Enhancing Software Engineering Learning Environment with Computer Games: A Case Study

Aaron R. Rababaah
Associate Professor of Computing
College of Engineering and Applied Sciences
American University of Kuwait - AUK
arababaah@auk.edu.kw

Abstract : Education gamification has been spreading in various disciplines such as languages, computer programming, medicine, natural languages, engineering, etc. Software Engineering is our interest in this work as we saw an opportunity of contribution to enrich literature and empirical studies in this area. Traditional methods of teaching Software Engineering could significantly benefit from gamification as a complementary component in student learning outcomes. We believe we can provide our students with more effective learning environment in number of aspects including: providing enjoyable practice, immediate feedback, enhancing the sense of responsibility, enhanced engagement and performance real time tracking. In this paper, we will present our case study in adopting a computer game in software engineering course. Further, we will present the results of a course exist survey that shows the responses of 114 participating students. The analysis of the survey showed significant positive impact on number of aspects including: student engagement, learning concepts and critical thinking. The overall mean of positive responses was 81.2%.

Aaron R. Rababaah
Associate Professor of Computing
College of Engineering and Applied Sciences
American University of Kuwait - AUK
arababaah@auk.edu.kw

Keywords: Software Engineering Education, Enhancing Learning Environment, Utilizing Games in Learning.

1. Introduction

Traditional ways of learning may be becoming less interesting to newer generations of students at all levels (IEAB, 2019; Dreyer, 2019). Information revolution, gamification, smart things, etc. bring a real challenge to the educational process on how to evolve to accommodate new generation of students. Throughout our experience as educators, we have seen intrinsic motivation of students decline in the last decade. Therefore, innovative methods are needed to make learning more interesting and enjoyable (Usher, 2019; Afdal, 2010; Rababaah & Rabaa'I, 2018; Liu et al., 2013; Taran, 2007; Claypool, 2005; Navarro, 2004; Connolly et al., 2008). There are many attempts to incorporate innovative technologies and pedagogical methods to improve learning environment as in (Eguchi, 2014; Rababaah & Raba'ii, 2018, Tomoko, 2015; Özüorçun et al. 2017; Gozcu & Caganaga, 2016). As these studies address the general need for enhancing learning experience for students, in our study, we focus on utilization of games to improve student engagement and interest in course work. It is evident in the literature that gaming has positive impact on student learning, engagement and performance but it is not yet wide-spread (Papadakis, 2018; Stathakis, 2019; IEAB, 2019, Greipl et al., 2020; Fellenhofer, 2018). Some of these benefits include: making learning more enjoyable, providing a context to learning objectives, providing rich environment that stimulate critical thinking, immediate feedback, tracking performance, enhancing memory by providing audio, visual, textual media and enhancing engagement via grapping attention and active participation (Stathakis, 2019; IEAB, 2019; Koivisto & Hamari, 2019). Therefore, we believe that our study will contribute to the spread of this interesting method of learning and confirm its positive impacts on students learning outcomes.

In this section we review some related work in the area of utilizing games in education in general and then in Software Engineering (SE) in particular. The work of (Bahadoorsingh et al., 2016) presented their experience in adopting Game-Based Learning (GBL) in power system analysis course. They authors argued the need for GBL as a facilitator of transforming theory into practice. Their work reported a successful usage of IBM's Innov8 game (IBM, 2020), where two consecutive cohorts were used and studied. The authors of (Al-Sharafat et al., 2012) investigated web-based gamification of language communication skills. Their study showed that GBL significantly improved students achievements compared to traditional learning. A game model was proposed by (Minovic et al., 2010) that considers not only knowledge integration into the Game but also, content reusability as well. Their model was successfully implemented and demonstrated that educators can create new GBL for their context by reusing already established knowledge base in the model. The satisfaction of students was investigated for using a game in language learning by (Tsai et al., 2017). Some important findings of their study include: intrinsic interest in language learning affects students' satisfaction of the Game, interest in playing games had no effects on students' satisfaction and playing time and satisfaction were correlated. The work of (Dreyer, 2019) presented a study that aimed at investigating the impact of GBL in teaching programming over traditional methods. The study reported a significant learning gains of GBL compared to traditional learning setting. A study of GBL in entrepreneurship learning was conducted by (Fellenhofer, 2018). The study is claimed to be one of the few in the area of entrepreneurship learning. The results of the study draw attention to significant positive influence in entrepreneurship learning elements such as attitudes, intention and behavior.

