

Entertaining Education – Using Games-based and Service-oriented Learning to Improve STEM Education

Jon Preston¹, Briana Morrison¹,

¹ Southern Polytechnic State University, 1100 S Marietta Parkway
Marietta, GA 30060 USA
{jpreston, bmorriso}@spsu.edu

Abstract. This paper addresses the development of a computer game design and development curriculum at the authors' institution. The basis for curriculum decisions, as well as comparison to the other institutions' curricula is covered. In situating the curriculum within the current degree programs, games-based versions of existing courses are also being offered. The experience of the authors with the initial offering of a games-based introductory programming course is also explained, along with the initial assessment of results from the experience. Our experience of using games-based learning in an introductory laboratory is presented. Finally, we demonstrate how games-based learning can be extended beyond the classroom as we work to promote science, technology, engineering, and mathematics (STEM) with local elementary schools; our current project develops an ocean ecosystem exploration game that teaches oceanography and ecological sustainability.

Keywords: curriculum, games-based, learning, programming, motivation, STEM, K12, development, sustainability

1 Introduction

Gaming education has become a hot topic for students and educators. Students are drawn to digital gaming for pleasure and this extends to career choices. Educators view digital gaming as an avenue to increase enrollment in the computing discipline and as a motivation tool.

This paper presents our encompassing vision of incorporating gaming at our institution. This includes defining a new game design and development curriculum, incorporating game-based learning into existing courses, and extending games-based learning beyond the classroom with the effect of promoting science, technology, engineering, and mathematics (STEM) education within our local community. We believe that games-based learning offers an opportunity to motivate students in existing disciplines as well as expand the educational opportunities within universities.

1.1 Game Development Curriculum - Motivation and Development

We became interested in incorporating gaming at our institution for many reasons. There is a wide, fascinating, and growing field of research in the use of computer games in education [9]. Video games are a significant cultural influence [9] and are useful in improving design strategies [5][7], enhancing and motivating learning [4][12], and increasing the success of job skills training [6]. While traditional educational environments and institutions may fail to properly motivate and situate learning, utilizing best practices of psychology and motivation theory, games-centric learning environments offer potential to meet the needs of competence, autonomy, and relatedness to enable students' success [4]. Research also shows that video game players achieve better scores in critical thinking, strategy development, and problem solving tests than non-players [9].

As we explored adding gaming, we were guided by the motivation theory associated with games in education and how this influences learning. While many debate the meaning of the idea of "fun," we are inspired by Malone's assertion that challenge, fantasy, and curiosity are the keys to making a game fun [9]; further, we see a significant corollary between these ideas of what makes a game fun and what makes a motivating, empowering, and successful games-based curriculum of study.

Challenge: like a game, a program of study must present students with clear (and multi-level) goals and provide feedback as to the progress toward achieving the goals.

Fantasy: student achievement and learning can be motivated by intrinsic and extrinsic settings ranging from mastering new topics/content and achieving good grades, and as Malone points out, these fantasies/settings vary by student.

Curiosity: providing learning environments of an "optimal level of complexity" can inspire students to expand their view of the subject matter such that their "existing knowledge seems incomplete, inconsistent, or unparsimonious" and deep learning can occur.

If, as Malone puts it, "computer programming itself is one of the best computer games of all," [9] then why not adopt computer game design techniques to the process of learning in a game design and development curriculum and within programming courses? This would not only create a new program of study to meet growing demand but also serve our current students to improve retention, motivation, and learning. These ideas led to our overall vision, which has three components. First, we have developed a computer game design and development curriculum based in part on these three keys. We have infused new games-based courses into our existing computing programs, and we have expanded the learning to outside the classroom with service opportunities for games-based learning.

1.2 Games-based Laboratory Learning

In an effort to improve performance, retention, and motivation in our computing programs (computer science, information technology, and software engineering) and better serve our non-majors, our school began offering a games-based flavour of our

introduction to programming course in the Fall of 2008. This has been extended in Spring of 2009 into two courses (introductory programming and intermediate programming). To compliment the lecture content of the course, we designed a series of games-based labs that bolstered the typical content of an introductory course. Our initial foray into games-based programming and the methodology used to assess our efforts may be valuable for others. Our empirical results of our study on student motivation and learning success indicate that students find games-based learning motivating, though there are some interesting challenges to overcome in their perceptions of their ability to create “real” programs.

