Computer Science Education, 2013
Vol. 23, No. 2, 138–157, http://dx.doi.org/10.1080/08993408.2013.777236

Routledge
Taylor & Francis Group

Mobile game development: improving student engagement and motivation in introductory computing courses

Stan Kurkovsky*

Department of Computer Science, Central Connecticut State University, New Britain, CT 06050, USA

(*Received 23 October 2012; final version received 11 February 2013*)

Computer games have been accepted as an engaging and motivating tool in the computer science (CS) curriculum. However, designing and implementing a playable game is challenging, and is best done in advanced courses. Games for mobile devices, on the other hand, offer the advantage of being simpler and, thus, easier to program for lower level students. Learning context of mobile game development can be used to reinforce many core programming topics, such as loops, classes, and arrays. Furthermore, it can also be used to expose students in introductory computing courses to a wide range of advanced topics in order to illustrate that CS can be much more than coding. This paper describes the author's experience with using mobile game development projects in CS I and II, how these projects were integrated into existing courses at several universities, and the lessons learned from this experience.

Keywords: games; mobile computing; mobile game development; engagement; motivation

Introduction

Ongoing enrollment crisis in computer science (CS) stems, among other factors, from the decreased appeal of CS as an academic discipline or a career choice. Furthermore, many CS students end up changing their majors after taking CS I or CS II, because they find the course material irrelevant to practical applications. If we are to reverse this trend, it is increasingly important for the CS curriculum to stay relevant to today's reality and engage students by making a strong connection between computing and their everyday lives. Introducing students to mobile applications and game development may help students to better relate to the course material and make stronger connections to real-world applications and gadgets they see and use every day.

Using computer games as a motivational and instructional tool in CS has become a popular subject. Introduction of game programming and game

*Email: kurkovsky@ccsu.edu

development courses or even entire degree programs has also received a broad attention. One of the main barriers to a wider and successful adoption of computer games as a learning context has been developing course software and curriculum that would balance the motivational and educational aspects. At the same time, there are very few reports from educators about using mobile game development to increase student engagement and interest in pursuing a CS degree. This is surprising because mobile game programming is by far less complex than traditional game development due to the smaller scale of mobile games, simpler graphics, and a number of other factors discussed in this article. Therefore, it may often be easier to adopt mobile game development in the CS curriculum than traditional game development.

The main goal of the effort described here is to improve student success and satisfaction in introductory CS courses. The author and his collaborators produced a set of learning modules consisting of laboratory projects and instructional materials that use mobile game development early in the CS curriculum as a motivational learning context to reinforce the learning of fundamental principles of programming, e.g. classes and inheritance. These learning modules also serve as a platform to introduce students to a range of advanced CS topics, such as artificial intelligence, databases, and networking. By demonstrating these and other non-programming and diverse aspects of the discipline to the students, this approach may help dissolve a widely popular misconception that "CS is all about coding." Research literature and our own experience demonstrate that most CS students seem to be very interested in computer game development, and introducing students to this topic early in the curriculum could serve as a good tool to increase student retention. More broadly, current research literature indicates that students perform better when they find their course material relevant and motivating.

Making the case for mobile game development

Computer games gained a wide acceptance as an engaging and motivating tool in the CS curriculum (e.g. Bayliss & Strout, 2006; Leutenegger & Edgington, 2007; Sweedyk, deLaet, Slattery, & Kuffner, 2005). However, designing and developing a playable game is a very challenging task and is best implemented in advanced courses where students already have sufficient experience in software development and exposure to other CS topics. Games for mobile devices offer the advantage of being simpler by nature and, therefore, easier to program. This makes it more feasible for lower level students to develop playable games as a part of their classroom experience.

Mobile applications are often easy for students to relate to, because mobile technology plays an increasingly important role in the lives of today's students. For many of them, their mobile phone is replacing a desktop computer as their primary computing device. Mobile applications and

games offer instant gratification in the sense that students can download them to their mobile phones almost immediately and show them off to their friends.

In our approach, we strive to make connections to advanced CS topics as early in the curriculum as possible. Many studies, including Beaubouef and Mason (2005), Biggers, Brauer, and Yilmaz (2008), Zyda, Thukral, and Ferrans (2008), indicate that students often get disillusioned in CS, because they see too many irrelevant or contrived examples that are boring or have very little to do with real-world applications. There is also a certain percentage of students who are interested in other areas of CS that emphasize design or theory over coding. By exposing students to a wide range of advanced CS topics early in their academic career, our approach aims to show students that CS is much more than coding and that there are many areas in which programming plays a supplementary role.

Why mobile game development suits introductory CS courses

Many CS programs experience high enrollments in their first introductory course, but witness a precipitous drop in the number of students who stay in the program and continue on with subsequent courses. While some students may be mismatched with the program, many others decide not to continue with studies in CS because they feel that the course material lacks connection with the real world. Some students feel that the problems being solved in introductory courses are too contrived and lack any relevance to real life; that their future work will involve nothing but coding; and that there will be little room for creativity and personal growth in their future careers if they choose to continue studying CS. Additionally, a CS I course is often the only CS course that students from other majors are required to take as part of their graduation requirements. Exposing these students to various higher level CS concepts through hands-on projects could attract them to this discipline and has the possibility to encourage these students to consider a second major or a minor in CS.

