Designing an Intervention for Novice Programmers Based on Meaningful Gamification: an Expert Evaluation

Jenilyn AGAPITO* & Ma. Mercedes RODRIGO
Ateneo de Manila University, Philippines
*jen.agapito@gmail.com

Abstract: Gamification is defined as the addition of game-like elements and mechanics to non-game contexts to encourage certain desired behaviors. It is becoming a popular classroom intervention used in computer science instruction, including CS1, the first course computer science students take. It is being operationalized to enhance students' learning experience and achievement. However, existing studies have mostly implemented reward-based game elements which have resulted to contrasting behaviors among the students. Meaningful gamification, characterized as the use of game design elements to encourage users build internal motivation to behave in a certain way, is contended to be a more effective approach. The foundation of this concept is the 'Self-Determination Theory', which states that there are three components associated with intrinsic motivation: mastery, autonomy, and relatedness. This paper describes the first part of a research on the exploration of how a tool founded on meaningful gamification will affect the achievement and learning experience of novice programmers. It focuses on the design and implementation of a programming-based activity management system embedded with game design elements that map to the different components of the Self-Determination Theory. The elements employed are: feedback cycles, freedom to fail, and progress to support mastery; control to enable autonomy; and collaboration for relatedness. CS1 instructors invited for an expert evaluation generally agree that said elements are present in the system.

Keywords: gamification, novice programmers, CS1

1. Introduction

Computer science educators and researchers are concerned about student retention and attrition (Beaubouef & Mason, 2005; Dehnadi & Bornat, 2006; Hoda & Andreae, 2014; Robins, 2010; Wortman & Rheingans, 2007). CS1, the first course that computer science students take, has a failure rate of at least 30% across institutions (Bennedsen & Caspersen, 2007; Watson & Li, 2014). Students typically enroll in CS1 with little to no background in programming. They are introduced to a fundamental but difficult skill area. This experience shapes their impression of both the course and degree program. However, some researchers speculate that traditional teaching methodologies may not be the ideal choice for programming classes (Hoda & Andreae, 2014, Watson & Li, 2014). In particular, it may no longer be effective for students of this generation. Young people these days are attuned to the use of technologies to reinvent social living, communication, and learning, causing their divergent perspective on the dynamics and operations of the world, including the classroom (Klopfer et al., 2009).

An approach to instruction reinvention dubbed as gamification has recently become popular. Gamification is defined as adding game-like elements and mechanics to a learning process (Deterding et al., 2011; Glover, 2013). It intends to provide users with a more gameful experience and to encourage certain desired behaviors (Deterding et al., 2013) by adding motivational affordances in an environment.

1.1 Gamification in Education

Gamification has become evident in education and classroom design. It is shown to be an effective method to help at-risk students succeed (Ross, 2011). Several studies focus on gamifying Computer Science subjects (Gibbons, 2013; Ibáñez et al., 2014; Neve et al., 2014; Pirker et al., 2014), including CS1 (Harrington, 2016; Kumar & Khurana, 2012; Sprint & Cook, 2015). They experiment with incorporating game design elements such as badges/reward systems (Harrington, 2016; Ibáñez et al., 2014; Neve et al., 2014; Pirker et al., 2014; Sprint & Cook, 2015), leaderboards (Ibáñez et al., 2014; Neve et al., 2014; Pirker et al., 2014; Sprint & Cook, 2015), point systems and leveling (Ibáñez et al., 2014), and microworlds (Neve et al., 2014) into certain aspects of the learning environment (e.g. home works and practical sessions). Students express preference of a gamified strategy over the traditional because it is able to address their need for fun, pleasure, and cooperation (Kim, 2013). It taps directly into their fundamental desire for recognition, reward, status, competition, collaboration, and self-expression (González & Area, 2013).