We describe the physical space created with computer and console setups to support our lab and how this space is used beyond the gaming courses. Further, we present how the concepts of the course were mapped to games-based labs to support core course concepts while all the time using motivating games-based learning. The students’ response to the labs and the projects that they were able to develop are positive and in general, while they often felt overwhelmed with the magnitude of the complexity of developing games, they were successful in their development and were ultimately quite satisfied and took ownership of their creations.

1.3 Grassroots, Service-based Game Development

Game studies are of growing importance and the focus of ever-expanding undergraduate programs throughout the United States and beyond [10]. In addition to programs that study game development, games and simulations have been utilized to motivate learning and allow students to explore educational content; such a constructivist approach to learning has shown promise [1][11], but often educational games (“edutainment”) lack the polish and in fact sometimes have a detrimental effect as students are turned off from “lame games” [4]. We have developed a sustainable and low-cost approach to developing compelling, fun, and educational games by tapping into the creative energies of our university students; this group works with a local elementary school and a local non-profit group to build these games and enhance the learning of our university students and the students playing our games. This service-centered approach to game design and development shows much promise, and as an added benefit, the group promotes science, technology, engineering, and mathematics (STEM) education.

We created a game play and development group of students and faculty to create community within our program and extended our game lab and development group to serve our local community. We present details of how our play/design group was able to provide valuable service to a local elementary school as we designed a learning game. Finally, we offer conclusions and discuss our future goals and activities for using games- and service-based learning to promote the education of our students as well as elementary students within our community.

2. Curriculum Development

Initially, we conducted a study of other universities that offered game related programs and courses; through this study, we sought to identify trends in other programs' curricula and learn from past successes so that we could adopt best practices from the inception of our program. Further, we wanted to ascertain how other programs incorporated the IGDA curriculum guidelines.

While hundreds of programs exist, we chose to locate comparable university programs by analyzing various computing education publication venues spanning the past four years. This identified 44 institutions. Of these, 22 offered full degrees, 16 offered a select few courses as special topics to existing, non-game degrees, and 6 offered certificates or concentrations/specializations in game design and development.

Table 1. Program Course Categorization

Category	Course Type
Game-related	Game design, level design, game business, game analysis, game engine development
Computing	Traditional CS courses such as data structures, operating systems, graphics, HCI, algorithms, software engineering
Arts/Humanities	2D and 3D modeling, art, animation, sound/audio engineering

We categorized the programs' content into three groups: art/humanities, computing (i.e., traditional computer science), and game specific. Each program's upper-division content was examined to determine which course hours fell into each of these three categories. Capstone, studio, and senior project courses were placed within the category that was most closely related to the program's description, but more often than not, these courses were categorized as game-related since the intent of these programs was game focused. Table 1 provides examples of our categorization.

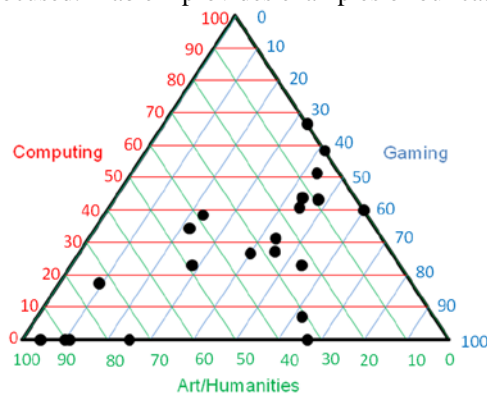


Fig. 1. Full Game Program Emphasis – Games-based programs from around the US offer varying emphasis on game development, traditional computing, and arts/humanities.

Figure 1 shows a ternary (triangle) plot of the 21 full degree programs and their relative (%) emphases on gaming, computing, and arts (one program was excluded

because it was completely customizable and thus didn't lend itself to our analysis). For example, if a point resides in the lower left corner, this program would emphasize arts with little or no computing or gaming; if a point resides in the top corner, this program would emphasize computing with little or no arts or gaming; and if a point resides in the lower right, this program would emphasize gaming with little or no arts or computing.

From this analysis, we observe that program emphases in gaming range from 9% to 67%, emphases in computing range from 0% to 67%, and emphases in arts range from 0% to 96%. Clearly there is a wide range of approaches in offering game-related curricula.

2.1 Supporting the New Degree

What courses are necessary for a computer game design and development program? As our background research into undergraduate games-related programs revealed, there are a myriad of solutions to this question – all of which serve different student constituencies. Within our new program, the students are exposed to the breadth of the field of computer game design and development, including digital media, human-computer interaction, the history and theory of gaming, game design, 2D and 3D graphics, simulation, modeling, software engineering, artificial intelligence, data structures, and algorithms. Current and emerging domains including online games and massively multiplayer games (MMOG), casual games, mobile games, and serious/educational games are also explored.