Computer games have been successfully applied to improve recruitment and retention (Barnes, Richter, Powell, Chaffin, & Godwin, 2007); *they have been shown to be a successful learning tool by leveraging students' enthusiasm towards computer games and their social relevance* (Morrison & Preston, 2009). Current students, more than at any time before, need a classroom experience that they can relate to, that is creative and challenging, and makes a difference in preparing for their careers. To address this challenge, the design, development, and implementation of computer games has been used in CS classrooms in many ways: as a general engagement tool (Bayliss & Strout, 2006; Clua et al., 2006); as motivators and examples in introductory programming courses (Barnes et al., 2007; Leutenegger & Edgington, 2007; Sweedyk et al., 2005); as capstone courses (Barnes et al., 2007;

Parberry, Roden, & Kazemzadeh, 2005; Wallace, Russell, & Markov, 2008); in interdisciplinary programs connecting CS with arts and humanities (Diefenbach, 2011; Murray, Bogost, Mateas, & Nitsche, 2006; Sweedyk et al., 2005, Wolz & Pulimood, 2007); as a well-fitting instrument to teach computer graphics (Sung, Shirley, & Rosenberg, 2007; Sweedyk et al., 2005), artificial intelligence (Wallace et al., 2008), project management (Wolz & Pulimood, 2007), and software engineering (Claypool & Claypool, 2005; Gestwicki, 2007; Lin & Sun, 2011); or as a concentration within existing CS programs (Coleman, Krembs, Labouseur, & Weir, 2005; Zhang and Jones, 2012; Zyda et al., 2008). Successes of several very large projects [including Alice (Cooper, Dann, & Pausch, 2003) and Scratch (Peppler & Kafai, 2007)] aimed at the broad application of computer game development as a learning framework have been widely publicized. A number of surveys indicate that the majority of students prefer programming projects and assignments that one way or another involves computer games (Cliburn & Miller, 2008). Furthermore, many students say that their choice of CS as a major was sparked by computer games (Parberry et al., 2005; Wallace et al., 2008).

The importance of context in CS education has long been recognized (Forte & Guzdial, 2004; Guzdial, 2009). Using a context allows educators to focus on a subject that is important and familiar to students while learning new concepts; it provides a motivational tool and offers a wider playing field for experimentation and engaging students in the educational process. As the data from an IBM study, 2008 suggests that current students might be much more interested in using their mobile devices than PCs, the role of such a context becomes even more important. *Using mobile game development as a motivational learning context has a strong potential to improve student success in introductory CS courses and increase student motivation to stay in the major*. ACM computing curriculum recognizes the role of mobile computing in today's world as well as in the CS discipline. According to ACM computing curriculum, building interfaces for mobile systems is among many areas of practical capabilities in which prospective graduates should have experience. At the same time, game development has been successfully applied as a CS learning context at many levels. Young people have a special affinity to mobile gadgets; they would find it difficult to live a day without using their mobile phones to talk, check email, send text messages, and play games. Using mobile ubiquitous devices in the educational process will add a social dimension to the success of this learning context – students will be able to see a connection between the technical material and their everyday lives. In fact, a number of recent publications described a number of novel approaches to using mobile devices as a learning context; for example, in the TouchDevelop project described by Tillmann et al. (2012), mobile applications are developed directly on mobile devices. CS educators realize the importance of not only offering individual courses in

mobile computing, but also incorporating the elements of mobile application development throughout the curriculum to leverage the engagement and motivation factor of this learning context (Fenwick et al., 2011).

Creating engaging and feature-rich games requires a skillful integration of a wide range of techniques from many areas of CS and, therefore, computer games can be used at different points in the CS curriculum. Most importantly, computer games can be successfully used to present the richness of CS as an academic discipline to students early in their studies. Student attrition is highest at that point, as many existing CS students choose to change their major after taking a few introductory CS courses because they perceive course material as dry and having no relevance to real-world applications (Beaubouef & Mason, 2005; Biggers et al., 2008). Many students indicate that they have to go through a substantial number of CS courses until they are offered projects that are interesting and that they perceive as relevant to their future careers. Some students are also put off by what they perceive as an emphasis on coding in CS I and CS II. Even in those programs that offer a breadth-first CS 0 course providing an overview of many topics in CS, students demand more hands-on experiences that would allow them to apply theoretical concepts acquired in early CS courses to a more practical context (Beheshti et al., 2008; Carter, 2006).

In contrast, our approach would provide enough flexibility for varying levels of student experience and background, with implementation of concepts being central to the learning process. It is very important to provide students with a hands-on experience where they can apply theoretical concepts in practice. It is one thing to tell students about computational complexity of an algorithm, but quite another to have the students apply the same algorithm to problems of different size and see firsthand a significant difference in the execution time on a mobile device with a relatively slow processor. It is one thing to tell a student about the properties of an algorithm exploring a game search space, but quite another to have students observe their implementation run out of the small amount of memory on a mobile device. Through such experiences, students gain a much deeper appreciation of the design issues at stake. *Using mobile devices as a learning context in introductory CS courses aims to provide a simple and elegant means to motivate students and communicate the diversity and power of many advanced CS areas in a manner that engages students in experiential education.* We believe that good educational experiences at the introductory level are necessary to excite students' curiosity and motivate further, deeper inquiry.

Furthermore, research has also shown that more participatory learning methods such as those used in mobile game development can level the playing field for different types of students. For example, CS and engineering have historically been less accessible for female and underrepresented minority students, and as a result, these students are underrepresented in

most CS and engineering departments in this country. A shift to a learning environment which values interactivity, cooperation, and collaboration can result in female and minority students feeling more comfortable and, by extension, can lead to greater persistence and success. In fact, several studies have shown that *these learning methods more closely match the learning styles and preferences of women and minorities* (Baxter-Magolda, 1992; Belenky, Clinchy, Goldberger, & Tarule, 1986; Lundeberg & Diemert, 1995; Wolz, Barnes, Bayliss, & Cromack, 2009).