We discuss number of related works to GBL in software engineering in this section as follows. A survey of gamification in software engineering (SE) education was done by (Campolina et al., 2018). The main goal of their survey was to reach out to SE educators and know the extent of their gaming adoption. The study revealed that most of professors are aware of gamification but only 24% adopted games in their SE courses. A game was developed and used in teaching risk management by (Taran, 2007). The main goal of the Game was to enhance practical learning and decision making through project simulations. The authors reported a clear learning advantage of GBL over traditional methods. Furthermore, the Game helped understand concepts and added enjoyment factor to learning. The work of (Claypool, 2005) reported a pilot study of GBL as an effort to improve students' interest and retention in their SE program. The used game provides learning about software (SW) life cycle, project real issues and team management. The study confirmed the advantages of GBL by improving student's participation and performance. The authors of (Ye et al., 2007) experimented with the online game “Second Life” and SimeSE game (SIMSE, 2020) in two computer classes for software engineering to enhance collaboration and communication between team members. Furthermore, the Game was used as virtual offices for instructors with office hours for students. The authors reported a positive impact of GBL on student learning compared to traditional methods.

Although we have seen evidence that GBL being utilized in SE education but, there is some evidence that SE education still lacks the recommended level of GBL (Campolina et al., 2018; Connolly et al., 2008; Thomas, 2007). Our goal in this paper is to present our case study in utilizing GBL in our SE course using the Game SimSE (SIMSE, 2020). We will go through our experience in adopting this game and present and discuss the results of a student survey of 120 students who played the Game as an assessed component in the SE course.

2. SimSE Game

SimSE was developed by (Navarro, 2004) as a 2D graphical game designed specifically for SE education. The Game is based on a simulator that provides diverse models of SE processes that allow students to be trained on different challenging practical problems. In this section we will provide explanation of the elements of the simulator and its capabilities. The different elements are labeled on the Game interface in Figure 1.

(1) Challenge Description: provides detailed description of the project challenge with the main items: customer requirements, budget including time and money allocated for the project, score criteria and hints about the specific project domain.

(2) Game Elements: in this section there are 5 elements that players should keep track of throughout their progress:

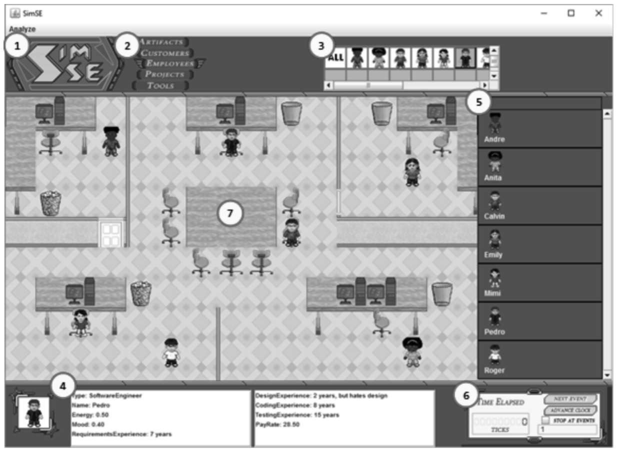
a. Artifacts: represent the SE activates needed to run the process of the current model which typically includes: requirements documents, design documents, implementation code and testing plans.

b. Customer: information of the customer requesting the software system.

c. Employees: keeps details about software engineers assigned to work on the project.

d. Projects: information about the requested system

e. Tools: available development computer aided software engineering (CASE) tools for engineers to acquire and use throughout the Game. These tools include: requirements capturing tool, design environment, integrated development environments (IDEs) and automated testing tools.