Mindful that the IGDA Curriculum Framework [3] does not prescribe specific courses but rather emphasizes core topics, we adopt the same approach in allowing flexibility within our program. Our program supports all but one of the IGDA Curriculum Framework's core areas (we are missing Audio Design, though this could be taught as a special topics course in the future). The required courses in the degree ensure students are exposed to the breadth of the field of computer game design and development. Students are also given flexibility to customize their experience and apply the knowledge gained in their required courses within a concentration where they may gain a depth of knowledge within their chosen area.

Finally, in their last phase of the program, students have a year-long studio experience wherein they develop games and systems utilizing the breadth and depth of their knowledge. This two-course sequence provides an opportunity for students to be mentored by faculty and their peers in the first semester and in turn mentor fellow students in the second semester. This studio experience may also be inter-disciplinary incorporating technical communication majors, majors with domain expertise as well as computer science and software engineering majors. The year-long capstone project developed in these courses is a vital component in graduates' portfolios and will be showcased on the program's Web site. Moreover, given the recruitment and hiring practices in the digital entertainment and computer gaming industry portfolios are crucial in helping graduates secure employment.

Figure 2 shows the core courses that constitute our computer game design and development curriculum. In total, 11 new courses were developed for the new program and we were able to make use of eight existing courses.

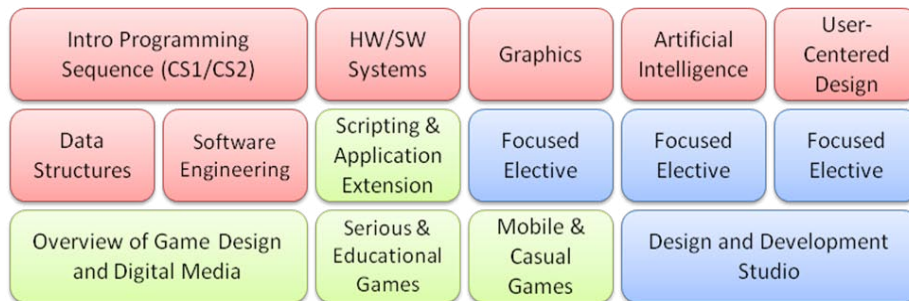


Fig. 2. Course Development – existing courses are shown in red, new courses design specifically for gaming are shown in green, and newly-created games-based elective and capstone courses are shown in blue.

The structure of this degree allows for Malone’s 3 components: the curriculum itself is challenging with feedback during each course, multiple types of “fantasy” can be explored in many different courses including the studio experience, and the concentrations allow for the students to fulfill their own “curiosity”.

2.2 Infusing the Game Development Courses into the Existing Programs

Another motivation in developing the new computer game design and development degree was to provide options for existing computing majors. Our school currently offers three Bachelors of Science degrees in Computer Science, Information Technology, and Software Engineering, a Bachelor of Arts in Computer Science, and three Masters of Science degrees in Computer Science, Information Technology, and Software Engineering. Consequently, the new program needed to fit in well with the existing courses, but we were mindful of the IGDA curriculum suggestion that any new games-based degree needs to go beyond simply adding a few courses to an existing program; our setting would have made such a plan easy given the volume of existing courses, but while the easiest route, this would not have been the best long-term approach for a successful games-based degree.

Core courses such as necessary math, introductory programming, and data structures were available and appropriate to incorporate into the new program. Upper-division courses critical to game development such as Artificial Intelligence, Software Engineering, and Graphics were also currently being taught and were adopted into our new game curriculum, although some additional learning outcomes specific to gaming have been added into the courses. Utilizing these existing courses proved to be vital in the program’s success due to budget limitations, and we were thus able to avoid creating an inordinate number of new courses for the new program; additionally, using these current courses ensures that we are able to offer them frequently as they serve students in as many as four different degree programs. Thus we avoid the problem of courses not being offered enough for students to schedule them and graduate in a reasonable amount of time.

While we will try to dispel any misconceptions during orientation sessions for prospective and incoming students and advising of existing students, it seems likely

that at least some students will eventually transfer out of the game design and development program into one of our other programs in computing. By building a curriculum that has a large number of lower-division courses in common with our other computing programs we afford such students the opportunity to shift majors without losing credits.