So far, compared to integrating traditional computer game development into CS curricula, the idea of integrating mobile games has received relatively little attention. Mobile game development has been successfully integrated into graduate degrees (Zyda et al., 2008); they have been found helpful as a motivating tool to increase student interest in CS and other STEM disciplines (Metcalf, Milrad, Cheek, Raasch, & Hamilton, 2008). Mobile devices have been integrated into CS curriculum in different ways, including teaching computer literacy (Mahmoud & Dyer, 2007) and introductory programming (Mahmoud & Dyer, 2008; Tillmann et al., 2012). Current research indicates that *mobile devices and mobile games are a valuable teaching and learning tool* (Chao, 2006), especially in the context of attracting schoolchildren to computing (Sharples, 2000), improving engagement and motivation in secondary education (Huizenga, Admiraal, Akkerman, & ten Dam, 2009), and as a powerful tool used in augmented learning (Klopfer, 2008). Mobile game development also gives students the benefit of instant gratification – they can quickly build a working graphical application (that is more satisfying than calculating an average salary), and play the resulting game on their own mobile device.

Many critics of using computer games in the curriculum argue that they promote misogynistic and violent content (Haller et al., 2008), and that women are generally less interested in games than men (Leutenegger & Edgington, 2007). Indeed, many console and PC games are marketed to young males (Natale, 2002). However, according to Entertainment Software Association (2008), 40% of all game players in the USA were women. Furthermore, according to several surveys, *women enjoy games-oriented introductory CS courses more than men* (Haller et al., 2008; Leutenegger & Edgington, 2007), *their average grades in CS courses that use game design are often higher than men's* (Chamillard, 2006), *and incorporating games into CS curriculum has been shown to increase percentage of females enrolled in these programs* (Wolz et al., 2009).

Casual games (Koivisto, 2006; Kurkovsky, 2009) are the most popular kind of mobile games, mostly due to the patterns of mobile phone usage. Casual games are characterized by extremely simple gameplay (e.g. puzzles or card games) and generally appeal to "casual consumers" who do not regard themselves as "gamers." Casual games are typically played in short bursts: during work/class breaks, while waiting in line, on public transporta-

tion, etc. Their rules are simple, and, unlike many console games, they do not require a long-term time commitment or any special skills. Most importantly, *current studies indicate that mobile gamer demographics are split fairly equally between males and females* (Entertainment Software Association, 2008; Koivisto, 2006).

One of the challenges in adopting game-related coursework into CS curriculum is to balance the educational and motivational aspects by choosing the right mix of the material that students find interesting and the material that would help them grow academically (Repenning & Ioannidou, 2008). As some critics of using games in CS classrooms noted, we should not focus on games simply because students like games (Haller et al., 2008). Another substantial barrier to adopting game development into the curriculum is the inherent complexity of developing a playable and well-designed game (Burd, Goulden, Ladd, Rogers, & Stewart, 2007). Developing playable games is a complex task, but due to its complexity it also opens an opportunity to introduce students to many other advanced areas in CS. We believe that mobile game development addresses both of these challenges. *Due to relative simplicity of many mobile games, a playable game can be developed within the timeframe of one semester by many CS students with reasonable Java programming skills*. Additionally, mobile games can also be successfully used to broaden the horizons of students in introductory CS courses by exposing them to a variety of advanced topics early in the curriculum.

Finally, experience in mobile applications and game development may also serve as an additional motivational factor, because *it is a skill in high demand by the industry* (Brandel, 2007). According to Gartner Research, the revenue in mobile gaming industry is expected to reach $4.5 billion in 2008, which represents a 16.1% growth compared to 2007. Gartner projects that the industry will grow at the rate of at least 10% per year through 2011. In the USA, 36% of surveyed heads of households reported that they play games on their mobile devices, up from 20% in 2002 (Entertainment Software Association, 2008). The target mobile phone market is very accessible and extremely large with estimates of over two billion phones in use worldwide.

Mobile game development learning modules

With the support from two NSF TUES/CCLI grants, we created a set of several curricular modules consisting of laboratory projects and related instructional materials for introductory computing courses in Java. Every module focuses on a single mobile game and aims to introduce students to a single advanced topic, e.g. artificial intelligence, algorithms, computer networking, databases, software engineering computer security, or human–

computer interaction. Each module is also designed to offer students a chance to practice at least one fundamental topic, such as classes, arrays, or loops.

Java has been purposefully selected as the development platform for this project due to several reasons. Today, Java is the dominant instructional language used by instructors in higher education, as well as in high schools and by the advanced placement exam. Additionally, a specialized version of the language, Java Micro Edition (Java ME), provides all the necessary tools for building mobile applications that include a basic class library supporting animation. There are several free software development environments that support Java ME on Windows, Linux, and Mac OS (most notably, Eclipse and NetBeans). As a result, there will be a minimal number of technical obstacles should an instructor or an academic program decide to adopt the instructional materials developed in this project. Although primarily designed for introductory courses, the source code of the software foundation supporting the learning modules is fully available to students and instructors, and could be also used in upper level courses for projects in mobile computing and as building blocks for more advanced mobile game development. Thus, our Java-based materials will provide a platform usable throughout the curriculum – from high school and freshmen courses to advanced courses – without requiring learning a new programming language.