Type: SoftwareEngineer
Name: Pedro
Energy: 0.50
Mood: 0.40
RequirementsExperience: 7 years

DesignExperience: 2 years, but hates design
CodingExperience: 8 years
TestingExperience: 15 years
PayRate: 28.50

Fig. 1 : Elements of SimSE game interface

(3) Elements Navigations: lists individual members in an element in icon strip. When an element is selected by a player, all current states of that element can be viewed using this panel then seeing them on information port in (4) below.

(4) Information Port: this is used to display all states of an individual item in a game element. For example, employee states include: energy, mode, payrate, task, experience, etc. and artifact states include: name, percentage completed, number of errors, percentage integrated, etc.

(5) Employees Panel: displays a list of icons for all employees and what are they working on currently. This helps to keep track of the team task assignments.

(6) Game Timer: keeps track of time spent on the project and allows players to pause the Game, run for number of ticks and stop or stop at events. Events could be an employee got sick, the customer requested a change, an employee has finished his/her task, etc.

(7) Work Area: representation of office areas of employees with their respective avatars. If an avatar is selected, it displays all of its states in information port as in item (4) above.

To successfully play SimSE, players must understand the concepts of SE process models as the simulator is designed and implemented based on these concepts. SimSE supports 6 SE process models: Waterfall, Incremental, Prototype, Extreme Programming (XP), Inspection and Rational Unified Process (RUP). In all of these models, there are a set of common activities including: requirements specification, design, implementation, testing and evolution. They are typically called activities but not stages as they may be interleaved and organized differently in different process models. The reader is invited to review (Sommer, 2016 & Shelly, 2011, IBM, 2020) for more details on these models. These process models are defined as follows:

Waterfall process model: SE activities are broken down and organized in distinct stages where each stage is dependent on the outcomes and deliverables of its predecessor stage. Development is done strictly one stage at a time; at the completion of each state the successor stage is initiated.

Incremental process model: the software system is broken down into deliverable testable increments. Each increment is a subset of the overall components of the system. Inclusion of a component in an increment is based on customer value, criticality and service scope of the component.

Prototype process model: this model starts by developing an initial version of the system focusing on functional requirements but not on non-functional requirements such as: efficiency, reliability, security, etc. After that, development is done iteratively to further develop and improve the initial version. Testing is conducted in each version to verify and validate requirements.

Extreme Programming (XP) process model: XP is a software development methodology that implements Agile SW principles. Main characteristics of this model include: incremental short-term planning, small system releases, minimum design and documentation and refactoring.

Inspection process model: this model applies verification and validation activates to implement quality control and quality assurance of the software system throughout the development process.

Rational Unified Process (RUP) model: it is development model that applies best practices in modern software development. These best practices include: iterative development, requirements engineering, component-based architecture, visual models using Unified Modeling Language – UML (OMG, 2020), quality verification and change control. RUP consists of 4 phases: Inception, Elaboration, Construction and Transition.

Next sections will present and discuss how these different simulated models helped us in complementing our traditional course work by adding game labs to the class and let students learn to solve practical software engineering problems as well as project management challenges such as team, budget and time management.

3. Game-based Labs

In this section we will explain the algorithm players should follow when playing SimSE. The algorithm is graphically depicted in the process flow diagram in Figure 2 below.

Select process model: players need to choose which SE process model they want to run for the project scenario. SimSE simulator has different internal logic and criteria for each process model including: waterfall, incremental, prototype, XP, Inspection and RUP.