A final benefit of this approach is that we were also able to create a Minor in Computer Game Design and Development; this set of courses allows our current students to explore the field of game design and development while retaining their current major. They are exposed to the “overview” courses and then can choose three other upper-division game courses; additionally, students in the minor may elect to participate in the studio experience to add to the inter-disciplinary teams for a rich experience.

3. Games-based Laboratory

Within our current curriculum, our three course programming sequence (introductory, intermediate, and data structures) all have a closed lab component. Students attend lecture for 3 hours each week and are required to attend a 2 hour lab to practice the concepts discussed in class. When we introduced the gaming based flavor of the introductory programming course, we recognized the need for a gaming based lab as well. We thus created our “game hive.”

The gaming lab occupies a dedicated room that consists of 14 Xbox 360s connected to development computers with Visual Studio 2005 and 2008 as well as the XNA Framework 2.0 and 3.0. Each development space can accommodate up to two students working side-by-side, so the space (and many of the labs) support a paired-programming model of XP (extreme programming), though this is not required and students can work alone.

Each development machine is bound to a different Xbox 360 with an XNA Creators Club Subscription. This allows the PCs to easily deploy to the Xbox 360s via Visual Studio and allows the students to play their games on the console. While the games run completely on the PCs, students were visibly and highly motivated when seeing their creations run on the Xbox 360.

Further, we have made many free audio and graphics editing toolkits available to the students. These tools include Audacity, Sony’s free Acid Express, Paint.NET, and the GIMP. Such tools allow students to generate the graphics and audio for their games, and since the tools are free, the costs are minimized for the university and the students (if they choose to install these tools on their personal computers). Similarly, there are free editions of Visual Studio 2005 and 2008 as well as the XNA Framework, so students can replicate the lab set up in their personal computers for free if they desire.

To maximize usage of the space, the other half of the room contains a couch, two 46” wall-mounted plasma televisions and two Wii consoles, one PS3, and another Xbox360 Elite. This “play” area is used by students to play one of the games from our library of over 50 titles. This allows for research of existing games as well as creating community among our large population of commuter students. Additionally,

we utilize our game development lab to support local schools and to promote STEM education. We want to encourage students to see computing and gaming as viable educational pursuits. We have had numerous visits as “field trips” to the university; these visits increase the ties between K-12 education and higher education that we have extended into a collaborative game development project. (See section 4.)

3.1 The Lab Exercises

When adding the games flavored introductory programming course, the closed lab assignments needed to be created. The labs have been designed to allow students to learn the programming concepts while implementing games. Our approach utilizes C# and XNA as the language and API for game development, but our pedagogical approach is language-agnostic and any modern programming language with a sufficient graphics API could be adopted.

Initially the core programming concepts were mapped to the lectures: data types, arithmetic operations, using classes, writing classes, selection structures, repetition structures, and arrays. We created a lab for each concept and added a lab for the introduction to the IDE. Each lab was based on an existing game and clearly identified the learning objectives and grading rubric. Each lab provides background as to the motivation of the lab, and then enumerates the directions that the students are to follow. Typically, students downloaded a “starter project” that contains much of the program already implemented and commented sections indicating the portions that the students had to complete.

3.2 Assessment

In assessing the results of the using the game-based flavor of the introductory programming course, we collected both quantitative and qualitative data. At the conclusion of the course, students were asked to provide anonymous feedback through a survey. The survey was given to students in the games-based section of the course as well as students who were taking the traditional, non-games-focused section of the course. A total of 91 students responded to the survey, yielding a 75.8% response rate. Forty of the gaming students (75%) responded and 51 students (76%) in the traditional offering responded. Students in each of the groups were similar: approximately 73% of each group planned to major in computing, 15% of the gaming students were female (n=6) while 21.6% of the traditional students were female (n=11), and grade expectations for both groups were similar, with the majority of both groups expecting Bs. None of the respondents expected to fail the class, and only 1 respondent from the traditional offering expected a D.

The responses from the gaming section were overwhelming positive. While many felt the course was difficult (as many students feel about their first programming course), they felt they had accomplished something of value. Over half of the students report that they continued to work on their assignments even “after they were working,” which indicates that the students were motivated by the content. Nearly two thirds of the students also reported that they showed their assignments to friends.

During the closed labs, we noticed that many students customized their games and tried out novel graphics and extended the game play.

Quantitative data collected comparing performance between the games and traditional sections will be reported in a future publication.