Each of the learning modules focuses on a particular advanced CS topic and includes a set of Java classes implementing a feature-rich game with an emphasis on that topic. Each learning module focuses on a different game, which includes a range of familiar puzzles and arcade-style games. In order to focus on the laboratory project at hand, students do not need to know all implementation details of the game. Instead, each module exposes one or two Java classes that need to be modified or extended in order to complete the project, which makes it easy to integrate our materials into the existing curriculum. We follow Youngblood's (2007) "design principles for game segments" to create our learning modules: the areas where students need to incorporate their code is clearly defined; students are able to easily change the cosmetics and/or behavior of the game so they can customize it and take ownership; learning modules are designed for competition or diversity to help show off the students' work in class and incorporating it back into the lessons; non-solution-revealing examples of the desired behavior are provided; provided source code and other materials are sufficient to clearly understand the development task and nature of the solution while avoiding tedious programming; and, learning modules encourage code exploration and game alteration by making the experience interesting and fun.

The following section illustrates the contents of a typical learning module.

Sample learning module: multi-player games and computer networking

Learning objectives: (a) Explain the basic concepts of computer networking; (b) Implement a simple two-way communication protocol; (c) Apply object-oriented programming techniques in a practical context.

Learning outcomes: Upon a successful completion of this project the students will be able to: (a) describe the basic concepts of computer networking; (b) implement a simple two-way communication protocol; (c) develop software that incorporates inheritance and class hierarchies, interface classes, exceptions, and event-driven graphical user interfaces.

Problem statement: The game Battleship is a guessing game played by two opponents, which has been played since early 1900s throughout the world as a pencil and paper game. In a typical game, each opponent uses a 10×10 board to place a number of ships of different configurations; each player has the same number and configuration of the ships, but is not allowed to see the opponent's board. Upon positioning of the ships, the game is played as a series of turns where each opponent announces the target square to be shot at. The game proceeds until all of the ships of one opponent have been sunk. The objective of this project is to complete an implementation of this game for mobile devices so that two opponents can against each other over a wireless network connection.

Programming project: A mobile implementation of this game is provided where opponents play the game on their mobile phones. A basic infrastructure for sending and receiving messages between the two devices is in place. Students are required to complete the project by designing and implementing a communication protocol for messages exchanged by the devices to inform each other about the game turns (the target square on the board) and their outcomes (whether there is a hit or miss). To successfully complete this project, students do not need to know the details of communication infrastructure provided in this project (although they do have access to that code). The code provided to the students can also handle all details relevant to maintaining the state of the game between the two devices (e.g. who's turn is next). This project is ideally suited for students working in teams of two, who first can develop and test out the protocol of a simple message/response exchange using a paper-based game, then work out the data format of the messages and responses, how they will be analyzed, and what decisions will need to be made by the game logic.

Demonstration game and sample solutions: Instructors are provided with a complete working implementation of the game. Running this implementation would demonstrate to the students how the game is intended to work without revealing the specifics of the project's solution. Students are provided with a clear interface exposing the methods for message passing and making game-play decisions so that they can focus on the learning objective and not on the inner workings of the rest of the implementation.

However, since the source code with low-level technical details of message passing using sockets will be available, highly motivated students will have an avenue for expanded learning opportunities in computer networking. Sample solution available to the instructors would allow them to assess the amount of time and effort needed by their students to complete the project, and help them in answering student questions.

Lecture notes: Instructors are provided with a set of lecture notes introducing computer networking at a level accessible to college freshmen. These materials emphasize the main concepts of computer communications (such as types of networks, protocols, networking devices, latency, etc.), and use examples from the Battleship game and other existing computer games to illustrate these concepts. Lecture notes contain a detailed description of the game and outline the strategies for developing a successful communication protocol between the opponents' devices. These notes also contain a range of additional introductory material that goes beyond the scope of the topics related to the Battleship game (such as network protocol layers, network management, and network security) in case if the instructor should decide to include them in preproject or postproject lectures and discussions.

Creative exploration: The game is implemented in such a way that all graphical elements (background, tiles of the game board, ship elements, etc.) and colors used in the game are be easily customizable. Students are also be provided with a substantial library of artwork that would be suitable for replacing some or all of these elements (e.g. sizes of the artwork elements in the library match those used in the game). Students will be encouraged to customize their implementation with the artwork from the library or create their own. Furthermore, by doing so, students will be able to substantially modify game semantics without making changes to the underlying logic. For example, by using an image of a green lawn as the background and images of different flowers as ship elements, students could create a game of "crazy gardeners," in which two gardeners will play against each other trying to destroy each other's flowerbeds.

Taking a step further: Students and instructors wishing to explore this project further are offered a number of directions to build upon this project. For example, students may choose to implement a capability to chat between players by adding extra features to the communication protocol and extending the game logic. Additionally, a supplement to this project provides a high-level interface to the socket-based message exchange and game state maintenance features of the project. This supplement could enable students to build multiplayer turn-based games of their own.

Topics for mobile game development learning modules

This section provides brief summaries of the projects and underlying games.

Artificial intelligence. Using an implementation of the game Connect Four, students are provided with one or more options to experiment with the game logic, implement several different heuristics of the search algorithm, and witness the results of their modifications firsthand. Changes made by the students to the game code will likely result in the varying quality of game turns made by the program and the speed of making these decisions. This module provides the instructor with an opportunity to further discuss turn-based games, search algorithms, heuristics, and the history of computer programs for playing chess.

Database management. Students are provided with a complete implementation of an arcade-style game that favors speed and reaction (along the lines of Super Mario Bros). An exhaustive set of game variables describing the game state (score, items collected, current level, time elapsed, etc.) is exposed. A mechanism to store and retrieve persistent data on the mobile device is also provided. Students are required to implement the logic for saving the current game state and loading one of the previously saved game states from the persistent storage. Students need to decide which of all possible variables must be saved and how their values need to be restored. Materials of this module could serve as a foundation to discuss such database-related topics as data organization and data models, indexing, and transactions.

