches for the integration in computer science courses in a specific area: computer programming.

On the first approach named game development class, the objective is to design and implement new games as a final product. Students learn and use concepts of quality, visual, audio effects, physical simulations and performance. All the concepts are important in a real final product.

The second approach designed game programming classes, is based on the study of some aspects related with design and implementation of games, but the objective isn't the realization of a final product. Some concepts that students have to learn are: event loops, path planning algorithms e terrain representation.

The last approach, game development client, is a final product game integrated in learning process used as a mechanism to facilitate abstract learning concepts or to illustrate a specific concept.

Although there is recognition in scientific literature on the benefits of using games in education, existing also references to some strategies, yet guidelines for its use in practice remain difficult to find.

For example, some questions that remains: What type of games should be used, to teaching and learning computer programming? What are the problems for students in using and learning through games? What are the problems for teachers in using and teaching through games? And can these problems be solved? Can games improve students' performance? Can games disturb learning?

Based on the previous questions we conducted a study composed by several phases. One of them is a 3D Based Web Game implementation, to teaching and learning computer programming concepts. This game will be integrated in computer programming disciplines.

This paper is organized as follows: first is presented a review literature about the benefits of games use in learning process; second is presented the related work; then we present The Cube Game framework. Finally, the paper presents the methodology approach to integrate game in learning process and the evaluation methodology approach, concluding with future work.

2. RELATED WORK

Acquire skills in computer programming is fundamental in computer science courses as in other technological courses. These skills are necessary for performance in professional life but also as a mean to develop capacity for problem solving and abstraction.

However student's motivation to learn computer programming remains very low, with high drop rates and retention. This is a common problem between national and international institutions [4, 5].

In scientific literature some reasons are pointed to this problematic. For instance, one of the reasons is student's abstract concepts understanding difficulty, basic concepts understanding difficulty, like data types, among others. These concepts don't have an easy perception because there aren't analogies with the real world. Another reason mentioned is the actual courses structure and the teaching traditional methods. These methods are based in expositive classes and with a poor interaction between students. The traditional methods used in classes, contribute to discourage students [6]. Students should be involved in relevant activities to their learning and understand the added value by carrying out these activities.

Several approaches are developed, used and tested to give an answer to this problem. In coming years the lack of qualified professional in computer programming may become a serious problem.

Many approaches are related with computers programming didactic and with various teaching/learning instruments use.

Another approach is tools integration based on technology to support teaching and learning process. The aim of this approach is increase the student's motivation and bring them to the learning center. Tools as visual programming and tools with similarities with games are used in this approach [7, 8].

In scientific literature there are several references to games and animation-based environments used in class, as a way to motivate and help in better matter understanding.

One of this tools is Jeliot3, a visual programming tool based on Java which allows visualization of how a program is interpreted. The target tool are novice programming users. The main objectives is to facilitate procedural and oriented-objects programming learning. The main feature is the automatic or semi-automatic data and control flow visualization and its a single-application. The programming concepts used are: basic oriented-objects, objects and inheritance visualization [9].

The application provides an interface based on one window, divided by areas. The learning concepts like method calls, variables and operation are displayed on a screen as the animation goes on, allowing student to follow step by step the execution of a program. Programs can be created from scratch or they can be modified from previously stored code examples. The Java program being animated doesn't need any kind of additional call. All the visualization is automatically generated.

Table 1 – Jelito3

Tool Type	Language	Computer Programming Concepts	Game Characteristics	Target
Program Visualization Application	-Syntax: Java	- object-oriented concepts - visualizing objects and inheritance	Without any Characteristics of Games	Novice computer programming users

Another tool referred in literature is Jeroo, which was developed to help the novice programming computer users. Through this tool use it's possible to learn control structure semantics, the basic notions to use objects to solve problems and learn to write methods that define the behaviour of objects. Jeroo was inspired in another tool named Karel [10].

The authors create a metaphor that is neither technical nor violent.

The metaphor is "*Santong island is an uninhabited speck of land in the South Pacific Ocean. In the late 1980's, naturalist Jessica Tong discovered two previously unknown species that are unique to this island. One is the Jeroo, a rare mammal similar to the*

wallabies of Australia. The other is the large Winsum flower that serves as the Jeroo's primary food source. The Jeroos hop about the island picking and planting flowers. As they move, they must avoid the water, and evade or disable nets that have been set by a zoo's species collector". Source: http://www.jeroo.org

Students can configure the island and develop programs to control the actions of up to four Jeroos and their interactions with their immediate surroundings.

