Programming Serious Games as a Master Course: feasible or not?

Abstract

Background. Higher education on simulation & gaming has a long history with several master degrees offered by Universities worldwide. Currently, the popularity of Serious Games (SGs) has resulted in an increased interest on relevant master degrees. Despite the large number of master degrees on games in general, the literature on designing and evaluating relevant courses in higher education is rather limited.

Aim. This article aims at contributing on the field of SGs higher education by reporting on the design and evaluation of a graduate course on 'SGs Programming'. The course aims at providing students a holistic overview of SGs design, development and evaluation in a single course offered at a master degree that is not dedicated to simulation & gaming. The main goal of the pilot study presented was to investigate whether the proposed course can fulfill its objectives, as well as to draw conclusions on the design and deployment of the course that would be of interest to educators in the field of SGs.

Method. In order to evaluate the course in terms of students' satisfaction and outcomes a specially designed questionnaire was filled in and analyzed, along with students' performance in their artifacts and final exams.

Results. The results of the questionnaire combined with students' performance show that the course fulfilled its goals in a high degree.

Conclusions. Teaching SGs design and programming in a single master course is feasible by applying an iterative, project-based and microworld approach, utilizing both educational and professional tools and programming languages.

Keywords: Educational programming environments, game engines, serious games, serious games course, serious games design, serious games programming

1. INTRODUCTION

Serious games (SGs) are games that use entertainment not as their main goal, but as a means of motivating players for achieving a serious goal through playing (Zyda, 2005) and learning through experiencing (Kolb, 1984). SGs are used in various domains, such as education, training and simulation, health, public policy, strategic communication, human performance engineering and game evaluation (Zyda, 2005). We must highlight that the idea of using games for serious purposes has a much longer history than we would expect. Faria et al. (2009), for example, published a review on business simulation gaming over 40 years and provided several examples of using business simulation games in business schools. SGs, no matter what their target domain is, seem to have a positive effect (Connolly, Boyle, MacArthur, Hainey, & Boyle, 2012) on the digital natives (Prensky, 2001) of the 21st century. However, designing and implementing such games is a difficult task that requires technical skills and knowledge of various Computer Science topics, such as programming, graphics, math, physics, artificial intelligence, as well as pedagogy. This fact, along with the exponential growth of interest for SGs in industry (Laamarti et al., 2014; McGill, 2012) over the last decade, has resulted in a high demand for educating students on designing and developing SGs, as well as on carrying out relevant research (Crookall, 2010). Such education is nowadays offered at the graduate level with dedicated master programs, such as: the MSc on 'Serious Games Development' at the University of Glasgow, School of Arts, UK; the MSc on 'Serious Games' at the University of Skövde in Sweden; and the 'Games and Meaningful Play' MSc at Michigan State University, Media and Information Department.

In this article the main challenges and the critical decisions made on designing a master course on SGs Programming in the context of a master program on Applied Informatics, as well as the first evaluation results of the course are presented. The course aims at providing students a holistic overview of SGs design, evaluation and development, as well as relevant research directions. This is considered rather challenging since it is the only course related to games both in the undergraduate and postgraduate

curriculum at the Department of Applied Informatics at the University of Macedonia in Greece where the master course is offered. So, the motivation for the research presented in this article, as denoted by its title, is to study to what degree a single course with the aforementioned aims can be successful. The rest of the article is organized as follows. In section 2 a brief history of the term "serious games" is presented, and courses offered at master programs in (Serious) Games are briefly reviewed along with a brief literature review on evaluation results of relevant courses in higher education. In section 3 the course design is presented, giving emphasis on the main challenges, the critical decisions made and the adopted teaching approach. In section 4 the methodology for evaluating the course is presented and in section 5 the results are analyzed. Finally, some lessons learned and conclusions are presented.

2. RELATED WORK

Serious Games is a term that is being widely used the last few years. However, the idea of using games for other purposes than entertainment is not new. Literature on the history of SGs (Djaouti et al., 2011; Wilkinson, 2016) attributes the term to Clark Abt that used it with a meaning close to the current one in his book entitled "Serious Games" (Abt, 1970). Clark Abt was also the team leader of the Cold War simulation game T.E.M.P.E.R designed in 1961, which was one of several games designed during the Cold War (Djaouti et al., 2011). Some other pioneers, besides Clark Abt, back in the 1960s that contributed to the field are James Coleman, WilliamGamson, Garry Shirts, Richard Duke, Allan Feldt and Harold Guetzkow (Klabbers, 2009). Richard Duke and Allan Feldt contributed in urban and land use gaming with games such as Metropolis designed in 1964 and Community Land Use Game in 1966 respectively (Klabbers, 2009). Richard Duke also co-authored with the sociologist Cathy Greenblat a book providing a theoretical explanation of gaming and its practical application (Greenblat & Duke, 1975) revised in 1981 (Greenblat & Duke, 1981), and a book presenting three frame games for teaching a variety of subjects (Greenblat & Duke, 1979). All these people and many more have contributed to the evolution of simulation games that appear in the literature also as teaching games, instructional games,

educational games and serious games. The main difference of contemporary SGs is that they are mainly digital games, as denoted in the highly referenced definition of SGs provided by Zyda (2005).