1.2 Pitfalls of Gamification

Learning environments are commonly gamified by using badges, levels/leaderboards, achievements, and points because they are relatively easy to implement. This practice is referred to as reward-based gamification (Nicholson, 2015). However, this is effective in contexts that call for short-term change in behavior and those that can continue to supply the rewards (Nicholson, 2015). In the classroom, studies report on disparate reactions from students. Some perceived the method enthusiastically and performed well, whereas others felt demotivated and disengaged. Badges can be perceived as an unattractive alternative to grades (Pirker et al., 2014). Competition-based leaderboards may stimulate only a few and induce a feeling of embarrassment when available for public viewing (Kapp, 2012). Some students continue to fight for the top rank while some disengage altogether (Nicholson, 2013). They generate uneasiness leading students to dismiss it as a good representation of knowledge (DomíNguez et al., 2013). Competition is not fun for some students. Another drawback is when they become too driven by the competition that they lose sight of the real purpose, which is to learn (Sprint & Cook, 2015).

1.3 Meaningful Gamification

Nicholson broadens (Deterding et al., 2011)'s definition of gamification as the use of game design elements to help a user build intrinsic motivation – motivation not triggered by external rewards – to encourage engagement in a specific context (Nicholson, 2015). This is referred to as Meaningful Gamification. The theory behind is known as the Self-Determination Theory (Deci & Ryan, 2002) which states that there are three components associated with intrinsic motivation: mastery, autonomy, and relatedness. Mastery is when one learns to the point of competence; autonomy means having a choice and control of paths; and relatedness is about one's social engagement. There are contentions that gamifying the classroom entails rigorous research and design efforts to maximize its benefits. Otherwise, it will become ineffective because of the potential adverse effects to some students.

This paper presents the first part of a study to explore the effectiveness of gamification in an introductory programming class and its impact on the learning experience and achievement of novice programmers. It focuses on the design and implementation of a programming-based activity management system embedded with game design elements based on meaningful gamification. The results of an expert review of the system are summarized to determine whether it implements certain game design elements.

2. Research Objective

The main objective of the study is to experiment on the use of gamification in an introductory programming class. It seeks to explore how certain game design elements will impact the learning environment and whether or not it will positively affect the students' achievement and learning experience. To realize this, the first goal of the study is to design and build a pedagogical tool embedded with game design elements based on meaningful gamification. Prior to actual system development, gamification techniques and methods were evaluated to support the choice of elements and how they were incorporated into the system. Expert evaluation with CS1 instructors was conducted to assess whether these elements are found in the system.

3. Significance of the Study

Computer science educators and researchers continue to be concerned about student retention and attrition. This research will contribute to the on-going exploration of gamification to increase novice programmers' motivation. Since most experiments apply gamification through reward-based elements and have been shown to be less advantageous to some, this study delves into a more student-centric design that is anticipated to target their internal motivation to learn. Should the results reveal a positive impact on their achievement and learning experience, game elements that are more effectively applied in learning contexts may be identified. This can possibly spearhead future design efforts in fabricating new ways of delivering CS1 and instrumenting learning tools appropriate for novice programmers.

4. Methodology

This section discusses the design and implementation of a gamified programming-based activity management system for introductory programming classes. It is a web-based platform that allows teachers to manage assessment activities such as quizzes and laboratory exercises typical of CS1. The questions that may be included in the activities are divided into three categories as in (Orji et al., 2013): 1) factual or conceptual; 2) comprehension; and 3) generation questions. Table 1 lists the specific assessment activities the system provides for each category.

Table 1: Assessment activities for each category.

Category	Assessment Activity
Factual or Conceptual	Multiple Choice Questions, True or False, Identification
Comprehension	Code Tracing
Generation	Program Writing

Conceptual questions test students' knowledge of programming elements. Questions assess whether they are able to recall and recognize concepts learned during lectures. *Comprehension* questions check whether students are able to read code by determining the output or the values of a certain variable. Teachers provide code snippets that students have to read. *Generation* questions are comprised of programming exercises that test the students' ability to write programs.

The students' view features a coding area for each programming problem that they can use to write their programs (Figure 2). Compilation can be done through a button click. Compiler outputs, if any, are displayed. Program execution is automatically done when compilation results to no errors. In such a case, program output is displayed.