Review project requirements and constraints: for each project scenario, the simulator provides a description of the requirements and constraints that we need to satisfy. Some examples of these are as follows:

- Project scenario: you are asked to develop an online grocery ordering website. Customers should be able to register online with their personal information including name, phone, address, username and password. They should be able to browse available grocery items, select items and quantities, place an order and pay with credit card.

- § Project budget: for each project the player is given an upper limit of money to be spent in buying automated tools such as requirement elicitation, integrated development environment, testing tools, design tools, etc. Employees are payed regardless they are kept busy or idle. Players can

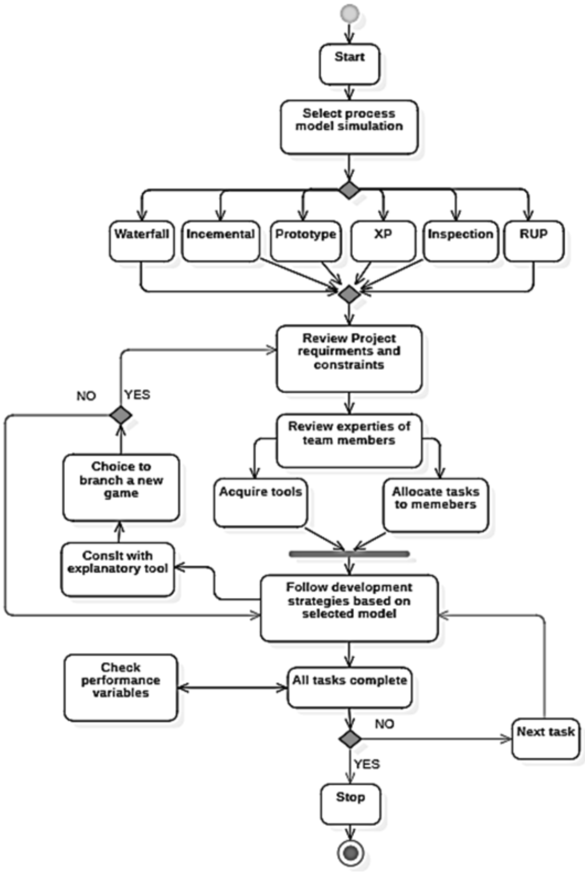

Start
Select process model simulation
Waterfall
Incemental
Prototype
XP
Inspection
RUP
Review Project requirments and constraints
NO
YES
Review experties of team members
Choice to branch a new game
Acquire tools
Allocate tasks to memebers
Conslt with explanatory tool
Follow development strategies based on selected model
Check performance variables
All tasks complete
NO
Next task
YES
Stop

Fig. 2 : Game playing process diagram

fire any employee during the Game. Budget is one of the main scoring criteria in the Game.

- Time limit: an upper limit of time is assigned to each project. Time is represented in simulation ticks where, SimSE already includes time estimate to finish each project. Violating this limit negatively affect score.

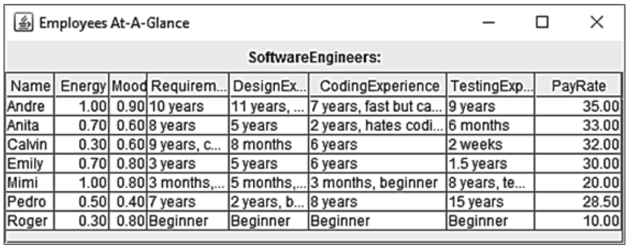
- Score criteria: the main elements of score criteria include: error percentage of the final system, completeness of system integration, budget and time schedule.

- Guidelines: Each game scenario will give players some guidelines to help them make effective decisions. Examples of these guidelines are: all hired employees are getting payed at every clock tick regardless of their status; utilizing CASE tools improve efficiency and accuracy; CASE tools are not automatically acquired, players must explicitly purchase these tools; employees mood and energy vary and need to be monitored and they can be improved via incentives and breaks.

Review expertise of team members: The Game tries to simulate real world diversity of team expertise and experience. Figure 3 shows a sample of employee profiles where players need to consult with frequently to ensure effective task allocation and decision making throughout the Game.