Higher education on simulation & gaming has also a long history with several master degrees offered by several Universities. The increased interest in SGs has resulted in new master degrees on 'Serious Gaming' as denoted by Dr. Scott Turner in an article published in the Independent regarding the new "Computing (Serious Games)" master degree at the University of Northampton: "*There are a lot of game courses. This is one where we are just focusing on serious games rather than the more traditional games. So it is quite an unusual course in that respect. I think it's a growing market.*" (Richter, 2015). In Table 1 the master degrees on SGs and courses related to the material we intended to cover in our course are presented. The list is extended with some of the many master degrees offered in the field of games in general. Moreover, several MOOCs are offered from time to time on platforms like edX. Representative examples of courses are the following: "Design and Development of Games for Learning", Massachusetts Institute of Technology; "Leading Change: Go Beyond Gamification with Gameful Learning", University of Michigan in partnership with Microsoft; and several courses related to various "Game Design" issues offered by the Rochester Institute of Technology.

One main difference of the proposed course is that it is offered in the context of a master degree on Applied Informatics and aims at combining in a meaningful way material presented traditionally in a sequence of courses in master degrees dedicated to SGs. Our main hypothesis was that a course on SG design, development and evaluation would give students the chance to produce and evaluate real SGs fulfilling their initial goals and thus would be more appealing, in contrast with a course that would focus exclusively on a single aspect of SGs, such as design, evaluation or programming.

Table 1. Master Degrees related to (Serious) Games and relevant courses

MSc	Courses
Serious Games and Virtual Reality, University of Glasgow, Glasgow School of Arts, UK	Serious Game Design and Research Serious Games Development Game Programming
Serious Games, University of Skövde, Sweeden	Serious Games Research and Development Game Design Experimental Game Evaluation
Games and Meaningful Play, Media and Information Department, Michigan State University	Foundations of Serious Games Game Design and Development I
Computing (Serious Games) MSc, The University of Northampton, UK	Modeling for Serious games
Master of Entertainment Technology, Carnegie Mellon University	
Game Design and Development, Rochester Institute of Technology, NY	
Game Development and Research, Technische Hochschule Köln – University of Applied Sciences, Germany	
Informatik: Games Engineering, Technical University of Munich, Germany	
Game and Media Technology, Utrecht University, Netherlands	
Game Technology, NHTV Breda University of Applied Sciences, Netherlands	
MSc in Games, IT Unviversity of Copenhagen, Denmark	
Computer Games Technology, City University of London, UK	
Computer Games, University of Essex, UK	
Computer Games Technology, Abertay University, Dundee, Scottland	

Despite the large number of master degrees related to games, the literature on the design and evaluation of relevant courses in higher education is rather limited. This might be due to the fact that "*the simulation/gaming literature has progressed relatively slowly in regard to the validity of its various practices*" as noted by Wolfe and Crookall (1998; p. 16). One of the first papers regarding a capstone undergraduate course on "Design and Implementation of Computer Games" offered at Colby College during the winter semester in 1999 was presented by Jones (2000). This paper described the motivation, material, topics and projects of the course without any evaluation results. Kirk (2004) reported on his long-term experience on the making of the course "Design and Use of Gaming Simulations" offered at a master's degree on Human Resources at Western Carolina University. In addition to the course structure and content, the author presents some common barriers and solutions for the development of the course, as well as a few successful student-made gaming simulations that can be considered as indications for the success of the course. Taylor and Basket (2009) proposed an approach to teaching computer games development to undergraduate students aiming at balancing the artistic and scientific aspects of the

development process. As a means of evaluating the proposed approach the authors briefly discuss the work produced by students. Chaffin and Barnes (2010) presented some lessons learned from an experimental course on "Serious Games Research and Prototyping" offered at the University of North Carolina at Charlotte. Ritzhaupt (2009) presented a summary of the challenges in creating a "Game development" course with limited resources at the University of North Carolina Wilmington, along with the course design. The article includes an evaluation of the course based on an extended students' survey on various aspects of the course.

3. COURSE DESIGN

3.1 Course Objectives and Skills

The aim of the course is for students to acquire:

- basic knowledge of the role, the types and the features of SGs, as well as the whole process of devising a SG
- capabilities of designing and implementing SGs using contemporary tools, interfaces and programming languages
- knowledge and capabilities of using/devising evaluation metrics for SGs based on the aims defined during their design.

Upon successful completion of this course students will be able to: evaluate the design quality of SGs and the degree they fulfill their initial goals; design SGs taking into account various factors\design principles; implement simple SGs using the object-oriented programming technique and game libraries\engines.