4.1 Evaluation and Selection of Game Design Elements

As discussed, gamifying learning environments do not always achieve the intended outcome. Ineffective classroom gamification could disengage students and in the most serious case, demean and demoralize to the point of dropping out (Nicholson, 2013). Reward-based practices, though easier to employ, do not maximize the potential of gamification. In this light, the choice of elements used in the system was anchored on meaningful gamification, purposely mapping them to the three components of the Self-Determination Theory. The learning module entitled Practical Guide to Meaningful Gamification discussed different ways that can support or oppose these components. A summary is found in Table 2.

Table 2: How to support or oppose autonomy, mastery, and relatedness.

	How to Support	How to Oppose
Autonomy	Providing students with choices	Mandatory participation
Mastery	Feedback cycles, allowing students to fail and retry	Lack of feedback, high stake consequences
Relatedness	Teams, competition	Working alone

Providing students choices make them feel in control of the paths they can take. Giving them several options for projects to work on or problems to solve affords them the opportunity to work on something they feel they are good at. This is a modest method of giving students a sense of autonomy.

The concept of "practice makes perfect" helps support mastery. When a system permits mistakes with no associated high stakes, students will be more willing to take risks knowing that they can recover. This is identified as a contributing factor to user engagement. When adopted in the classroom, this offers them a positive perspective on failure as a necessary part of learning (Lee & Hammer, 2011). Allowing the weaker students to reattempt failed work imposes a wider avenue for learning, improvement, and self-actualization (Nicholson, 2013). Immediate feedback is likewise important to help them understand the effects of their actions.

Lastly, designing social experiences can make them feel more like a part of the learning community. Involvement in a group plays a part in one's academic success (Ohno et al., 2013). Group learning can be a valuable component that maximizes learning gains (Pirker et al., 2014).

These concepts shaped the selection of the game design elements to be included in the system which are listed in Table 3. Further details on each one are discussed in the succeeding sub-sections.

Table 3: Game design elements employed in the system.

	Game Design Element
Autonomy	Control
Mastery	Feedback cycles, freedom to fail, progress
Relatedness	Collaboration

4.1.1 Autonomy

The students are afforded a sense of control by giving them the ability to choose which questions to answer (e.g. quizzes), problems to solve (e.g. laboratory exercises), or out-of-class challenges (optional) to undertake. The system allows teachers to include extra questions/problems and specify the number of required items that the students should answer. Students can choose from the set of questions/problems and answer/solve only up to what is required (e.g. solve 3 out of the 5 problems).

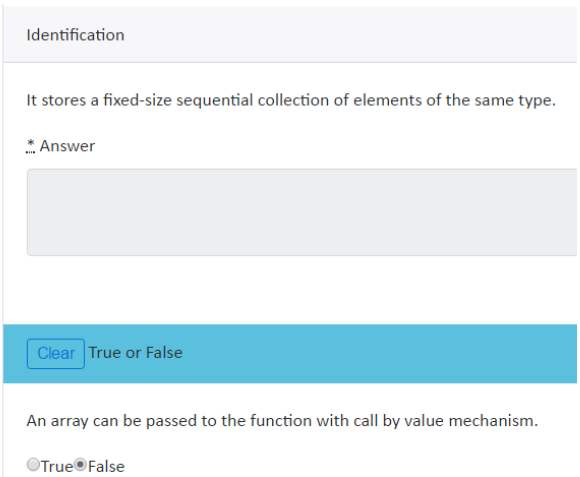

Figure 1. UI changes when a student answers the number of item required.

Activity items will be shuffled. Hence, each student will see them in varying orders. To prevent them from answering more than what is required, the system keeps track of the number of items they have answered so far. Questions, by default, will have blue headers. When they have reached the limit, extra items will be disabled. Header colors will turn to gray and will no longer be editable. Should a student want to change items, he/she can click on the “Clear” button beside each question previously answered. This will enable the extra questions, making them available for him/her to answer again. The “Clear” button is initially hidden and is only displayed when an item is answered. Figure 1 shows these changes in the UI.

4.1.2 Mastery

Mastery is supported through feedback cycles and the freedom to fail. Promptly giving feedback will let the students know whether their answers are correct or not. Freedom to fail gives them the chance to reattempt failed work which will optimally lead to mastery.