Software engineering. This is one of the simpler modules, in which students are provided with a complete implementation of a well-known and simple game, e.g. Frogger. The code responsible for the main logic of the game is intentionally left with many opportunities of refactoring, e.g. there are several instances of very similar groups of statements with similar semantics, identifiers with poorly chosen names, etc. Students are asked to optimize the code by eliminating these redundancies, and thus improving the readability and reusability of their code. This module provides instructors with an opportunity to discuss other essential topics of software engineering appropriate for students in introductory courses, such as software system design, software quality, testing, and teamwork.

Human–computer interaction. In this module, students are provided with a complete implementation of a classical game (e.g. Space Invaders) with poorly designed graphical elements (characters and backgrounds), inconvenient controls, and sound effects with mismatched volumes. Students are asked to try playing the game, discuss their experiences, and suggest possible improvements to make the game more enjoyable without altering the main logic of the game. Students are provided with a library of alternate artwork and sound effects. Using these materials, students can modify the game code to possibly reassign different buttons to control the game, load different background images, sprites, and sound effects, and programmatically adjust the volume of each individual sound effect, if needed. Completion of the project provides an opportunity to discuss such topics as

software usability, interface design and assessment, and interaction techniques.

Computer security. In this module, students are provided with a complete networked Tic–Tac–Toe game. The game is played against the computer. Since this is a wireless application the case can be made that the user's move must be authenticated by the server. The students are therefore required to implement the code to digitally sign each input move of the user. A hash function and a key can be used to digitally sign the input move before transmitting it to the server. This gives the instructors an opportunity to discuss basic computer security concepts such as authenticity of digital messages and the application of cryptographic techniques.

Computer systems. In this module, students are provided with the basic code for a Sudoku game. The game will be able to create a 6×6 grid. The students are required to modify the game to increase the size of the grid. Students are encouraged to determine how large the grid can become and what are the limitations as the size of the grid gets larger. Instructors can use this example to explain the constraints in mobile architectures – low processing power, smaller memory. In addition, the instructor can also talk about the memory hierarchy and how that affects the response time for an application.

Algorithms. In this module, students are provided with a mobile implementation of Text Twister. In this game, the user is given a six-letter word with the letters scrambled. In order to get to the next level of the game, the user must unscramble the six-letter word. Users can also gain bonus points by creating three-, four-, and five-letter words with the letters provided. Only five six-letter words will be included in the implementation. In order for the game to be meaningful, it must be able to challenge the user with more than just five words. Students need to extend this game to include 15 six-letter words. They are required to analyze various search algorithms and determine the best algorithm for a mobile device with limited memory and computational power. Instructors can use this example to motivate the idea of complexity of algorithms and how to determine when an algorithm is optimal.

Assessment of mobile game development learning modules

Several of the developed modules have been tested at Central Connecticut State University (CCSU) and Rose-Hulman Institute of Technology (RHIT). In testing these modules, we employed three assessment strategies: (1) student course grades and course completion, (2) the Classroom Survey of Student Engagement (CLASSE) administered at the end of the term, and (3) the Motivated Strategies for Learning Questionnaire (MSLQ) administered at the beginning and end of the academic term. The modules were used relatively late in the courses, during the last few weeks of the term.

In the first year of the project period, a total of 161 students enrolled across eight sections of introductory computing courses at CCSU and RHIT that were in the scope of this study, although the total number of students who completed each assessment varied. A total of 84 students were enrolled in a test section in which one or more curricular modules using mobile game development were used to convey basic concepts, and 77 students were enrolled in paired control sections taught by the same faculty member in which mobile game development was not used as a curricular feature. Not all students completed the course, however, nor did all of those students who completed courses participate in the both administrations of MSLQ and the one administration of CLASSE. The course population varied between the two institutions: RHIT course was open only to computing and engineering majors, while CCSU course was open to students of any major, as long as they met the course prerequisites.

According to the assessment report (Hosch, 2011), results from the first year of the assessment of the project yielded findings in various areas of interest. We highlight a few of these areas.

Student persistence

Persistence rates appear independent of administration of mobile game development curricular modules but were significantly different between the two institutions, with students at CCSU persisting at lower rates. Because mobile game development curricular modules were administered toward the ends of academic terms, there would have been limited opportunity for them to affect student persistence. The differences between universities in persistence rates very likely reflect the selectivity of the institutions, composition of the class population, and academic inputs of students taking the courses, with students at the less selective CCSU exhibiting lower persistence rates and lower levels of academic performances.

Course grades

Student course grades were independent of the use of the curricular modules. This suggests that the mobile game development modules as administered at both institutions did not affect student performance on the assessments used by these instructors.

Direct measurement of student learning during the first year of the project relied upon course exams and course grades in these introductory courses, and no meaningful differences were observed between test and control groups. Differences between students at the two institutions were pronounced, with CCSU students in all earning an average course grade of 2.56, more than a full letter grade below students in all sections at RHIT who earned an average course grade of 3.63. Further, no RHIT

students earned course grades below C, and all finished the course, compared with 22 CCSU students (29%) who earned course grades below C (2.0) or withdrew from the course (six of these students earned grades of C- which is considered passing at CCSU with 1.7 grade points, but RHIT does not use the minus designation for letter grades). Many students did not complete one or more assessments and the performance differential between test and control groups is not meaningfully different.

Student ratings of instructor and course quality

Patterns of student ratings of instructor quality and course quality varied for institution. Among RHIT students, ratings of instructor quality and course quality as measured on CLASSE at the end of the course were lower for test sections than for control sections at statistically significant levels. No significant differences were observed among test and control sections at CCSU. Perceptions of course quality exhibited similar patterns, although CCSU results for the test group tilted slightly more positive, albeit at a level that did not rise to statistical significance.