Jeroo tool has four components: user interface, Jeroo programming language, editors and runtime module and have the syntax close to Java and C++. There aren't data types and variables other than references to Jeroo objects. However the tool has eight predefined directional constants, subdivided by: relative directions and compass directions. Relative directions are, for instance, Left, Right, Ahead and Here. The Jeroo object has attributes like, location, direction and number of flowers. It is possible to view the code created in small transitions to Java, C++, C# and Python.

Table 2 – Jeroo Tool

Tool Type	Language	Computer Programming Concepts	Game Characteristics	Target
Program Visualization Application	- Syntax close to Java and C++	- Control Structures, methods, objects	Challenges: without score	Novice computer programming users

RAPTOR is an algorithmic visual programming tool [11]. RAPTOR allows students to create algorithms by combining graphic symbols. Students can run algorithms step by step or in continuous mode. Environment allows visualization of symbols location that are being carried out and contents of all variables. The tool was developed with ADA, C# e C++ and its run in .NET

The graphical environment starts with a begin and an end symbols in an empty window. Student can then add symbols corresponding to loops, selections, procedure calls, inputs and outputs from palette selection and insert them in a particular program point.

Table 2 – Raptor tool

Tool Type	Language	Computer Programming Concepts	Game Characteristics	Target
A Visual Programming Environment	Based on symbols	Teaching Algorithmic Problem Solving	Without any Characteristics of Games	Novice computer programming users

Another tool named Alice, "*is a 3D programming environment that makes it easy to create an animation for telling a story, playing an interactive game, or a video to share on the web. Alice is a freely available teaching tool designed to be a student's first exposure to object-oriented programming. It allows students to learn fundamental programming concepts in the context of creating animated movies and simple video games. In Alice, 3-D objects (e.g., people, animals, and vehicles) populate a virtual world and students create a program to animate the objects.*

In Alice's interactive interface, students drag and drop graphic tiles to create a program, where the instructions correspond to standard statements in a production oriented programming language, such as Java, C++, and C#. Alice allows students to immediately see how their animation programs run, enabling them to easily understand the relationship between the programming statements and the behaviour of objects in their animation. By manipulating the objects in their virtual world, students gain experience with all the programming constructs typically taught in an introductory programming course." Source: http://www.alice.org

Table 3 – Alice tool

Tool Type	Language	Computer Programming Concepts	Game Characteristics	Target
Visual Programming 3D Environment	Java, C++ and C#	- object-oriented concepts	Without any Characteristics of Games	Novice computer programming users

Table 4 – Karel the Robot tool

Tool Type	Language	Computer Programming Concepts	Game Characteristics	Target
Web Game	Visual Programming based on commands	Visual commands to execute instructions: moveon() turnleft() and others. Conditionals and Recursion implementation with the visual commands	Challenge: with level score and publication to others community members (optional)	Novice computer programming users and Expert computer programming users

Another tool named Karel has been develop and adapted, and some versions referred in literature are: Karel++, Karel J.Robot, JKarelRobot and Waterloo.

Karel the Robot is a Robot living in a sample world that can perform tasks and Karel World is a grid-work of streets and avenues that robot transverses, containing special objects that Karel can sense. These tool has six primitive instructions: turnOn(); move(); turnleft(); pickbeeper(); putbeeper(); turnoff();

The last tool is the light-Bot. This tool is a based web-browser and free to use. The tool is available through this web link: http://armorgames.com/play/6061/light-bot-20. This tool is based

on game-puzzle and uses programmer-style logic. The objective is to "program" the robot to do tasks that are line up tiles on the grid at the right and trying to light up all the blue squares. Light-bot version2 has available several levels of difficulty: basic, recursion, conditional and expert. In this version, the creators made available the new functionality: the level editor. It is possible to create new levels and share them with the user's community.

Table 5 – Light-Bot Tool

Tool Type	Language	Computer Programming Concepts	Game Characteristics	Target
A Visual Programming Environment	The original based on Pascal. Versions based on original, with several computer programming languages (e.g. C/C++)	- object oriented concepts	Challenge: without score	Novice computer programming users

Final conclusions based on related work review:

- All tools have the novice computer programming users as target;
- The programming paradigm more used in these tools is oriented objects programming;
- However, the computer programming concepts presented in these tools, are common to two programming paradigms: procedimental and oriented-object;
- Most tools are single-applications. Interaction and information sharing between users isn't possible;
- Most tools don't have game characteristics and any additional elements to promote motivation, as well as, mechanisms to reward student's performance.