The feedback mechanism includes (1) automatically showing the score once an activity has been completed, and (2) marks indicating which items are correct or wrong. Once a student submits an activity, the results plus details about the activity are displayed at the top of the page. See Figure 2.

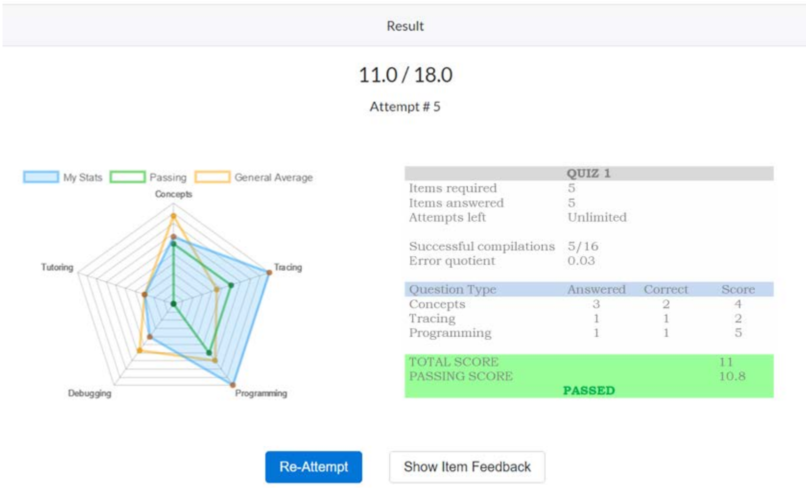

Figure 2. Results displayed upon activity submission.

A radar chart illustrating how they fared in the activity is also included (more on this below). A “Show Item Feedback” button that a student can switch on or off is available. When switched on, marks for correct and incorrect items are displayed. Headers of correct items are colored green; incorrect items are red, as shown in Figure 3. The student can then reattempt the activity and try to correct previous mistakes. When switched off, only the score and activity summary are shown. Though reattempts are still allowed, the student would not know which items were marked correct or wrong. This design choice was employed to be able to gather data that will help determine whether students are taking advantage of this feedback mechanism through the clicks on the “Show Item Feedback” button.

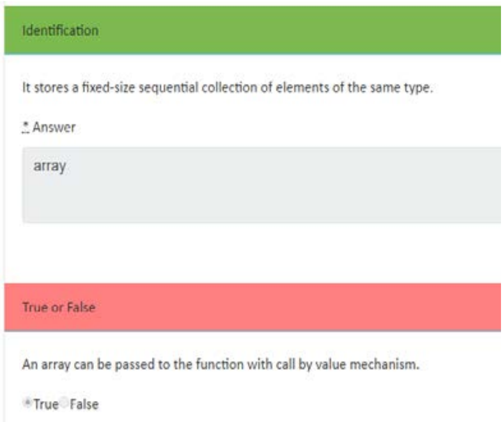

Figure 3. Correct / incorrect marks per item.

Still on mastery, progress for various skills is represented as a radar chart to give the students a general picture of how they are moving forward relative to the system-facilitated activities. This is displayed in their “Profile”, as shown in Figure 4, along with other information. The skills that will be tracked within the system are listed in Table 4 with the question types/activities to assess them. Students’ progress for

‘Concepts’, ‘Tracing’, and ‘Programming’ are based on their scores on the corresponding question types/activities in Table 4.

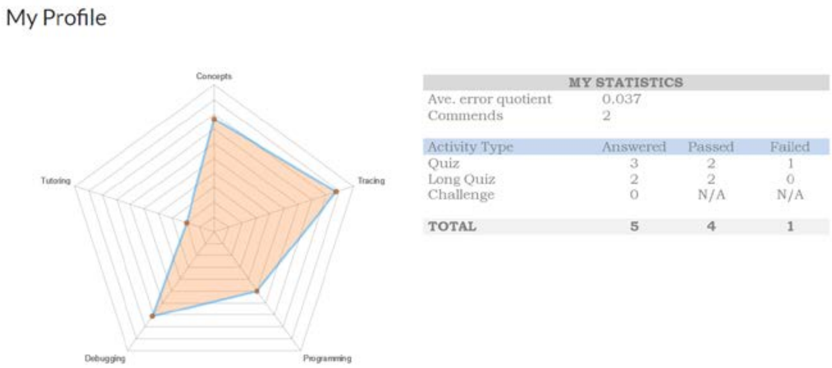

Figure 4. A student’s profile view.