Absent other pronounced differences between test and control groups at RHIT, it seems plausible that the implementation of the mobile game development modules in test sections prompted this difference in the perception of effective teaching. Causes for this may have stemmed from the integration of a new and perhaps unusual curricular feature into a seamless course based on other pedagogies. It is possible that additional course planning, increased familiarity with the game development modules, or modifications to instructor materials could prompt more even ratings. It is also possible that some teaching or learning styles may better lend themselves to use of the mobile game development modules than others. CCSU results may be more dispersed across this scale than the RHIT results because of heterogeneity among the student populations, although all of these explanations remain speculative.

Student engagement with course

Students in test groups reported differences in terms of course engagement in areas of note taking, making connections across courses, course difficulty, and interest in course material. Some of these differences suggested mobile game development modules were associated with positive effects, while in other areas, for RHIT students only, findings suggested these modules may have had a negative effects in the areas of instructor comfort and communication as well as in interest in course material.

Note taking

Students in test sections were more likely to take notes than those in control sections. While the number of respondents at each institution was too low on this assessment instrument to yield statistically significant results at the campus level, this difference between test and control groups on this item was 24% among CCSU students and 12% among RHIT students. Reasons for this difference are not immediately clear, but they may involve the need to pay closer attention during demonstrations of mobile game development curricular units or to write down instructions during these classes.

Putting together ideas or concepts from different courses

Students in test groups at both institutions reported putting together ideas or concepts from different courses when completing assignments or during class discussions more frequently than did students in control groups. No students in either test or control groups at CCSU reported putting together ideas or concepts from different courses when completing assignments or during class discussion, but 26% of students in the test group reported making such connections 3–5 times compared to 13% of those in the control group. Students at RHIT made such connections much more regularly, with 59% of students in the test group reporting putting together ideas or concepts from different courses when completing assignments or during class discussion 3 to 5 or 5 or more times, compared to 47% of students in the control group who made such connections. It is possible again that RHIT students, who enter the institution with on average stronger academic backgrounds, are accustomed to such habits more so than students at CCSU.

Perceptions of course difficulty

While no differences between test and control groups at RHIT were observed in terms of how students perceived the difficulty of course material, among CCSU students, those in test sections reported that course material was less difficult than did students in control sections. While 35% of students in the control group at CCSU reported the course material was very difficult, just 11% of students in the test group reported a similar level of difficulty. It is important to note, however that respondents in the control groups were distributed bi-modally with a large cluster (39%) reporting the course material was just somewhat difficult, while 17% reported it was difficult. By contrast, less than 10% of RHIT students in either the test group or the control group reported the course material was very difficult. Again this difference may be related to the higher selectivity of RHIT as an institution and students who may be better prepared to negotiate college-level work in a CS course.

Interest in course material

RHIT students in the test group reported lower levels of interest in course material than did students in the control group, although these differences were significant only at a borderline level. These proportions were reversed among CCSU students, but the lower number of respondents to CLASSE at CCSU did not allow these differences to rise to a generally accepted level of statistical significance. While 78% of RHIT students in the control group reported they were very interested in course material at the end of the term, only 59% of students in the test group reported the same high levels of interest. By contrast, among students at CCSU, 30% of students in the control group reported they were very interested in course material, while 52% of students in the test group reported this same high level of interest. Of further note, 18% of students in the test group and 26% of students in the control group at CCSU reported that they were either uninterested or very uninterested in course material, while just one student among the RHIT test and control groups (about 1.5% of respondents) reported being uninterested in course material, suggesting again that RHIT students may have been predisposed to be more engaged in introductory CS concepts that students at CCSU.

Taken as a whole, findings about student engagement in the course present a mixed bag of results that suggest limited positive effects of using the mobile game development curricular modules in increased note taking, making connections across courses, and for the students at CCSU, whose academic preparation for college was not as strong as those at RHIT, perceptions of course difficulty decreased. Use of the mobile game development curricular modules with the better prepared RHIT students, however, exhibited some negative effects in the area of instructor comfort and communication as well as student interest in the material. It is possible that these negative findings were a result of students who expected a more traditional classroom experience with these concepts or that the curricular modules did not fit seamlessly into the rest of the course as taught by RHIT instructors.

Student motivation

In general, motivation among students in CCSU CS classes declined, but this decline was less pronounced in test sections than control sections. By contrast, RHIT student motivation stayed level or increased slightly over the course of the course of the term for both the test group and the control group, with motivation gains among the control group slightly outpacing those in the test group.

Student motivation was measured using the short (31-question) version of MSLQ; this instrument was administered in the first two weeks of the term and again at the end of the term. Results were analyzed only for

students who participated in both administrations of the instrument. In general, student motivation among CCSU students declined, but this decline was less pronounced in test sections than control sections. At RHIT, however, student motivation stayed level or increased slightly during the term for both the test group and the control group, with motivation gains among the control group slightly outpacing those in the test group. These changes on MSLQ indices amounted to differences of as much as half a standard deviation between test and control groups, but they did not rise to the level of statistical significance.

Summary and future work

Preliminary results described here suggest that in each of the areas highlighted above, the test groups did at least as good as the control groups. In fact, students in the test groups showed higher levels of engagement with the course material in which the modules were presented and their motivation did not decline as much as in the control groups. We will continue to refine the modules and adjust the points in the courses in which the mobile game development projects are offered to the students. Also, giving students more time to complete the work related to these curricular modules and to apply what they learn to other topics may produce more positive results.