Acquire tools: players are advised to purchase CASE tools not only to improve efficiency but to improve accuracy. The simulator makes a significant difference in advancing development and estimating errors in SE activities if players choose not to use CASE tools which will degrade their score eventually.

Allocate tasks: after understanding the project at hand, customer requirements, project guidelines, constraints and employee profiles, players can make informed decisions how to allocate tasks to team members in timely fashion.



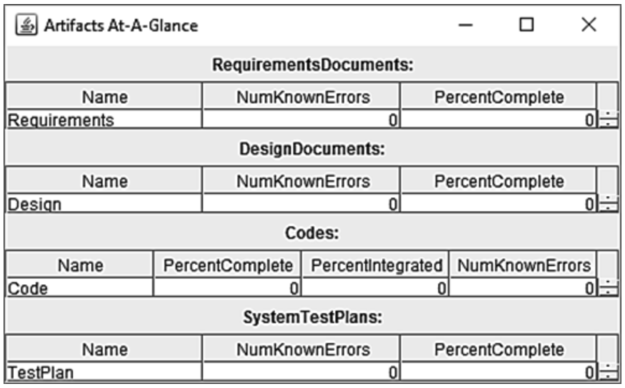
SoftwareEngineers:

Name	Energy	Mood	Requirem...	DesignEx...	CodingExperience	TestingExp...	PayRate
Andre	1.00	0.90	10 years	11 years, ...	7 years, fast but ca...	9 years	35.00
Anita	0.70	0.60	8 years	5 years	2 years, hates codi...	6 months	33.00
Calvin	0.30	0.60	9 years, c...	8 months	6 years	2 weeks	32.00
Emily	0.70	0.80	3 years	5 years	6 years	1.5 years	30.00
Mimi	1.00	0.80	3 months,...	5 months,...	3 months, beginner	8 years, te...	20.00
Pedro	0.50	0.40	7 years	2 years, b...	8 years	15 years	28.50
Roger	0.30	0.80	Beginner	Beginner	Beginner	Beginner	10.00

Fig. 3 : Employees at a glance

Follow development strategies: the 6 mentioned models we have defined earlier in section (2) follow different strategies in accomplishing software development for example, Waterfall model breaks down system development into distinct stages, each stage is dedicated to a single SE activity so development is strictly sequential. On the other hand, Extreme Programming model work on all SE activities in an interleaved fashion so all activities are active at every stage which promotes parallel, iterative and incremental development. Therefore, players are required to learn the model of choice and manage their project accordingly to match the expectation of the simulator as much as possible.

Check tasks for completion: periodically, players need to consult with states of project artifacts such as: requirement documents, design documents, code implemented, test plans, system integration, etc. players keep iterating on tasks until all are accomplished. Figure 4 depicts these artifacts.



RequirementsDocuments:

Name	NumKnownErrors	PercentComplete
Requirements	0	0

DesignDocuments:

Name	NumKnownErrors	PercentComplete
Design	0	0

Codes:

Name	PercentComplete	PercentIntegrated	NumKnownErrors
Code	0	0	0

SystemTestPlans:

Name	NumKnownErrors	PercentComplete
TestPlan	0	0

Fig. 4 : Artifacts at-a-glance

Check performance variables: players need to continuously monitor budget, time, errors, employee energies, etc. to make corrective actions as needed. Catching problems early will enhance the chance of meeting game expected final outcomes.

Consult with explanatory tool: while playing a game, it is highly recommended that players look at performance analysis tool provided by SimSE. This tool includes number of important educational features such as: view graphical charts of employees to monitor their attributes; plot action graphs where, player actions are plotted vs. time which helps players to trace and track the effectiveness and consequences of their actions to the progress of the project and view rules of the Game where, players can get insight how their scores are calculated.