Table 4: Skills represented in the progress chart.

Skill	Question / Activity	System Name
Knowledge of programming elements	Multiple Choice Questions, True or False, Identification	Concepts
Ability to trace code	Code Tracing	Tracing
Ability to correct syntax errors while writing a program	Program Writing	Debugging
Ability to write correct solutions to programming problems		Programming
Social/collaborative skills	Peer Mentoring	Tutoring

‘Debugging’ characterizes how students struggle with syntax errors while programming and is evaluated using the Error Quotient (EQ) algorithm, a quantification of students’ compilation behaviors by considering the type, location, as well as frequency of encountered syntax errors (Jadud, 2006a; Jadud, 2006b]. The EQ is a single number that may range from 0 to 1. An EQ of 1 implies that a student’s successive compilations ended with exactly the same error. Conversely, an EQ of 0 denotes a student is able to fix syntax errors in subsequent compilations. Data for EQ computations are unobtrusively collected while students write and compile their programs.

‘Tutoring’ represents how students collaborate with other students through peer-mentoring. This is discussed more in the next subsection. Aside from the radar chart on a student’s profile which represents their general performance, each activity will likewise have its corresponding chart representative of their performance in that activity. This is shown in Figure 4. It illustrates the student’s score against the activity’s passing mark. In addition, the chart will modestly show their general performance so they are able to see in one glance how they fared in the activity and how it may have affected their overall performance.

4.1.3 Relatedness

Targeting relatedness is the opportunity for collaboration through peer-mentoring. A student may be awarded with “tutor points” by another when the former mentors the latter. They get to decide who among their classmates they seek for help and whether they will reward them or not. They can do this through a ‘Commend’ button that appears beside a student’s name in the class’ view. ‘Tutoring’ is one among the five skills included in the radar chart discussed previously. This is to assess whether employing such a design element encourages students to work together and to help one another.

5. Expert Evaluation

Prior to testing the system with novice programmers, an expert evaluation was conducted to draw feedback from CS1 instructors with regards the implementation of the different game design elements into the activity management system.

5.1 Participants

Seven instructors from Ateneo de Naga University who have handled CS1 classes participated, two females and five males. They were of varying ages with the youngest at 27 and the oldest at 50. CS1 teaching experience ranged from one to five years. Five of them were familiar with the term “gamification” and two of them have implemented it by conducting a competition-based programming activity in their CS1 classes. Only one (1) used a gamified system, the Programming Contest Control System (PC^2) which was developed in support of the ACM computer programming contests.

5.2 Methods

The evaluations began with the instructors answering a demographics questionnaire. An introduction followed which aimed to put context to the evaluation being conducted. Gamification was briefly described particularly because two of the respondents were not familiar with the term. The details and objectives of the project were discussed. This included emphasis on the different game design elements and how they were implemented into the system. Then, a demo of the system was presented. After which, the instructors were asked to answer the debriefing questionnaire.

5.3 The Debriefing Questionnaire

The debriefing questionnaire was a modified version of the gamification inventory, an instrument for the structured assessment of gamification in a given system (Broer, 2015; Broer & Breiter, 2015). Items relevant to the elements incorporated into the system and that map to the components of the Self-Determination theory were selected. Other items that may be applicable were also included. Out of 38 items, twelve (12) were selected to be part of the debriefing questionnaire – one (1) for autonomy; five (5) for mastery; one (1) for relatedness; and five (5) other questions. Sample items are in Table 5.

Table 5: Example items from the debriefing questionnaire.

