Preface

The game industry is one of the fastest growing sectors in the worldwide economy (Zackariasson, 2012). According to the research company Gartner, global game sales may have reached $111.1 billion in 2015, due in part to the growth in mobile game play and the recent release of the new generation of game consoles. In order to increase engagement and player retention, video games include several common features such as leaderboards and achievements. The massive use of this approach and the impressive growth of the number of players led to the concept of gamification as a service and its use in broader domains. In fact, the use of game concepts and mechanics in non-game contexts is an effective way to engage users. Gamification is currently a word of order in different domains, from marketing to e-learning (Hamari, 2014). One of such domains is Computer Science.

Computer Science is the study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, and their impact on society (Tucker, 2003). In the education context, Computer Science encompasses several topics such as algorithmic problem-solving, computing and data analysis, human-computer interaction, programming, security, Web design, robotics and many others. In Computer Science courses students develop computational and critical thinking skills and how to create, not simply use, new technologies.

THE CHALLENGES

Despite the need to learn Computer Science by every student in every school, there are many issues that must be addressed. Introductory programming courses are characterized by an extensive curricula and a high enrolment of students. This poses a great workload for faculty and teaching assistants responsible for the creation, delivery, and assessment of student exercises. These courses are also regarded as difficult and often have high failure and dropout rates (Ala-Mutka, 2005; O’Kelly, 2006; Robins, 2003). Researchers have pointed out several causes for these rates from teaching methods (lectures on programming language syntaxes), passing by the lack of feedback, to the subject complexity. All these issues pose huge challenges in the computer programming teaching-learning process and also opens the door to new methodologies and techniques aiming to improve retention and foster the motivation and competitiveness of computer programming learning.

While the concept of “winners and losers” can hinder the motivation of students (Vansteenkiste & Deci, 2003), competitive learning is a learning paradigm that relies on the competitiveness of students to increase their programming skills (Burguillo, 2010; Siddiqui et al’, 2008).
This book presents a comprehensive and recent view of the different pedagogical and technological strategies to address computer programming learning issues through the use of gamification techniques.

Gamification is the use of game mechanics in non-game contexts to engage users in solving problems and increase users’ contributions. The complexity of computer programming domain and the advantages of using gamification are the kick-off for this book that aims to enumerate and explain different approaches, techniques and methodologies for integrating gamification in the teaching-learning process of computer programming. These contributions included several game design principles, an interesting set of gamification strategies in the computer programming learning domain and also includes various frameworks and tools to improve students’ motivation and progress.

DESCRIPTION AND ORGANIZATION OF THE BOOK

This book presents a comprehensive and recent view of the issues of learning computer programming and shows different pedagogical and technological gamification strategies to address these issues. At the same time, it identifies new trends on this topic from pedagogical strategies to technological approaches.

The book is organized into 14 chapters. A brief description of each of the chapters follows:

Chapter 1 presents and summarizes a variety of Game Backend services. The goal of these cloud services is to free the programmer of the implementation of the game infrastructure and give him more time to dedicate in the game logic. The idea is to not replicate the implementation of the game features in each version of the game for several platforms and adhere to a service oriented architecture providing cross-platform game services that lets you easily integrate popular gaming features such as achievements, leaderboards, remote storage and real-time multiplayer in games. This chapter surveys several Game Backend services based on their technical and social features. In general, all the services studied provide a set of advanced social features that fosters the use of gaming cloud services. Some distinguish points are related with technical points such as the pricing strategy or the platforms supported. Regarding their openness, the potential use of them for the integration with complex domains such as, the computer programming domain is enormous. This study is part of an effort to select a set of services on which to base the development of a service for gamification of learning activities.

Chapter 2 presents the subjects on a Computer Science course based on game activity for the purpose of acquiring the required competences. The following points are developed: 1) technical skills: to devise a simple and basic game that enables students to learn by playing as well as being able to contribute more components to the game itself; 2) social skills: since this is an interactive multiplayer game, players are required to interact and to collaborate with each in order to arrive at the final solution. This sets the challenge to improve and develop social skills in a positive and mutually helpful way; 3) creative skills: the design and appearance of the game should give participants a feeling and understanding of how to follow the game. This will teach them the importance of the solution design from the point of view of the final user. Thus, they will be able to propose the most appropriate framework for building new components or even improve existing ones in order to make them more appealing and/or user-friendly.

Rather than providing a specific game, the authors present a design of the main features that a game should have in order for students to acquire the corresponding skills, the technical skills provided from the course syllabus, and the social and creative skills established by the course designer.

Chapter 3 aims to bridge the gap in the literature on the thorough literature consolidation of educational computer games, educational video games, and serious games. The extensive literature of educational
Preface

computer games, educational video games, and serious games provides a contribution to practitioners and researchers by describing the multifaceted applications of educational computer games, educational video games, and serious games in the game-based learning environments.

Chapter 4 identifies features that promote computer programming learning in a digital game context. Consequently, this study presented ten principles that emerged from this experience. Some of these principles are focused on the game elements and others on instruction. An active monitoring and support with a substantial amount of players (60 students) fostered the development of an operational casual serious game. Its source code is available into a version control repository promoting its reutilization and speeding up studies on serious game development and use.

Chapter 5 shares information about an introductory Java programming course by incorporating Problem-Based Learning (PBL) and game design. While the course has been successful in a face-to-face classroom setting, transitioning to an e-learning environment presents some challenges. This chapter describes aspects of the course structure, content and delivery, and makes recommendations to help transition those aspects to the e-learning environment. Specifically, the objectives of the chapter are to 1) provide a brief discussion about the difficulty of learning to program and the usefulness of certain approaches to course design for the introductory programming course, 2) present pedagogical strategies that have been successful in the authors’ introductory programming course, and 3) discuss the implementation of those pedagogical features in an e-learning environment.