Branch to new game: SimSE allows players to freeze current game at any point in time and branch to a new game to try different decisions and learn from parallel results. It is a "what-if" analysis that is very ideal for student learning cause-effect relationship. An example of number of games played by the same player is shown in Figure 5 below where, the chart shows the overall score for each game with time. This feature is optional so, players can proceed with the same game forward till the end as it can be observed in the flowchart in Figure 2.

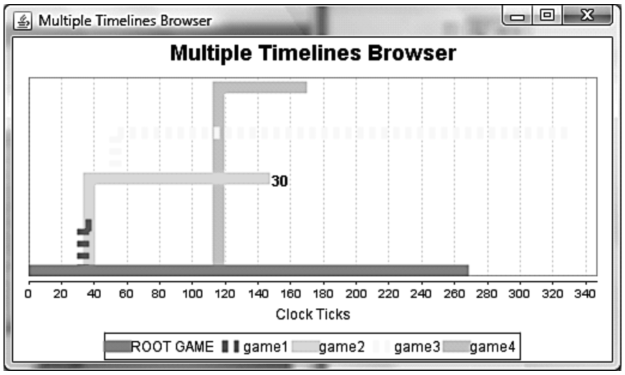


Fig. 5 : Multi-game branching by one player (Navarro, 2020)

4. Mapping Course Content to the Game

Our SE course content is based on typical SE undergraduate course structure that includes: Introduction to SE, SW process models, requirements engineering, system modeling, architectural design, system design, implementation, testing, evolution, project planning and project management (Summerville, 2016; Pressman, 2014; Stephens, 2015). In this section, we will describe how these topics were mapped to SimSE game in-class labs and take-home homework assignments. Students do multiple in-class labs and multiple off-class labs. The number may vary depending on the semester. Students are exposed to the Game when they have sufficient knowledge in SW processes. This stage is reached typically after completing these subjects: introduction to SE, process models and project management.

SW process models: as we have seen in the discussions about SimSE, the Game supports several models of SE including Waterfall, Incremental, Prototype, XP, Inspection and RUP. Students are exposed to these models and they get to practice playing them after understanding their concepts and their tradeoffs.

Requirements engineering (RE): students get to learn practically where RE fits in the SE process and how critical it is to the overall success of the SE process. The Game penalizes players if they completely missed RE activity or they do not properly apply it to the project at hand.

System Design: students learn to apply design actions in the Game, review and correct their designs before they move to the next activity in the SE process. Students get to appreciate the design-review-improve cycle since the Game will penalize them if they deviate from this cycle. They typically lean to catch errors as early as possible as it gets very expensive as advance forward.

Implementation: most students, if not all, before they join SE course, they start with implementation as the first step in their programming project. After learning SE concepts and process models, etc. they appreciate all the required professional work before implementation including: planning, requirements engineering, system modeling and design. The Game applies these concepts to train students on following these SE activates in the right manner and in the right order.

System Testing: testing is part of broader umbrella of Verification and Validation (V&V) (Summerville, 2016). V&V apply principles of inspection, user requirements, system requirements and code execution to ensure quality in all SW development stages. The Game teaches students to apply these principles in timely fashion to discover errors and correct them gradually as they progress otherwise, their score in the Game will suffer eventually.

Project Management: as discussed earlier, SimSE game is project oriented. So, it is not only about going through SE process but also applying project management principles as well. The following are examples of where in the Game these principles are applied:

- CASE tools: students learn that CASE tools are feasible to invest in and use throughout the project. Requirements elicitation tool for example is used to collect, review and validation user requirements early on in the project. This tool helps to expedite requirement collection and ensure their consistency. IDEs for coding are essential tools to rapidly edit, organize, debug and test software. Testing packages are highly recommended as they facilitate testing automation which improves efficiency and quality of the development process.

- Task allocation: one of the important activities in project management is allocation of work among team members. Students practice how to review attributes of hired employees in the Game such as mood, energy, expertise area(s), length of experience, payrate etc. these attributes are vital to know when making decisions on who should do what. Inadequate decisions in task allocation will result in low performance or project failure.