Component	Question
Autonomy	Does the system give the user a feeling of autonomy? Can the user decide what to do next? (autonomy)
Mastery	Does the system promote learning / mastery? (learning / mastery)
Relatedness	Does the system provide the user with a feeling of relatedness to other users? Is there social contact? (relatedness)

It used a 5-point scale with response options from Strongly Disagree (1) to Strongly Agree (5). The respondents were to select Neutral (3) if they think the system potentially exhibited the game design element, but relied on user input for it. They were asked to elaborate on their levels of agreement. If they agree/strongly agree that an element is implemented, they were asked to identify the specific feature/s of the system that exhibits the element.

5.4 Findings

As mentioned, the game design elements employed are: control to enable autonomy; feedback cycles, freedom to fail, and progress to support mastery; and collaboration for relatedness. The expert evaluation was conducted to specifically assess the presence of these elements along with other items that seemed relevant.

5.4.1 *Autonomy, Mastery, Relatedness*

The frequency of responses for the questions that fall under autonomy, mastery, and relatedness is summarized in Figure 5.

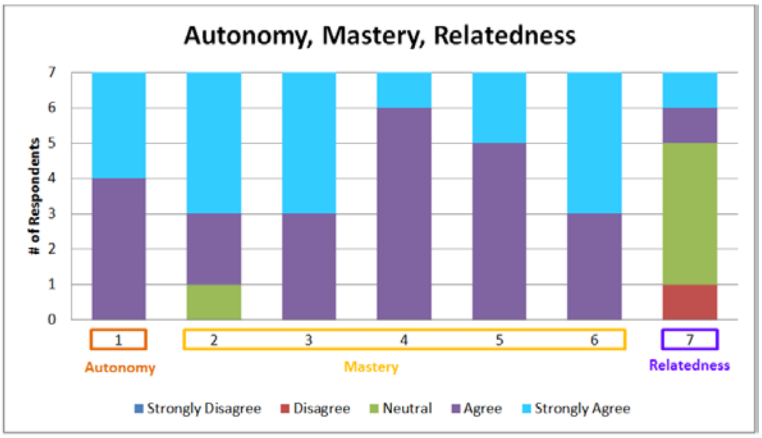

Figure 5. Frequency of responses for autonomy, mastery, and relatedness questions.

The evaluators agree that the system provides for autonomy by allowing students to select items to answer in a certain activity. This gives the students a sense of control over the questions that they would like to answer, allowing them to choose those they feel they are capable of answering.

They generally agree that the elements for mastery are implemented. The system provides students with a sense of accomplishment through the radar chart and instant feedback (Q2). According to them, the radar chart can motivate the students to be an "expert" in all areas or on certain favored areas. It may encourage students to work more. The radar chart, activity statistics, and immediate feedback can give them a feeling of competence (Q3). Displaying the passing score in an activity's radar chart shows the students that they must achieve certain marks to get better scores. Learning/mastery is promoted by allowing students to re-attempt activities. The repetition and familiarization can enhance learning. Through the feedback mechanism, a student can understand which parts of an activity he/she correctly understood. However, one respondent raised a concern about the possibility of students avoiding certain questions – those that may be too difficult for them (e.g programming). Meaningful feedback is provided through displaying the scores, activity statistics, and the correct/wrong answers (Q4). Progress is provided through the radar chart and the summary statistics allow students to sense how well they are doing (Q5).

As for relatedness, a couple of respondents agree that awarding tutor points can encourage peer-mentoring among the students. Majority, however, responded with neutrality, as seen in Figure 7. According to them, subjectively giving out points may not necessarily encourage students to mentor their peers and feel more connected with one another. One respondent even disagreed and supposed that students might just "pair-up" and give each other points.

5.4.2 *Others*

Four out of seven agree that the different question types available in the system can provide for challenging tasks for the students (Q1). However, teachers play an important role in constructing challenging questions. One of them disagreed and suggested that allowing students to view other students' profiles can help support this game element. Showing the progress of the students through the radar chart may increase the students' attachment to outcome (Q2). Since the chart