3.3 Student Reflection on Their Ability

It is particularly interesting to note that our study finds that more students taking a games-based section of the programming course thought that they had less ability than their counterparts in a non-games-based section. For the gaming students, a majority (52.5%) thought they could not yet write an interesting or meaningful program while only 41.18% of the traditional offering students believed this.

We believe that this is because of the nature of the development programs and the complexity of the labs and assignments in the games-based course. The complexity of the programs they were exposed to was considerably higher than those in a typical introductory programming course. The typical introductory programming assignment is approximately 100 lines of code or less. The game programs utilized in our course consisted of 200 or more lines of code, some of them reached into the thousands. Instead of considering a single class, most programs had four or more classes that the students were to be concerned with.

The overhead of the XNA framework forces students to use additional abstraction skills. The XNA framework requires that the game is initialized and assets (audio, sprites, etc.) are loaded, then the main game loop begins - wherein the update and draw methods are repeatedly called. This event- and loop-driven model of execution may be more difficult to grasp than a top-down model of execution (which is what is used in console-based programs that dominated the non-games-based section of the course).

The gaming students' final assignment was team based--to design and implement their own game. By asking the students to create from scratch their own game, the authors believe the students have a clearer picture of all the skills necessary to create a large scale project and that this is reflected in their answers to the survey question. They may not be able to write a program *alone* that is interesting or meaningful, but they can with a team. This was reflected not only in the final projects created (which were complex and required asset creation and game development) but also in the responses to the survey. Most gaming students were most proud of their final project which can be seen in the following sample of responses that capture the tone of the students' feelings about what they were most proud of and like most about the course:

"The final project where we got into groups and created a game. It was meaningful to me because it was something we all came up with and created, not simply a game we added code to so that it would work."

"That I got the tank assignment working without any help. (It seemed -so- much harder than it was.)"

"1301 Gaming was really fun. Learning how a game actually works was probably the best part, and the time and effort it took for a game to be made. Making my own games was fun also."

“CSE 1301 was fun because I was learning new things. It was also fun to make games with other people that became friends.”

“I was really proud of the tank game, because it was the first homework assignment that we had to add code to, to make it work. At first it seemed extremely out of my league, but once I took the time to look at it and tinker with it, I finished it.”

4. Service-oriented Game Development

Beginning in the Fall of 2008, we created a game play and development group at our university. This group was extracurricular and completely voluntary. The group met weekly and alternated between playing games and developing/critiquing games, but soon we realized that having meetings twice a week would allow for more “serious” progress. We wanted to create an atmosphere that allowed students to meet new friends, support their learning via impromptu peer-tutoring, and provide a vehicle by which they could apply their design and development skills in a fun way via game development. This group began with four students and has grown to over a dozen from various majors ranging from computer science to fashion design. We have even had students switch majors into our program as a result of this group.

This cross-discipline game group allows students to incorporate and practice their learning from various computing fields such as usability and HCI, graphics, and algorithms. Non-computing majors are also involved and add perspectives concerning design, playability, fun, and narrative.

4.1 Service to the Community

One of the key elements of our new computer game design and development program is service and partnerships with local businesses and non-profits. In the final year of our program, student teams are to work on long-term (2-semester) projects with real customers such that they are developing games with a purpose (as well as for their portfolios). To this end, we view entrepreneurialism and service as central to our program.

To jump start the service aspect of our vision, we asked our game play and development group of students explore the possibility of creating an educational game for a local elementary school. With the help of a local non-profit group, the university students met with a group of elementary school students to begin exploring a game about oceanography.

4.2 Collaborative Game Design

Similar to [5] and [7], our game design group sought to incorporate the input of the elementary students that would be playing the game. While this could be considered typical “listen to your customers early in the design process,” we consider this step important to this project specifically as we wanted to not only incorporate the elementary school students’ suggestions into the design of the game, but we also

wanted to motivate their STEM learning; it was critical that the elementary students gain insight into the design and development process so that they aren't just learning about oceanography but also learning what goes into the design and development of the simulation.

As Figure 3 shows, the students (both elementary and university) that worked with us on the design really got involved and took ownership of the process. They had many creative ideas and also confirmed some of our original design elements such as submarine customization, achievements and trophies, and mini-games. Some ideas, such as adding a popcorn maker in the ocean research base, were surprises to us, but are easily added to the game and we hope that such Easter eggs will make the students feel they had a strong impact on the development of the game.