- Budget and Time: every game scenario specifies upper limits for money and time to be spent on the entire project. Students learn to monitor money and time variables as they get closer and closer to reaching the upper limits. They get to learn how every decision they make during the game can impact these two resources. For example, a player can decide to not to buy a CASE tool to money but he/she will be surprised later that development process is slow and errors are more frequent.

- Employee states: the Game changes energy and mood of workers as they spend more hours working on the project. Students learn to monitor these two states and try to fix them by giving worker incentives or breaks as needed. These actions improve productivity and accuracy of employees which consequently affects the success of the project.

- Team work: Students in the same team can play the same model on separate computers trying different strategies then meet and discuss the effectiveness of their strategies. This helps the team to work in parallel to tryout alternatives, evaluate them and select and adopt the best to tackle the scenario at hand.

- Critical thinking: SimSE game provides a good environment for critical thinking. The following are good examples on where students practice this important skill:

- Understanding process models: careful review and understanding of the process model to be applied in a game scenario is the most important factor for the success of the project. For example, if the selected model to be played is XP and players applied Waterfall model instead, they will find out soon that the project is not working well and will properly result in failure.

- Customer requests: during the game students will always experience that their customers try to interrupt their work by sending design change requests. The role of students is to carefully evaluate these requests and take decision whether to accept or decline them. Some times these changes make since as they come with extended budget and/or time.

- Game analysis: students have the opportunity to freeze the game in time and conduct "what-if" analysis. For example, if they see error rate is high during a game they can stop and branch to fix the problem. In this particular case, they may look into root causes of error which could be: wrong task allocation as inexperienced workers could be assigned activities of review and testing or document reviews are not done at all. The new branch of the game can fix these problems and hopefully better results are observed.

- Project management activities: as described earlier, many activities in project management also involve critical thinking skills such as: budget and time management, task allocation and game performance tracking. So, students get to practice situation analysis and decision making throughout the game.

Table 1 below contains a summary of course subjects support and their learning opportunities in the game. The table is meant to give the reader a quick

Table 1 : Summary of Software Engineering course content to SimSE game. Legend: 0=none (red), 1=low (orange), 2=medium (yellow), 3=high (green).

Course Subject	Game's support	Concentration in the game	Learning Opportunity
Introduction to SW	1	1	1
Process models	3	3	3
Requirements engineering	2	1	2
System modeling	0	0	0
Architectural design	0	0	0
System design	2	2	2
Implementation	2	1	1
Testing	3	3	3
Evolution	2	2	2
Project management	3	3	3

look at-a-glance how SimeSE supports learning of typical content of SE course. The following key/legend is used: 0=none (red), 1=low (orange), 2=medium (yellow), 3=high (green). It can be observed that the three main areas with highest support and greatest learning opportunities in the Game are: process models, testing and project management.

5. Student Feedback Survey

An anonymous 5-point Likert scale survey was designed to investigate the effectiveness SimSE game on student learning of the relevant topics in the course. The survey was created using Google survey service (Google, 2020) and sent to our students in 8 consecutive semesters from Spring 2016 to Fall 2019 where, SimSE game were used in our SE classes. The questions of the survey are listed in Appendix (A). We collected 114 responses and we listed their individual results in Appendix (B). The survey consisted of 30 questions that include the 4 categories described in Table 2.

Table 2 : Categories of the student survey

No.	Category	Questions
1	Demographics	1-3
2	Confidence in own major	4
3	Game usability	5-10
4	Game effectiveness in student learning	11-22 & 26-27
5	Class structure supporting the Game activities	23-25
6	Student final conclusion and future recommendation	28-30

To quantify results summaries, we combine the Likert scale two top levels (Strongly agree & agree) from responses for each variable we are studying and we compute their ratio to the total number of responses as expressed in Equation (1).

$$RPR = \frac{NPR}{NTR} \qquad (1)$$

Where:

RPR = ration of positive responses

NPR = number of positive responses = "strongly agree" + "agree"

NTR = total of total responses = NPR + "neutral" + "disagree" +"strongly disagree"

In this section, we look at each variable in the demographics independently to understand its effect on the survey results in each different category.