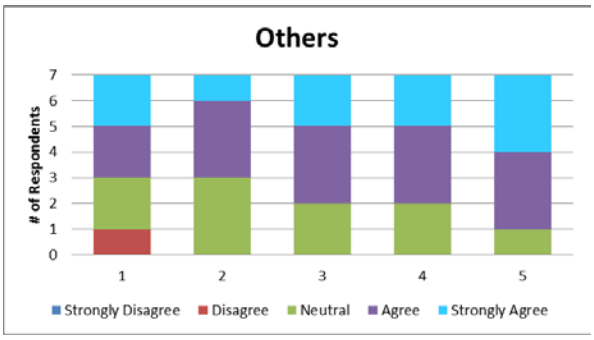

Figure 6. Frequency of responses for other items.

depicts their progress in terms of the different activities facilitated through the system, this can possibly urge them to work to further their performance in certain or all areas. Positive emotions can be triggered by displaying their scores and progress, especially if they see improvements in their scores after several attempts (Q3). Most of them agree that incorporating the different game design elements can spark more interest and increase focus/attention (Q4). Lastly, the radar chart and summary statistics can represent students' behavior/performance (Q5).

6. Conclusion and Future Works

This paper presented the design, implementation, and expert evaluation of a gamified activity management system for novice programmers as part of a larger study on the exploration of the effects of gamification on students' achievement and learning experience in CS1.

Gamification is becoming popular as a classroom intervention. However, previous studies mostly applied reward-based game elements and were shown to be not effective for some. Hence, this research intends to explore the impact of a tool founded on meaningful gamification. The selection of game design elements implemented into the system should map to the different components of the Self-Determination Theory. The elements employed are: control to enable autonomy; feedback cycles, freedom to fail, and progress to support mastery; and collaboration for relatedness.

Expert evaluation was conducted to seek insights from CS1 instructors with regards to the presence of said elements in the system. The respondents generally agree that the elements for autonomy and mastery existed. Though, they were more neutral with regards to relatedness. They do not think that subjectively awarding tutor points encourages students to mentor one another. As for the other additional items, responses were generally positive.

Moving forward, the system will be tested on novice programmers. Control and experimental groups will be asked to use a non-gamified and gamified version, respectively, to be able to gather data that can possibly differentiate students' experience and performance. Data will then be analyzed to determine whether the students' behaviors in response to the system can essentially be attributed to the game design elements.

Acknowledgements

We would like to thank the instructors for responding to the invitation to evaluate the system. We also like to thank the Department of Computer Science, Ateneo de Naga University for providing a venue for the evaluation. Much gratitude to the Department of Science and Technology–Engineering Research and Development for Technology (DOST-ERDT) for the scholarship awarded to the student researcher. Lastly, we thank the Ateneo Laboratory for the Learning Sciences (ALLS), Ateneo de Manila University for supporting this study.

References

Beaubouef, T., & Mason, J. (2005). Why the high attrition rate for computer science students: some thoughts and observations. *ACM SIGCSE Bulletin*, 37(2), 103-106.

Bennedsen, J., & Caspersen, M. E. (2007). Failure rates in introductory programming. *ACM SIGCSE Bulletin*, 39(2), 32-36.

Broer, J. (2015). Is this new? Family Resemblances in Gamification in Education. *BULLETIN OF THE TECHNICAL COMMITTEE ON LEARNING TECHNOLOGY, 17*(4), 14-17.

Broer, J., & Breiter, A. (2015). Potentials of Gamification in Learning Management Systems: A Qualitative Evaluation. In *Design for Teaching and Learning in a Networked World* (pp. 389-394). Springer International Publishing.

Deci, E. L., & Ryan, R. M. (2002). Handbook of self-determination research. University Rochester Press.

Dehnadi, S., & Bornat, R. (2006). The camel has two humps (working title). Middlesex University, UK, 1-21.

Deterding, S., Björk, S. L., Nacke, L. E., Dixon, D., & Lawley, E. (2013, April). Designing gamification: creating gameful and playful experiences. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems* (pp. 3263-3266). ACM.

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011, September). From game design elements to gamefulness: defining gamification. In *Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments* (pp. 9-15). ACM.

DomíNguez, A., Saenz-De-Navarrete, J., De-Marcos, L., FernáNdez-Sanz, L., PagéS, C., & MartíNez-HerráIz, J. J. (2013). Gamifying learning experiences: Practical implications and outcomes. *Computers & Education*, 63, 380-392.