3.2 Challenges and Critical Decisions

Designing and deploying a "Serious Games Programming" postgraduate course is accompanied with many challenges, which can be summarized as follows:

Challenge 1: A wealth of material and different subjects have to be covered in a short period of time

Games programming combines various Computer Science subjects (Chaffin and Barnes, 2010) and the time of a single course is limited for teaching all the material (Chaffin and Barnes, 2010; Kirk, 2004). In the case of a course on SGs, pedagogical issues and techniques for evaluating the learning outcomes have to be covered in addition and students don't have relevant knowledge and/or experience (Kirk, 2004). Consequently, it was clear that special attention should be paid on the syllabus of the course so as to contain all the fundamental concepts of the underlying subjects, omitting unnecessary details and providing students with a holistic overview of SGs design and programming. Special attention should also be paid on the teaching approach utilized in order to help students comprehend this wealth of material.

Challenge 2: What programming language should be used in the course?

Various programming languages are used in game programming: C++ is the most popular programming language in the game industry (Thin et al., 2013); Java is a cross-platform programming language that is widespread in higher education and is considered to be more appropriate for helping students concentrate on "*the higher-level concepts of game design and implementation*" (Jones, 2000; p. 262); C# has gained wide acceptance in the game industry, it is used in the popular game engine of Unity, and finally moving from C# to C++ is considered relatively easy (Thin et al., 2013). Taking into account the fact that several students might have limited technology skills (Kirk, 2004) in combination with the fact that in our institution and the majority of higher education institutions in Greece Java prevails, Java was considered to be a good choice for an introduction to game programming. However, C# is considered important for students in terms of employability, while the "Masters in Serious Games Curriculum Framework" proposes developing 2D games using C# for the "Serious Games Programming Core" module (Thin et al., 2013). Although, it is not so common to use more than one programming language in a games programming course, we decided to use and teach the use of both Java and C# for the aforementioned reasons.

Challenge 3: What programming tools should be used in the course?

Game development can be accomplished using various tools that support developers more or less in designing the graphics of the game and programming its logic. Our goal was to familiarize students with representative tools of all the main categories:

Greenfoot (Kölling, 2010) is a well-known *educational programming environment* that is used for introducing novices to OOP in Java with games and simulations. However, Greenfoot was considered ideal for the first part of the course for several reasons: it uses Java as a programming language; it simplifies the creation of the game world, the players and non-player characters through a straightforward library and a simple environment; it handles the game loop and gives the chance to postpone some game concepts for a subsequent time when students are more ready to absorb them.

GameMaker is a *game engine* targeted both to novice and professional game developers that want to expedite the development of games. This game engine was considered appropriate, since our goal was just to present the process of developing a game with a game engine in just a lecture.

Eclipse is a *professional IDE* that was used at the beginning of the course for presenting students the source code of freely available Java games (see Table 2). Microsoft Visual Studio that is widely used in the market was utilized for programming game concepts in C#.

Challenge 4: What should the number and type of assignments be?

Based on the goals of the course it was clear that three types of assignments should be used:

(i) *Designing SGs using existing design frameworks*: students should be able to comprehend and use existing SGs design frameworks for designing their games taking into account various factors and their interrelations, such as (Yusoff et al., 2009): instructional content, intended learning outcomes, learning activities, reflection, game attributes, game genre and game mechanics.

(ii) *Developing SGs* using the aforementioned programming tools.

(iii) *Evaluating SGs using existing evaluation frameworks and criteria*: students should be able to evaluate existing SGs taking into account various categories of criteria, such as (Sanchez, 2011):

motivation, autonomy, learning objectives, freedom, rules and feedback, emotional aspects and game integration. Emphasis is given on the degree that a SG fulfills its clearly defined serious purpose.

The main dilemma regarding the assignments was whether several short assignments (on a weekly basis) or less and more advanced assignments should be used. The main advantage of the weekly assignments is that they demand from students that they work throughout the course on the material of each and every lecture. However, more demanding assignments require students to synthesize and deepen on the acquired knowledge, to use their creativity and deliver an artifact that they can show case.

3.3 Teaching Approach

In order to deal with the aforementioned challenges and support the decisions made, it was necessary to adopt an appropriate teaching approach. It was clear from the very beginning that adopting a teaching approach similar to the microworld or mini-language approach to teaching introductory programming would be helpful, at least for the modules on game programming. Based on the *microworld approach* to teaching programming (Brusilovsky, Calabrese, Hvorecky, Kouchnirenko, & Miller, 1997): a language with simple syntax and semantics is used in order to help novices focus on the important aspects of programming; the language is built on a metaphor that is engaging and visually appealing and is usually based on some sort of an actor; interesting problems are solved, while the actions of the actor that is programmed are constantly visualized. This microworld approach to teaching game programming was applied in the proposed course by:

- Using at the beginning of the course a programming language that students were familiar with (Java in our case) and that would help them concentrate on programming fundamental game concepts (instead of the syntax of the language) from the very beginning.
- Using Greenfoot as the first game programming tool. Greenfoot supports Java and has the following desirable features: the Application Programming Interface (API) of Greenfoot contains a limited number of simple classes that give the chance to focus on experiencing and programming the most important game

features at the beginning, leaving out the details; it uses a metaphor of worlds (levels) and actors (players, non-player characters, bots and so on) that can be easily extended; it provides direct manipulation and visualization features that make both the development and the comprehension of the mechanics of games easier. Greenfoot was expected to give students the chance to develop their first working games in a short period of time and help them experiment on various game-related concepts on their own.