3. THE CUBE GAME – Conceptual Proposal

Based in literature review and related work, it was developed a game to teaching and learning computers programming. The initial requirements were:

- A tool for learning computer programming concepts;
- Target: novice programming computer users;
- Game characteristics;
- Game type: non-violent and non-technical difficulty
- Web Based Tool / Game.

We choose a web based tool, because we want to allow distance learning. Nowadays the distance learning is a fact. But we also want to register and analyse the user interaction and web architecture with database support, being this the best choose.

Defined the architecture, the following aspects were identified:

- Learning objectives and computer programming concepts;
  - The following computer programming concepts were identified: constant; variable; counter; arithmetical operator; logical operator; boolean condition; code block sequential execution; repetition structures; if-else structures and functions.
- Game characteristics;
  - The following games characteristics were identified: Challenge, Collaboration and Personalization.

The Game platform is composed by several modules, being the principals: Game Engine, Level Editor, Users Management, Historic Management, User Profile, Message and Avatar Edition.

The management user's module will allow the user identification in platform game. The system saves the name, number and class. The system associates interactions with the user, save all these interactions and save the last attempt to pass a level. Level editor module allows teacher to create, edit and eliminate the game levels. Through this module it will be possible to adjust the level to the student's performance. Historic module allows viewing all the users' interactions and game module is where the game rules were defined.

Now we present a brief tool description.

We described some game rules and game mechanics. Type game is a "puzzle" game. The environment is composed by cubes. The minor or bigger difficulty level game is based in the number of cubes disposed in the environment. An avatar must be moved from a start point to an end point.

To achieve this objective, several actions must be performed. The principal actions are: forward, turn right, turn left, jump down, jump up, and back forward. All these actions are defined in interface through graphical icons, with code correspondence. This code stays visible in a small window in the interface.

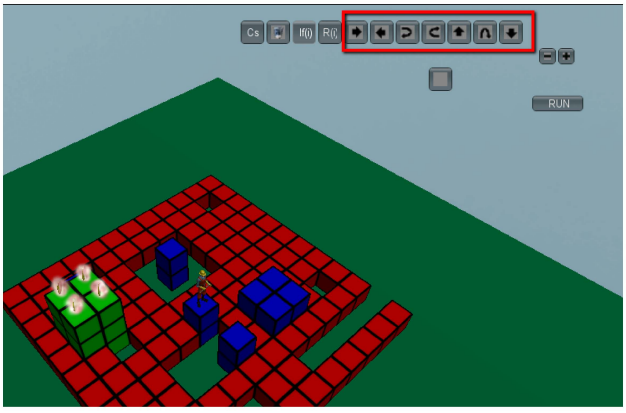

Figure 1- Cube Game (1)

The students' motivation is one of the elements we must keep in mind. To promote student's motivation in game the challenge

characteristic was implemented, and this challenge results in achieving a best level resolution to acquire medals. Students can achieve gold, silver and plate medals. Based on the main idea of challenge all students will see the best results of their colleagues.

We intend to prevent the students demotivation. So, even if a student cannot reach the medals, they can pass the level. We defined a score "passed", that allows student to pass the level with a few instructions.

The game modes defined were: the avatar makes routes from a starting point to an end point with a minor instructions number; collect coins; collect cubes to reach the end point; and collect hidden treasures.

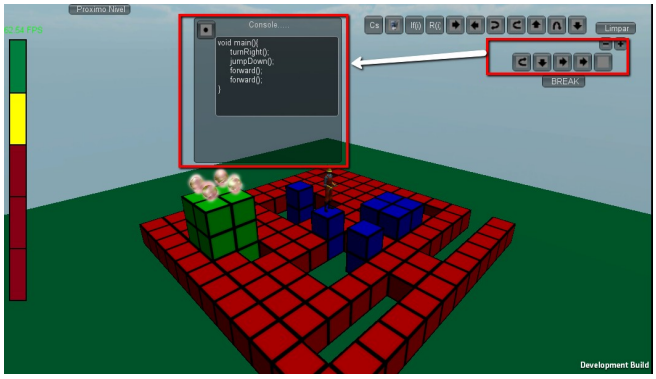

Figure 2 - Cube Game (2)

The game levels have learning objectives associated. These objectives are based in computer programming concepts before mentioned. However, in initial levels players only need to play the game without concerns about code. We want to prevent initial negative reaction to computer programming from the student.

3.1 Technologies

Most technologies chosen were open source technologies and are listed in the below table.

Table 6 – Game Technologies

Web Technologies	DataBase	Graphical Game Environment
• HTML • JavaScript • PHP • XML	• SQL • MYSQL	• Unity 3D (free version)

4. FUTURE WORK

Now we describe the future work. These works consist in three principal tasks: Methodology definition for integrated game in computer programming disciplines; evaluation methodology; results analyse and discussion.