Gibbons, T. E. (2013, October). COR: A new course framework based on elements of game design. In *Proceedings of the 14th annual ACM SIGITE conference on Information technology education* (pp. 77-82). ACM.

Glover, I. (2013). Play as you learn: gamification as a technique for motivating learners.

González, C., & Area, M. (2013). Breaking the rules: Gamification of learning and educational materials. In *Proceedings of the 2nd international workshop on interaction design in educational environments* (pp. 7-53).

Harrington, B. (2016, May). TrAcademic: Experiences With Gamified Practical Sessions for a CS1 Course. In *Proceedings of the 21st Western Canadian Conference on Computing Education* (p. 25). ACM.

Hoda, R., & Andreae, P. (2014, January). It's not them, it's us! Why computer science fails to impress many first years. In *Proceedings of the Sixteenth Australasian Computing Education Conference-Volume 148* (pp. 159-162). Australian Computer Society, Inc..

Ibáñez, M. B., Di-Serio, A., & Delgado-Kloos, C. (2014). Gamification for engaging computer science students in learning activities: A case study. *IEEE Transactions on Learning Technologies*, 7(3), 291-301.

Jadud, M. C. (2006a). An exploration of novice compilation behaviour in BlueJ. University of Kent.

Jadud, M. C. (2006b, September). Methods and tools for exploring novice compilation behaviour. In *Proceedings of the second international workshop on Computing education research* (pp. 73-84). ACM.

Kapp, K. M. (2012). Games, gamification, and the quest for learner engagement. *T+ D*, 66(6), 64-68.

Kim, S. (2013). Effects of the gamified class in engineering education environments. *Journal of Convergence Information Technology*, 8(13), 253.

Klopfer, E., Osterweil, S., Groff, J., & Haas, J. (2009). Using the technology of today in the classroom today: The instructional power of digital games, social networking, simulations and how teachers can leverage them. *The Education Arcade*, 1, 20.

Kumar, B., & Khurana, P. (2012). Gamification in education-learn computer programming with fun. *International Journal of Computers and Distributed Systems*, 2(1), 46-53.

Lee, J. J., & Hammer, J. (2011). Gamification in education: What, how, why bother?. *Academic exchange quarterly*, 15(2), 146.

Neve, P., Livingstone, D., Hunter, G., Edwards, N., & Alsop, G. (2014). More than just a game: Improving students' experience of learning programming through gamification. A paper from the *STEM Annual Conference* 2014.

Nicholson, S. (2013). Exploring gamification techniques for classroom management. *Games+ Learning+ Society*, 9.

Nicholson, S. (2015). A recipe for meaningful gamification. In *Gamification in education and business* (pp. 1-20). Springer International Publishing.

Ohno, A., Yamasaki, T., & Tokiwa, K. I. (2013, August). A discussion on introducing half-anonymity and gamification to improve students' motivation and engagement in classroom lectures. In *Humanitarian Technology Conference (R10-HTC), 2013 IEEE Region 10* (pp. 215-220). IEEE.

Orji, R., Mandryk, R. L., Vassileva, J., & Gerling, K. M. (2013, April). Tailoring persuasive health games to gamer type. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 2467-2476). ACM.

Pirker, J., Riffnaller-Schiefer, M., & Gütl, C. (2014, June). Motivational active learning: engaging university students in computer science education. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 297-302). ACM.

Robins, A. (2010). Learning edge momentum: A new account of outcomes in CS1. *Computer Science Education*, 20(1), 37-71.

Ross, P. (2011). Math teacher uses gamification to help at-risk students succeed. *Recuperado el*, 24.

Sprint, G., & Cook, D. (2015, March). Enhancing the CS1 student experience with gamification. In *Integrated STEM Education Conference (ISEC), 2015 IEEE* (pp. 94-99). IEEE.

Watson, C., & Li, F. W. (2014, June). Failure rates in introductory programming revisited. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 39-44). ACM.

Wortman, D., & Rheingans, P. (2007). Visualizing trends in student performance across computer science courses. *ACM SIGCSE Bulletin*, 39(1), 430-434.