Moreover, it was decided to use an *iterative* and *project-based* approach that would provide the chance for *active learning*. This means that issues on design, evaluation and development of SGs would be presented through existing games that should be: critically evaluated regarding their design and achievement of initial goals using the design and evaluation frameworks analyzed in the course; analyzed in terms of the game features already implemented and extended in order to improve user experience or add new features in the game. An iterative approach was also considered important both for revisiting the way that game features are implemented in different languages and tools and also for a deeper and more complete comprehension of these game features.

3.4 Syllabus and assignments

Based on the goals of the course, its contents, as well as the aforementioned decisions the syllabus of the course was organized in the modules presented in Table 2. In order to fulfill the objectives of the course four assignments were designed constituting 40% of the final grade. The assignments are presented in Table 3 along with the related course modules, deliverables and intended learning outcomes.

Table 2. Syllabus of the course.

Week	Topic	Content	Programming activity
1	Foundations of SGs	The role of serious games as tools for educating, skills acquisition and simulation in various sectors, such as education, health and business processes; Game genres (i.e. puzzle, strategy, role-playing games); Types and features of SGs; Review of representative examples of serious games.	Analysis of the text-based adventure game "World of Zuul" in Java
2	Designing & Evaluating SGs	Design frameworks and principles for SGs; Using design frameworks for designing, but also for evaluating the design quality of existing SGs and the degree they fulfill the initial goals. Issues on designing SGs are revisited in the subsequent weeks while programming games and in the context of the assignments. A final overview on design and evaluation of SGs is provided in Week 11.	Analysis of a "predator-prey" simulation in Java. Emphasis is given on the simulation loop, data structures for actors, improving GUI.
3-5	Programming games with Greenfoot	Review of tools for developing games: Educational Programming Environments (Greenfoot, Alice, Scratch); Game Engines (GameMaker, JMonkey, Unity), Programming Languages (Java, C++, C#, Python), Frameworks. Using Greenfoot for presenting fundamental game concepts: the world of the game, creating actors, images, sound, locating the edges of the world, random behavior, controlling actors with keyboard, movement, smooth movement, movement vector, physics, collision detection, calculating scores, score board.	A simple action game. A simulation. A first person shooter game.
6	Programming games with GameMaker	Features and examples of game engines. Using GameMaker for developing a game.	Following a tutorial for developing a simple game.
7-10	Programming games with C#	Designing graphics: text, shapes, images, rotating & inverting images; Designing and moving sprites: implementing a class for representing, designing and moving sprites; Collision detection; Game loop; Designing and creating game levels: tilemap editor, tile palette, loading level data from an XML file, scrolling; Adding objects during game play, moving sprites in scrolling maps, adding portals; Dialogue between characters and creation of an inventory.	Several existing projects for presenting the underlying concepts and experimenting with the source code.
11	Evaluating SGs	Evaluation frameworks and questionnaires for SGs (revisited).	
12	Case study: the SG CMX	CMX – a SG for learning programming: MMORPG, design framework of CMX, implementation challenges, incorporating learning analytics, evaluation framework of CMX.	Using a version of CMX

Table 3. Overview of assignments.

Assignment	Related course modules	Deliverable	Learning outcomes
Evaluating the design quality of an existing SG using a design and/or evaluation framework	Foundations of SGs (1) Designing & evaluating SGs (2)	Evaluation report for a SG of students' choice	Searching and using SGs. Comprehending and applying a design - evaluation framework for describing and assessing the degree that the SG fulfills its goals
Implementing a simple game in Java	Programming games with Greenfoot (3-5)	Design & implementation of a simple game in Java. Brief report with the goals & rules of the game, documentation of source code.	Designing the first (serious) game. Implementing a simple game with all the fundamental features presented in the course.
Implementing a prototype of a SG in C#	Programming games with C# (6-10)	A functional prototype of a SG in C#	Utilizing and extending the C# classes presented in the course as a game engine for implementing a prototype of a SG
Designing and developing a SG	All modules	A functional SG. Brief report with the analysis and design of the SG.	Implementing an entertaining and immersive SG, taking into account guidelines and criteria for making it successful in fulfilling its goals

4. COURSE EVALUATION METHODOLOGY

4.1 Research Questions

The main research question of the pilot study is the following:

> *RQ: Does the proposed course on Serious Games Programming fulfill its goals, both in terms of students' satisfaction and performance?*

Since the course was offered for the first time and also the instructor was teaching game design and development for the first time, it was of particular interest - just as Ritzhaupt (2009) stated - to investigate students' level of satisfaction with the course. Specifically, we wanted to investigate students': factors for course selection; initial expectations and final satisfaction with the knowledge and skills acquired; satisfaction with the syllabus and the educational material; satisfaction with the assignments in terms of the help provided for achieving the goals of the course; satisfaction and agreement with the choice of programming languages and tools.