The current oceanography game that we developed contains three main components: a submersible builder (fully customizable with lights, textures, measurement instrumentation, etc.), an ocean environment exploration game, and a system to adjust and "fish out" certain species. By utilizing all three of these pieces, the elementary students can learn about design trade-offs, exploration, and ecosystems. Our game components support the learning objectives of the teachers, who were very pleased and supportive.

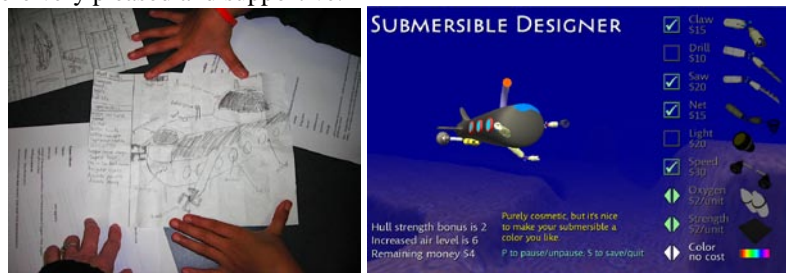


Fig. 3. Collaborative learning and game development projects can enhance the learning experience of elementary and university students. The completed submersible design game allows students to make choices that affect their ocean ecosystem exploration.

5 Conclusion and Future Work

It has been exciting to see how motivated students at our university have been with the addition of gaming education at our institution. The degree program has received very favorable reviews from students and has generated interest from perspective students. The gaming based versions of the software development courses have proven so popular that additional sections have had to be added to the course schedule. The community that our play and development group of students has developed is also impressive, and it is inspiring to see our university students making connections with and motivating the elementary students. In an era where student interest in STEM education seems to be waning, we hope that we have had a long-term impact on students to see that science, mathematics, and computer science are all interesting fields where creativity and imagination are key elements of success.

For the future, we will fully implement the degree program and expand the games flavor of course offerings to additional courses (specifically data structures). We expect to extend the process of game design as service into other areas. These include partnering with a non-profit group at Emory University to work on a game in the field of medicine for a mobile/handheld platform. We would like to extend the current oceanography game and make it available at aquariums such as the Tennessee and Georgia aquariums so others can explore and learn while playing. We have also proposed our sustainable service-based game development process into the field of engineering education to help students learn by completing virtual labs that take the place of difficult-to-obtain or hazardous laboratories. We look forward to developing these partnerships as well as others.

References

1. Aguilera, M. and Mendiz, A.: Video Games and Education (Education in the Face of a "Parallel School"). In *ACM Computers in Entertainment*, vol. 1, no. 1, pp 1-14. (October 2003)
2. Bennedsen, J. and Caspersen, M. E. 2007: Failure rates in introductory programming. *SIGCSE Bull.* vol. 39, no. 2, pp. 32-36. (Jun. 2007). DOI=<http://doi.acm.org/10.1145/1272848.1272879>
3. Church, D. et al: IGDA Curriculum Framework: The Study of Games and Game Development (beta 2.3). (February 2003) http://www.igda.org/academia/curriculum_framework.php.
4. Denis, G. and Jouvelot, P.: Motivation-Driven Educational Game Design: Applying Best Practices to Music Education. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, Valencia, Spain, pp 462-465. (2005)
5. Good, J. and Robertson, J.: Computer Games Authored by Children: A Multi-Perspective Evaluation. In *IDC 2004*, College Park, MD, pp 123-124. (June 2004)
6. Greitzer, F., Kuchar, O., and Huston, K.: Cognitive Science Implications for Enhancing Training Effectiveness in a Serious Gaming Context. In *ACM Journal of Educational Resources in Computing*, vol. 7, no. 3, art. 2. (November 2007)
7. Knudtzon, K., et al: Starting an Intergenerational Technology Design Team: A Case Study. In *IDC2003*, Preston, UK, pp 51-58. (July 2003)
8. Lewis, John: *C# Software Solutions: Foundations of Program Design*. Addison-Wesley, Boston (2007)
9. Malone, T.: 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, CA, pp 162-169. (1980)
10. Morrison, B. and Preston, J.: Engagement: Gaming throughout the Curriculum. In *Proceedings of the 2009 ACM SIGCSE Conference*, Chattanooga, TN. (2009)
11. Prensky, M.: Digital Game-Based Learning. In *Computers in Education (CIE)*, vol 1, issue 1, pp 21-24. (October 2003)
12. Steiner, B., Kaplan, N., and Moulthrop, S.: When Play Works: Turning Game-Playing into Learning. In *IDC 2006*, Tampere, Finland, pp 137-140. (June 2006)