As we pointed out in our current evaluation, there is still a substantial room for improvement in this project. We believe that incorporating mobile game development projects into the introductory computing curriculum will promote teaching and learning, help improve student experience in the critical introductory courses in computing when student attrition is at its highest, and ultimately will help train outstanding computer scientists. Currently, we are finalizing our mobile game development modules to address the concerns raised as the result of the ongoing project assessment. We shared one of the modules at the 2011 ACM Symposium on Computer Science Education (SIGCSE 2011) and at several conferences of the Consortium for Computing Sciences in Colleges (CCSC) (Defoe, Kurkovsky & Graetz, 2011) and it was very well received.

Acknowledgments

The author would like to thank Dr Archana Chidanandan who was instrumental in getting this project started and for her contribution to the development of the learning modules at RHIT, Dr Delvin Defoe for his continued involvement with maintaining this project at RHIT, and Dr Braden Hosch for his invaluable contribution in the evaluation and assessment of this project. This material is based upon work supported by the National Science Foundation under Grants DUE-0630953 and DUE-1060352.

References

Barnes, T., Richter, H., Powell, E., Chaffin, A., & Godwin, A. (2007, June). Game2Learn: Building CS1 learning games for retention. *SIGCSE Bulletin, 39*, 121–125.

Baxter-Magolda, M. D. (1992). *Knowing and reasoning in college: Gender-related patterns in students intellectual development*. San Francisco, CA: Jossey-Bass.

Bayliss, J. D., & Strout, S. (2006, March). Games as a "flavor" of CS1. *SIGCSE Bulletin, 38*, 500–504.

Beaubouef, T., & Mason, J. (2005, June). Why the high attrition rate for computer science students: Some thoughts and observations. *SIGCSE Bulletin, 37*, 103–106.

Beheshti, M., Alo, R. A., Fernandez, J., Gates, A. Q., Ranjan, D., Boadi, A., … Barker, L. (2008). *CS0 course implementation in computer science*. In Proceedings of the 2008 frontiers in education conference. Saratoga Springs, NY, pp. F4A-7–F4A-9.

Belenky, M. F., Clinchy, B. M., Goldberger, N., & Tarule, J. M. (1986). *Women's ways of knowing: The development of self, voice, and mind*. New York, NY: Basic Books.

Biggers, M., Brauer, A., & Yilmaz, T. (2008). S*tudent perceptions of computer science: A retention study comparing graduating seniors vs. CS leavers*. In Proceedings of the 39th SIGCSE technical symposium on computer science education. Portland, Oregon, pp. 402–406.

Brandel, M. (2007). 12 IT skills employers cannot say no to. ComputerWorld, Jul. 2007.

Burd, B., Goulden, J., Ladd, B., Rogers, M., & Stewart, K. (2007, March). Computer games in the classroom, or, how to get perfect attendance, even at 8 AM. *SIGCSE Bulletin, 39*, 496–496.

Carter, L. (2006). *Why students with an apparent aptitude for computer science don't choose to major in computer science*. In Proceedings of the 37th SIGCSE technical symposium on computer science education. Houston, TX, pp. 27–31.

Chamillard, A. T. (2006, March). Introductory game creation: No programming required. *SIGCSE Bulletin, 38*, 515–519.

Chao, C. (2006). *An investigation of learning style differences and attitudes toward digital game-based learning among mobile users*. In Proceedings of the 4th IEEE international workshop on wireless, mobile and ubiquitous technology in education (WMTE), 16–17 November. Washington, DC: IEEE CS, pp. 29–31.

Claypool, K., & Claypool, M. (2005, September). Teaching software engineering through game design. *SIGCSE Bulletin, 37*, 123–127.

Cliburn, D. C., & Miller, S. (2008, February). Games, stories, or something more traditional: The types of assignments college students prefer. *SIGCSE Bulletin, 40*, 138–142.

Clua, E., Feijó, B., Rocca, J., Schwartz, J., das Graças, M., Perlin, K., … Barnes, T. (2006). *Game and interactivity in computer science education*. In ACM SIGGRAPH 2006 Educators Program (Boston, MA, 30 July–3 August), SIGGRAPH '06, New York, NY: ACM.

Coleman, R., Krembs, M., Labouseur, A., & Weir, J. (2005, February). Game design & programming concentration within the computer science curriculum. *SIGCSE Bulletin, 37*, 545–550.

Cooper, S., Dann, W., & Pausch, R. (2003). *Teaching objects-first in introductory computer science*. In Proceedings of the 34th SIGCSE technical symposium on computer science education, Reno, Navada, pp. 191–195.

Defoe, D., Kurkovsky, S., & Graetz, E. (2011, May). Short mobile game development projects for introductory CS courses: Conference workshop. *Journal of Computing Sciences in Colleges, 26*, 141–143.

Diefenbach, P. J. (2011, May–June). Practical game design and development pedagogy, *Computer Graphics and Applications*. *IEEE, 31*, 84–88.

Entertainment Software Association. (2008). *Essential facts about the computer and video game industry*. Retrieved from http://www.theesa.com/facts/pdfs/ESA_EF_2008.pdf

Forte, A., & Guzdial, M. (2004). *Computers for communication, not calculation: Media as a motivation and context for learning*. In Proceedings 37th Hawaiian international conference on systems sciences, 5–8 January, Big Island, HI.

Gestwicki, P. V. (2007, March). Computer games as motivation for design patterns. *SIGCSE Bulletin, 39*, 233–237.

Guzdial, M. (2009, May). Teaching computing to everyone. *Communications of the ACM, 52*, 31–33.

Haller, S., Ladd, B., Leutenegger, S., Nordlinger, J., Paul, J., Walker, H., & Zander, C. (2008, February). Games: Good/evil. *SIGCSE Bulletin, 40*, 219–220.

Hosch, B. J. (2011, August). *Mid-point assessment report for NSF CCLI grant: Using mobile game development to improve student learning and satisfaction in introductory computer science course*s.