Besides students' satisfaction with the course, we considered necessary to analyze students' performance and more specifically the deliverables of the assignments as in (Kirk, 2004; Ritzhaupt, 2009) and the grades in the final exams that provide more objective insights for the success of the course.

4.2 Participants

The majority of the 18 students (11 male & 7 female) that attended the course had a BSc degree in Informatics/Computer Science (89%), while one student had a degree in Applied Mathematics and one in Marketing and Advertising. The course was selected by 13 out of the 20 students studying in the "Computer Systems and Network Technologies" stream of specialization were the course is offered. Also, it was selected by 3 students from the "Business Computing", one student from the "Computational Methods and Applications" and one student from the "E-Business and Innovation" stream of specialization (students can select one course from the other specializations). All of them were full-time students.

4.3 Research instruments and data analysis method

In order to investigate students' satisfaction with the course, a questionnaire designed by the instructor and delivered online was used. The questionnaire included 30 closed-type questions and 1 open-type question for comments and suggestions, covering various aspects of students' satisfaction as described in section 4.1. Students expressed how satisfied they were with each element of the course in a 5-point Likert-type scale: not at all (1), a little (2), averagely (3), much (4), very much (5). This questionnaire was answered by 14 out of 18 students and the median and range (R = max value - min value) for each question was calculated. Moreover, results (mean) from an anonymous questionnaire administered by the Quality Assurance Unit of the Institution (answered by 16 out of 18 students) were used. Finally, a qualitative analysis of students' performance in their artifacts created in the context of the assignments and the final exams is presented.

5. ANALYSIS OF RESULTS

In this section students' evaluation results regarding various aspects of the course are analyzed. Moreover, students' suggestions are presented and their performance is analyzed.

5.1 Factors for Course Selection

When examining the reasons that led students to select the course on Serious Games Programming, we found out that the main reason for half the students was the fact that *"the course seemed interesting".* It is quite surprising though that the majority of them (64%) did not know in advance what SGs are, while several students stated that they did not (14%) or rarely (29%) played even entertainment games. A small percentage of students (14%) stated that selected the course because they *"like programming"*, and even fewer students (7%) because they *"like games"*. Nearly one third of students (29%) stated that selected the course because it seemed interesting and they liked both games and programming.

5.2 Overall Satisfaction with the Course

Since the course was offered for the first time it was important to investigate at what degree it fulfilled students' expectations, whatever they were. It came out that students' *expectations* were much (57%) or very much (29%) met for the majority of them (Median=4, R=2). Specifically, nearly four out of five students were much or very much satisfied with the *knowledge* acquired (Median=4, R=2) and the *skills* developed (Median=4, R=2) in the context of the course. Regarding the skills developed, the skills in *evaluating* SGs were considered the strongest (Median=4, R=2) for the majority of students, followed by the skills in *designing* SGs (Median=4, R=2) and finally the skills in *developing* SGs (Median=4, R=3). The aforementioned results are in conformance with the results of the questionnaire administered by the Quality Assurance Unit of the Institution. The relevant questions and mean values are:

- '*Was the quality of the course high?*' – Mean = 4.12
- '*Was the organization and presentation of the course impeccable?*' – Mean = 4.56
- '*Was the field of the course interesting and useful for your studies?*' – Mean = 4.25

5.3 Satisfaction with Syllabus and Educational Material

An important aspect of any course is its syllabus and the educational material prepared for supporting it. All the students, with just one exception, were much (64%) or very much (29%) satisfied with the *syllabus* of the course (Median=5, R=2) presented in Table 2. The majority of students were also much (43%) or very much (43%) satisfied with the *quantity of the content* covered in the course (Median=4, R=2). Students also evaluated positively the *material for Greenfoot* (Median=5, R=2), SGs (Median=4.5, R=2) and *game programming with C#* (Median=4, R=2). Students were not so satisfied with the *material for GameMaker* (Median=3.5, R=2) that was mainly based in two tutorials incorporated in the engine and were covered in just one lecture. In the questionnaire administered by the Quality Assurance Unit of the Institution the mean in the question '*Was the educational material (books, notes, exercises, papers etc) sufficient for the needs of the course*' was 4.19.

5.4 Evaluation of the Effectiveness of Assignments

The majority of students were much or very much satisfied both with the *number* (86%) and the *type* (78%) of the *assignments* (Median=4, R=3 & Median=5, R=3 respectively). At this point we must state that the 3rd assignment was not realized due to time limitations and the complexity of the rest of the assignments. Although, several students stated that the three out of four assignments that were initially scheduled demanded a great deal of work and time, the majority of students stated that the assignment for extending the projects presented in C# would help much (43%) or very much (29%) (Median=4, R=4). Students evaluated positively the *effectiveness* of the assignments (especially the programming ones) in terms of helping them achieving the goals of the course. Nearly all the students (93%) were much or very much positive about the two programming assignments. Students stated that these assignments helped them in *designing and programming simple games in Java* (Median=4.5, R=2) and in *designing and programming SGs* (Median=5, R=2). However, the first assignment also helped the majority of students (72%) in designing and *evaluating* SGs.