Chapter 6 describes the required processes to carry out the gamification of a Parallel Programming course, together with some interesting results and situations. The main purpose of this work is to publish not only the positive results obtained, but also describe the tool used, the experience process, the feedback received, and our conclusions regarding the gamification usefulness.

Chapter 7 discusses the difficulties that students encounter in programming and offering possible solutions and approaches that provide motivational and enjoyable learning environments for programming are proposed. Additionally, gamification strategies can be used in an educational software process are presented. For the purpose of presenting scientific data on the effectiveness of the gamification approach, the experimental design and findings of a sample research were shared with readers at the end of the chapter. The results of this research revealed that gamified environments support motivational improvements and the academic growth of individuals significantly. However, the literature still lacks studies on using different pedagogical techniques in programming language education. Further research is required to carry out the effectiveness of game-based approaches in programming language training at different age groups.

Chapter 8 presents several experiences in the context of CP course for the master’s degree for different kinds of audience, introducing new innovative solutions with mini Games (mG) to overcome new difficulties brought out by context evolution (new participants’ expectations with the dialogism between diploma and personal evolution toward targeted skills and the decrease of resources: time and funding). The underlying topic is the pedagogical innovation act of trainers in their specific context, referring to the U theory of innovation by Scharmer (2009) and with a Quality Assurance (QA) concern with respect to the challenge of Higher Education and the success of the met students, apprentices and learners in alternation; adding higher constraint and less flexibility in the schedule with cost constraints too.

Chapter 9 presents the development and results of a project conducted in order to investigate the acceptance and feasibility of the use of 3D Virtual Worlds integrated to remote experimentation laboratories, in science classes at Brazilian public schools. Therefore, this initiative includes the development of a 3D virtual environment entitled Virtual World of Plants, integrated to a microscope remote lab, focused on
the teaching of science in basic education. This environment was contextualized with the content on the morphology of angiosperms, with a trail where students, through their avatars, can transit through the environment that presents several theoretical concepts that will be viewed dynamically, such as videos, quizzes or other learning objects. After transiting the environment and interact with learning objects students can access the remote experiment to consolidate the concepts learned.

Chapter 10 focuses on how teaching object-oriented programming can be done resourcing to a game-programming framework. To illustrate how the author see it, he uses two different programming languages (although based on a same paradigm and a virtual machine): C# and Java. And for each one he chooses a game development framework: the MonoGame Framework (the open-source fork of XNA) for the C# language, and LibGDX Framework for the Java language. This way the author is able to demonstrate how basic OO programming concepts can be taught using different languages, and different platforms: from Windows native applications to Android applications.

Chapter 11 presents an innovative game-based educational tool for Moodle to support the teaching and learning process of programming languages and paradigms for engineering students of several study programs. The experiences were carried out during the academic year 2015 – 2016 in two courses with positive results.

Chapter 12 presents Code, a friendly environment where students without previous programming experience can explore core-programming concepts in a motivating manner. It consists of a platform style game where students can control a virtual robot having their first contact with the construction of small pseudocode blocks. Research in teaching and learning of programming present several reasons for the difficulties students face when starting programming. In this chapter we have pointed out some of them. The studies performed also indicated that the aspects related to problem solving are the principal factors to attack in order to solve this problem. However, the main concern is how to provide a motivating environment that engages the student in computational thinking and problem solving. Code Factory’s main objective is to create a new strategy making learning more stimulating for the students emphasizing problem solving in order to help develop fundamental programming skills.

Chapter 13 presents four types of scripting environments that are pivotal for the development and usage of gamified Learning Management Systems in programming education were identified: the student’s code execution environment; the scripting wrapper for processing input and output of the code execution environment; the game-based rule processing environment; and the environment for processing scripts embedded in the course content. The functionality of each of these environments was described, with exemplary usage scenarios provided. The key requirements for each environment were specified, and adequate design and technological solutions suggested. For open design decisions, consequences of choosing particular options were investigated.

Chapter 14 presents the architecture and design of Enki, an Integrated Development Environment (IDE) for learning programming languages on Massive Open Online Courses (MOOCs). This environment can be used as a tool by a Learning Management System (LMS) and a typical LMS such as Moodle can launch it using the Learning Tool Interoperability (LTI) API. Student authentication tokens are passed via LTI, thus integrating Enki in the single sign-on domain of the academic institution. The proposed tool has a web user interface similar to those of reference IDEs, where the learner has access to different integrated tools, from viewing tutorial videos, to solving programming exercises that are automatically evaluated. Enki uses several gamification strategies to engage learners, including generic gamifications services provided by Odin and the sequencing of educational resources. The course content (videos, PDFs, programming exercises) is progressively disclosed to the learner as he successfully completes
exercises. This is similar to what happens in a game, where new levels are unlocked as the previous are completed, thus contributing to the sense of achievement.

CONCLUSION

The proposed book aims to share new approaches and methodologies for the use of gamification on the computer programming teaching-learning process. At the same time, it identifies new trends on this topic from pedagogical strategies to technological approaches. The proposed book could be used as a valuable resource for practitioners and as a reference for research scholars and computer science teachers and students pursuing computer science related subjects.

In short, this book clearly impacts the field and contributes with new trends to foster computer science education. In a world where technological advances appear at a dizzying pace all computer science educators must keep up with these changes and adapt teaching methodologies in order to enhance the study of computer programming and motivate students to achieve this complex, but exciting and emerging area.

Ricardo Alexandre Peixoto de Queirós
Polytechnic Institute of Porto, Portugal

Mário Teixeira Pinto
Polytechnic Institute of Porto, Portugal

REFERENCES