Huizenga, J., Admiraal, W., Akkerman, S., & ten Dam, G. (2009, August). Mobile game-based learning in secondary education: Engagement, motivation and learning in a mobile city game. *Journal of Computer Assisted Learning, 25*, 332–344.

IBM Study finds consumers prefer a mobile device over the PC. (2008). Retrieved from http://www-03.ibm.com/press/us/en/pressrelease/25737.wss

James B. F. Jr., Barry L. K., & Joel H. (2011). *Teaching mobile computing and developing software to support computer science education*. In Proceedings of the 42nd ACM technical symposium on computer science education (SIGCSE '11). ACM, New York, NY, USA, pp. 589–594.

Klopfer, E. (2008). *Augmented learning: Research and design of mobile educational games*. Cambridge, MA: MIT Press.

Koivisto, E. M. (2006). *Mobile games 2010*. In Proceedings of the 2006 international conference on game research and development, 4–6 December Perth, Australia. ACM International Conference Proceeding Series, Vol. 223, pp. 1–2.

Kurkovsky, S. 2009. *Engaging students through mobile game development*. Proc. SIGCSE, Mar. 2009, 44–48.

Leutenegger, S., & Edgington, J. (2007, March). A games first approach to teaching introductory programming. *SIGCSE Bulletin, 39*, 115–118.

Lin, J., & Sun, W. (2011). Engaging students in active learning by introducing game development into software engineering. *International Journal of Information Technology, 17*, 312–335.

Lundeberg, M. A., & Diemert, S. (1995). Influence of social interaction on cognition: Connected learning in science. *Journal of Higher Education, 66*, 312–335.

Mahmoud, Q. H. & Dyer, A. (2007). *Integrating BlackBerry wireless devices into computer programming and literacy courses*. In Proceedings of the 45th annual southeast regional conference, 23–24 March, Winston-Salem, NC. ACM-SE 45. ACM, New York, NY, pp. 495–500.

Mahmoud, Q. H., & Dyer, A. (2008, June). Mobile devices in an introductory programming course. *Computer, 41*, 106–108.

Metcalf, D., Milrad, M., Cheek, D., Raasch, S., & Hamilton, A. (2008). *My sports pulse: Increasing student interest in STEM disciplines through sports themes, games and mobile technologies*. In Proceedings of the 5th IEEE international conference on wireless, mobile, and ubiquitous technology in education, 23–26 March, Beijing, China, IEEE CS, Washington, DC, pp. 23–30.

Morrison, B., & Preston, J. (2009). *Engagement: Gaming throughout the curriculum*. Proc. SIGCSE, March 2009, 342–346.

Murray, J., Bogost, I., Mateas, M., & Nitsche, M. (2006, June). Game design education: Integrating computation and culture. *Computer, 39*, 43–51.

Natale, M. J. (2002, June). The effect of a male-oriented computer gaming culture on careers in the computer industry. *SIGCAS Computer Society, 32*, 24–31.

Parberry, I., Roden, T., & Kazemzadeh, M. B. (2005, February). Experience with an industry-driven capstone course on game programming: Extended abstract. *SIGCSE Bulletin, 37*, 91–95.

Peppler, K., & Kafai, Y. B. (2007). From supergoo to scratch: Exploring media creative production in an informal learning environment. *Journal on Learning, Media, and Technology, 32*, 149–166.

Repenning, A., & Ioannidou, A. (2008, February). Broadening participation through scalable game design. *SIGCSE Bulletin, 40*, 305–309.

Sharples, M. (2000, April). The design of personal mobile technologies for lifelong learning. *Computers & Education, 34*, 177–193.

Sung, K., Shirley, P., & Rosenberg, B. R. (2007, March). Experiencing aspects of games programming in an introductory computer graphics class. *SIGCSE Bulletin, 39*, 249–253.

Sweedyk, E., deLaet, M., Slattery, M. C., & Kuffner, J. (2005, February). Computer games and CS education: Why and how. *SIGCSE Bulletin, 37*, 256–257.

Tillmann, N., Moskal, M., de Halleux, J., Fahndrich, M., Bishop, J., Samuel, A., & Xie, T. (2012). *The future of teaching programming is on mobile devices*. In Proceedings of the 17th ACM annual conference on innovation and technology in computer science education (ITiCSE '12). ACM, New York, NY, pp. 156–161.

Wallace, S. A., Russell, I., & Markov, Z. (2008). *Integrating games and machine learning in the undergraduate computer science classroom*. In GDCSE 2008: Microsoft academic days conference on game development in computer science, Celebrity Century, Miami, FL, February 28–March 2.

Wolz, U., Barnes, T., Bayliss, J., & Cromack, J. (2009). *Girls do like playing and creating games*. Proc. SIGCSE, March. pp. 199–200.

Wolz, U., & Pulimood, S. M. (2007, March). An integrated approach to project management through classic CS III and video game development. *SIGCSE Bulletin, 39*, 322–326.

Youngblood, G. M. (2007, February). Engaging students in advanced computer science education using game segments. *Journal of Game Development, 2*, 33–45.

Zhang J., & Jones E. J. (2012). *Developing a gaming concentration in the computer science curriculum at an HBCU (abstract only)*. In Proceedings of the 43rd ACM technical symposium on computer science education (SIGCSE '12). ACM, New York, NY, p. 672.

Zyda, M. J., Thukral, D., & Ferrans, J. (2008). *Incorporating mobile games into a computer science game degree program*. In GDCSE 2008: Microsoft Academic Days Conference on Game Development in Computer Science, Celebrity Century, Miami, FL, February 28–March 2.