5.5 Agreement with the Choice of Programming Languages

One of the main decisions taken during designing the course was the use of two distinct programming languages and various programming environments and tools. Although our rationale seemed sound for us, it was of great importance to investigate at what degree the students agreed with this choice. We were delighted to see that the majority of students agreed much (14%) or very much (71%) with *using both Java and C#* for presenting fundamental game programming concepts (Median=5, R=3). Just a few students would prefer using only C# throughout the course (Median=2, R=4) and even fewer only Java (Median=1.5, R=4).

5.6 Agreement with Choice of Game Programming Tools

Students also agreed with the choice of the game programming tools utilized in the course:

- Greenfoot helped the students much (43%) or very much (50%) in *comprehending fundamental game programming concepts* (Median=4.5, R=2). Moreover, all the students agreed that Greenfoot helped them *develop a fully functional game in a short period of time* (Median=5, R=1).
- Microsoft Visual Studio helped students *focus on important issues on game programming*, making easy the development of the basic GUI (Median=4, R=2).
- GameMaker helped the majority of students much (43%) or very much (21%) in *comprehending how a game machine functions* in a short period of time (Median=4, R=3). However, for one out of five students (21%) the assistance provided was of little importance.

5.7 Students' Suggestions

In the open type question where students could comment on any aspect of the course, and also in the context of informal interviews the following suggestions were made:

- Four students would prefer to devote less time in developing games with Grenfoot and more time in game programming with C#.
- Although, effort was made to present theoretical and programming concepts through examples and complete projects, as well as to have students make changes or extend the given source code it seems that there is a need to apply *active learning* in a larger extent, as proposed by Ritzhaupt (2009). Five students mentioned in one way or another that they would like to participate even more actively in each lecture by implementing new features in the presented games. For example, the extensions given for practice at home could be implemented within the course.
- Two students would like to be presented with more theoretical issues about SGs.

5.8 Students' Performance

Students' performance in the assignments and the final exams is presented in Figure 1, while a qualitative analysis of the assignment deliverables is presented in Table 4. The grades of both the assignments and the exams are in the scale [0..10], however they participated in the final grade with the percentage of 40%

and 60% respectively. The majority of students showed a uniform performance in the assignments and the final exams. However, one out of three students did not have a uniform performance. Specifically, students 2, 4, 11, 13 and 17 achieved a considerably higher grade in the exams. When students were interviewed it came out that four of them could not devote the necessary time to the assignments because they worked full-time, while one of the students over-estimated his work and had the impression that it was of much better quality. Students 11 and 13 that received the grade of 2 did not submit the first two assignments. On the other hand, student 16 that had an excellent performance in the assignments could hardly pass the exams.

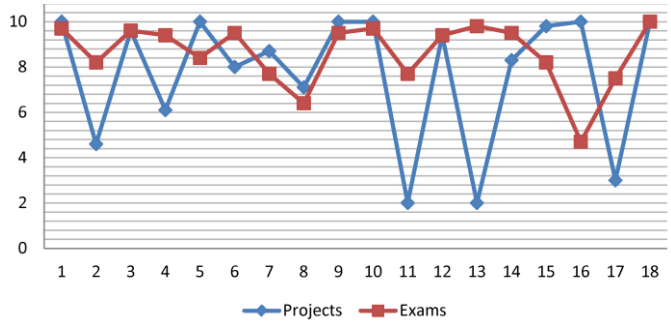

Figure 1. Students' performance in assignments and exams.

Based on the deliverables of the first assignment, students comprehended the utilized frameworks and all the fundamental SG design issues at a good level and analyzed the selected SGs in a clear and concise manner. Indicatively, the "Conceptual Framework" (Yusoff et al., 2009) that was used by half the students takes into account the following factors: capabilities (skills to develop through playing); instructional content; intended learning outcomes; game attributes; learning activities; reflection; game genre; game mechanics; game achievement. Based on the results of the second assignment, it came out that students were able to develop a simple Java game at this point in the course, but none of them developed (and most probably could not) a SG. This was expected and that was the reason for not asking students to develop from the very beginning a SG. The results from the last assignment on designing and

implementing a SG were quite encouraging. Five of the eighteen SGs created were positively judged by stakeholders and presented in two National Conferences.

Table 4. Qualitative analysis of deliverables for the assignments.