In this paper we describe briefly the first two tasks.

The Cube Game will be used in introductory computer programming courses. We use Cube Game for the first four, five weeks of a 15 week semester. The computer programming concepts available in game are aligned with the concepts learned in introductory computer programming discipline.

Students can learn the concepts computer programming in traditional class, through the expositive method, tutorials and others. However, student can also learn through the game in the correspondent level.

The learning activities and the exercises (levels) in game will be conditioned. Our intention is to student achieve learning objectives only on the correspondent week. Teacher hides or shows the levels with the activities alignment provided in the initial discipline planning.

One of the objectives' is the alignment between the game activities and discipline planning. We present a simple example from what we can do to achieve this goal.

For instance, in a week x the learning objective is to apply repetition structures. This content will be exposed in the classroom and the extra activities can be exercises alternatively realized. (1) Student can realize exercises without game support or (2) realize them in game. All exercises are similar.

In first approach the teacher makes available the exercise, for example, in Moodle platform, and in second approach the student plays the game levels correspondent to week content.

The student will be evaluated and the evaluation results will be reflected in final grade. The assessments criteria must be equal in two exercise modes.

With these approaches we want to give freedom to the students. They choose how they learn and train the concepts. Then will be possible to analyse what are the preferred learn method choose by students.

For the evaluation tool we think it is important to focus on three dimensions: programming concepts, student's satisfaction and teacher's satisfaction.

In programming concepts performance dimension is one of the variables we want to analyse, for example how game use can affect students' performance? To obtain these results we will consider two student groups: students who played the game and those who did not play. The research instrument will be the statistical analysis from the students' assessment results.

To analyse the student's satisfaction, we will use the questionnaire instrument research, based in likert scale and we intend to obtain the satisfaction level felt by student in game using and how this satisfaction level is related with the student's performance.

Finally, to analyse the teacher's satisfaction, we will use the questionnaire as an instrument research. We want to analyse aspects related with the teacher's acceptance of tools based on technology and the perceived benefits to education.

5. REFERENCES

[1] Valiathan, P., Anand, P., "Designing Games for e-Learning: A Frameword", ASTD, Available from: http://www.astd.org/lc/2008/1008_purnima.html [viewed January 2010]

[2] Hirumi, A., Kebritchi, M., "Examining the Pedagogical Foundations of Modern Educational Computer Games", doi:10.1016/j.compedu.2008.05.004. Available from: http://www.elsevier.com/locate/compedu [viewed 5 May 2010]

[3] Clark, D., "Games and eLearning", Caspian Learning, Available from:http://www.caspianlearning.co.uk [viewed 01 November 2009]

[4] Sung, K. "Computer Games and Tradicional CS Courses". (December, 2009). Communications of the ACM. Vol.52. N.12. DOI:10.1145/1610252.1610273

[5] Robins, A., Rountree, J., Rountree, N., "Learning and Teaching Programming: A Review and Discussion", Computer Science Education, 2003, Vol.13, No.2, pp.137-172. Available from:

http://www.cs.otago.ac.nz/staffpriv/anthony/publications/pdfs/RobinsRountreeRountree.pdf [viewed April 2011]

[6] Miliszewska, I., Tan, G. Befriending Computer Programming: "A Proposed Approach to Teaching Introductory Programming". (2007).Volume 4. Available from:

http://proceedings.informingscience.org/InSITE2007/IISITv4p277-289Mili310.pdf

[7] Felden, M., Clúa, O., "Games As A Motivation For Freshman Students To Learn Programming", 34th ASEE/IEEE Frontiers in Education Conference, Savannah,GA, October 2004.

[8] Whiton, N.,"Motivation and Computer Game Based Learning", Proceedings ascilite, Singapura, 2007. Available from http:\\www.ascilite.org.au/conferences/singapore07/procs/whitton.pdf [viewed 23 February 2011]

[9] Bem-Ari, M. , Moreno, A., Myller, N. , Sutinen, E., "Visualizing Programs With Jeliot3", AVI'04, May 25-28, 2004, Gallipoli(LE), Italy.

[10] Sanders, D., Dorn, B., "Jeroo: A Tool for Introducing Object-Oriented Programming", SIGCSE'03, February 19-23, 2003, Reno, Nevada, USA.

[11] Carlisle, M, Wilson, T., Humphries, J., Hadfield, S., "RAPTOR: A Visual Programming Enviroment for Teaching Algorithmic Problem Solving", SIGCSE'05, February 23-27, 2005, St. Louis , Missouri, USA.