Assignment	Outcomes
Evaluating the design quality of an existing SG using a design and/or evaluation framework	• **Submissions**: 16 out of the 18 students submitted the assignment • **Types** of SGs analyzed: - *Educational games,* such as 'LightBot' for programming and 'Lure of the Labyrinth' for mathematics - *Awareness games* for energy and environmental issues, such as 'Electrocity', 'Bioharmonious', and 'Fate of the world' - *Social problems awareness games*, such as 'Spent' for unemployment, poverty and homeless people; - *Emergency preparedness games*, such as 'Young Meteorologist Program' for extreme natural phenomena. • **Design/evaluation frameworks**: various frameworks were used (Annetta, 2010; Ibrahim & Jaafar, 2009; Malliarakis et al., 2014; Mitgutsch & Alvarado, 2012; Yusoff et al., 2009) • **Most common framework**: half the students used the Conceptual Framework (Yusoff et al., 2009)
Implementing a simple (serious) game in Java	• **Submissions**: 16 out of the 18 students submitted the assignment • **Game genres**: mainly shooter and action games • **Tools**: Greenfoot and Java • **Implemented game features**: all the required game features were implemented
Designing and developing a SG	• **Submissions**: all students submitted this assignment • **Tools**: 16 students developed their SG using Java and Greenfoot and 2 students C# • **Best SGs –** the best SGs were educational games: (1) MATH OPERATIONS - addition and subtraction ; (2) MATH BALLOONS - multiplication & divisibility; (3) RAINING LETTERS – spelling; (4) MR. FINGERS - learning the keyboard; (5) FOOD BALL - Awareness about good food habits; (6) learning the basics about investments and economy. • **Stakeholders**: - SGs (1), (2) and (3) were used in a pilot study by 245 students and 21 teachers in two Primary Schools. Both teachers and students evaluated positively the 3 SGs. - SGs (4) and (5) were extended and awarded in the context of an open source operational programme. Their source code is open to the educational community for extension and/or usage in new games. - The aforementioned games were presented and published in two National Conference papers.

6. LESSONS LEARNED AND CONCLUSION

The aim of this article was to present and evaluate a single MSc course on 'Serious Games Programming' offered in a master on Applied Informatics. The course provides a holistic overview of SGs design, development and evaluation and fulfilling its goals was considered rather challenging. The fact that we could not find a similar course - but instead a sequence of courses - to rely on, combined with the limited literature on evaluation of related courses in higher education, led us in the decision to evaluate students' satisfaction with the course and their performance after its first offering.

The microworld approach to introducing students to the main concepts of game programming using Greenfoot had a positive impact on presenting a wealth of material in a limited time (Challenge 1). It turned out that Greenfoot allowed students to develop their first fully functional games in a short period of time. The decision to use Java at the beginning of the course and gradually move to C# was also positively evaluated by students (Challenge 2). It is important to state, however, that an iterative and project-based approach was applied focusing on game and SG concepts and relevant programming skills rather than specific tools. It turned out that students managed to focus on comprehending the main concepts and acquiring the necessary skills and applying them using a variety of tools, which were positively evaluated (Challenge 3). The role of assignments for evaluating, designing and implementing SGs was considered of vital importance (Challenge 4), while active learning should be reinforced (Ritzhaupt, 2009). Students' demand for active learning in the context of the lectures was probably the most prominent direction for future work and surely not an easy task for such a demanding course.

It is clear that in order to draw safer conclusions a longer-term and more in-depth evaluation using more instruments, such as semi-structured interviews, has to take place. This will provide greater insights into how and why student learning proceeds in the course and will give us the chance to improve it. The results of the first evaluation proved that providing such a course is actually feasible and worth the time and effort to keep working on it and moreover it helped us draw conclusions on some aspects of the course that worked well and could be adopted by other educators as well:

- *Use a microworld approach to introducing students to game programming*: using a tool that helps students focus on experiencing and programming the most fundamental game concepts in a simple and straightforward manner can help students comprehend in a short period of time the foundations of game programming and deliver their first fully functional products.
- *Start with a programming language that students are familiar with and move to a production language using an iterative approach*: although specific programming languages have prevailed in

the game industry, educators should consider introducing students to game programming with a language that students are already familiar with. Combined with the microworld approach, this will help students to acquire a solid background on game programming and afterwards to transfer their knowledge in another production language (Xinogalos 2012) used in the game industry, using an iterative approach.

- *Apply active learning*: active learning is considered important for students and educators should strive to provide opportunities for active learning in the context of lectures even for theoretical issues regarding the design and evaluation of SGs.

REFERENCES

Abt, C. C. (1970). Serious games. New York, NY: Viking Press.

Annetta, L. A. (2010). The "I's" have it: A framework for serious educational game design. *Review of General Psychology*, *14*(2), 105-112.

Brusilovsky, P., Calabrese, E., Hvorecky, J., Kouchnirenko, A. & Miller, P. (1997). Mini-languages: a way to learn programming principles. *Education and Information Technologies*, *2*(1), 65-83.

Chaffin, A., & Barnes, T. (2010). Lessons from a course on serious games research and prototyping. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games* (FDG' 10). ACM, New York, NY, USA, 32-39.

Connolly, T. M., Boyle, E. A., MacArthur, E., Hainey, T. & Boyle, J. M. (2012). A systematic literature review of empirical evidence on computer games and serious games. *Computers & Education*, *59*(2), 661-686.

Crookall, D. (2010). Serious games, debriefing, and simulation/gaming as a discipline. *Simulation & Gaming*, 41, 898-920.

Djaouti, D., Alvarez, J., Jessel, J. P., & Rampnoux, O. (2011). Origins of serious games. In Ma, M., Oikonomou, A., & Jain, L. C. (Eds.), Serious games and edutainment applications (pp. 25-43). London, England: Springer.

Faria, A. J., Hutchinson, D., Wellington, W. J., & Gold, S. (2009). Developments in business gaming a review of the past 40 years. *Simulation & Gaming*, *40*(4), 464-487.

Greenblat, C. S., & Duke, R. D. (Eds.). (1975). Gaming-simulation: Rationale, design, and applications. New York, NY: Halsted Press/Wiley.

Greenblat, C. S., & Duke, R. D. (1979). Game generating games: A trilogy of games for community & classroom. Beverly Hills, CA: SAGE.

Greenblat, C. S., & Duke, R. D. (Eds.). (1981). Principles and practices of gaming-simulation. Beverly Hills, CA: SAGE.

Ibrahim, R. & Jaafar, A. (2009). Educational Games (EG) design framework: Combination of game design, pedagogy and content modeling. In *Proc. IEEE Int. Conf. Electr. Eng. Inform.* (Vol. 1, pp. 293-298). IEEE.

Jones, R. M. (2000). Design and implementation of computer games: a capstone course for undergraduate computer science education. *ACM SIGCSE Bulletin*, 32(1), 260-264.

Kirk, J. J. (2004). The making of a gaming-simulation course: A personal tale. S*imulation & Gaming*, *35*(1), 85-93.
Klabbers, J. H. (2009). The magic circle: Principles of gaming & simulation. Rotterdam, The Netherlands: Sense Publishers.
Kolb, D. A. (1984). Experiential Learning – Experience as The Source of Learning and Development. Englewood Cliffs, NJ.: Prentice Hall. 256 pages.
Kölling, M. (2010). The Greenfoot programming environment. *ACM Transactions on Computing Education (TOCE)*, *10*(4), Article 14.
Laamarti, F., Eid, M. & Saddik, A. E. (2014). An Overview of Serious Games. *International Journal of Computer Games Technology*, vol. 2014, Article ID 358152, 15 pages.
Malliarakis, C., Satratzemi, M. & Xinogalos, S. (2014). Designing educational games for computer programming: A holistic framework. *Electron. J. E-Learn.*, *12*(3), 281–298.
McGill, M. (2012). The Curriculum Planning Process for Undergraduate Game Degree Programs in the United Kingdom and United States. *Trans. Comput. Educ.*, *12*(2), Article 7 (April 2012), 47 pages.
Mitgutsch, K. & Alvarado, N. (2012). Purposeful by design?: a serious game design assessment framework. In *Proc. of the International Conference on the foundations of digital games*. ACM, New York, NY, USA, 121-128.
Prensky, M. (2001). Digital natives, digital immigrants part 1. *On the horizon*, *9*(5), 1-6.
Richter, H. (2015). University of Northampton offers students master's degree in 'serious gaming. Independent. Friday 18 September 2015. http://www.independent.co.uk/student/news/university-of-northampton-offers-students-masters-degree-in-serious-gaming-10508086.html
Ritzhaupt, A.D. (2009). Creating a Game Development Course with Limited Resources: An Evaluation Study. *Trans. Comput. Educ. 9*(1), Article 3, 16 pages.
Sanchez E. 2011. Key criteria for Game Design. A Framework. Retrieved from: www.reseaucerta.org/meet/Key_criteria_for_Game_Design_v2.pdf
Taylor, M. J., & Baskett, M. (2009). The science and art of computer games development for undergraduate students. *ACM Computers in Entertainment*, *7*(2), Article 24, 9 pages.
Thin, A. G., Lim, T., Louchart, S., De Gloria, A., Mayer, I., Kickmeier-Rust, M., Klamma, R., VeltKamp, R., Arnab, S., Bellotti, F., Boyle, L., Prada, R., Westera, W., Nadolski, R., & Abbas Petersen, S. (2013). Masters in Serious Games Curriculum Framework. Deliverable 5.3 of the Games and Learning Alliance Network of Excellence. Available at http://www.seriousgamessociety.org/download/SGMastersFwk.pdf.
Wilkinson, P. (2016). A brief history of serious games. In Dörner, R., Göbel, S., KickmeierRust, M., Masuch, M., Zweig, K.A. (Eds.), Entertainment computing and serious games, LNCS 9970 (pp. 17-41). Basel, Switzerland: Springer International Publishing AG.
Wolfe, J., & Crookall, D. (1998). Developing a scientific knowledge of simulation/gaming. *Simulation & Gaming*, *29*(1), 7-19.
Xinogalos, S. (2012). An Evaluation of Knowledge Transfer from Microworld Programming to Conventional Programming. *Journal of Educational Computing Research*, *47*(3), 251-277.
Yusoff, A., Crowder, R., Gilbert, L. & Wills, G. (2009). A conceptual framework for serious games. In *Proc. 9th IEEE Int. Conf. Adv. Learn Technol.* (pp. 21-23). IEEE.
Zyda, M. (2005). From visual simulation to virtual reality to games. *IEEE Computer*, *38*(9), 25-32.