5.1 Results of category 2 (Confidence in own major)

Figure 6 depicts RPR for female and male respondents for question 4 (confidence in major). As it can be seen, RPR is 70% for female and 79% for male respondents. This question helps us to support our confidence in the results as it shows how enthusiastic the population sample about their majors. For the different majors chart, one alarming observation is the Engineering other "EO" which shows 0.0% confidence that students are confused about their major. As for the other three majors, the confidence level is acceptable (70%) or very good (81%). As for the college year, the chart shows that acceptable percentage of students who are confident of their majors although, one may expect this confidence to be positively correlated with age but the given chart does not reflect that.

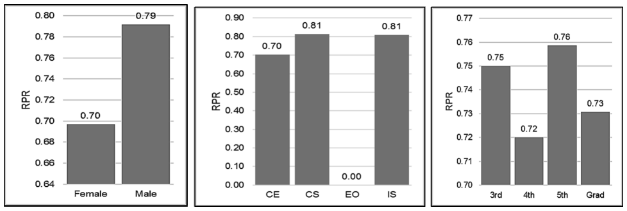


Fig. 6 : Demographics influence on Question 4. Left: Female/male, mid: major and right: college year.

5.2 Results of category 3

(Game usability) Figure 7 shows two parts, Left - detailed RPRs for questions 5-10 and the average RPRs on the right. It can be observed that male respondents find the game usability higher than the female respondents with an average RPR of 77% t o71% respectively.

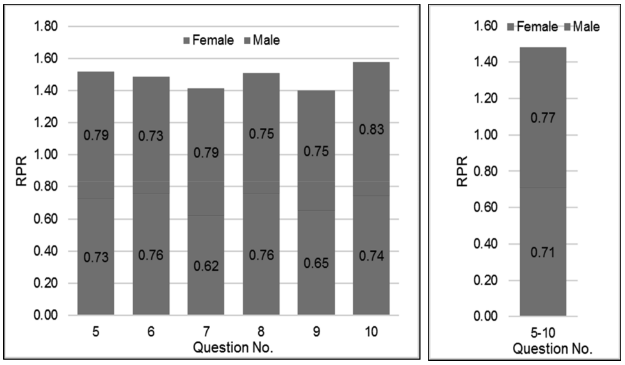


Fig. 7: Female/Male responses for Category 3 (Game usability). Left: detailed RPR by question No., right: averaged RPR of all 5-10 questions.

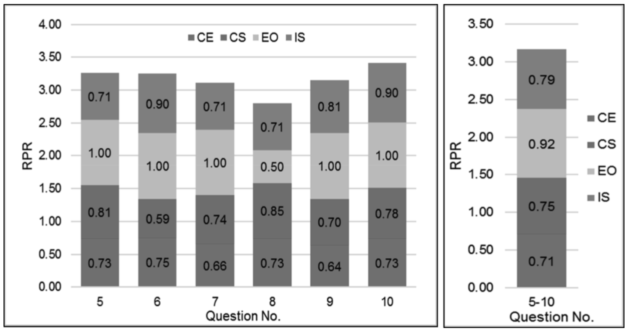


Fig. 8 : Responses based on major for Category 3 (Game usability). Left: detailed RPR by question No., right: averaged RPR of all 5-10 questions.

5.3 Results of category 4 (Game effectiveness in student learning)

The responses based on major for "Game usability" is shown in Figure 8. It is clear that major "Engineering other" students have the highest level of positive feedback at 92% while other majors range between 71-79%. Figure 9 shows Game usability responses based on college year. Which shows that the highest level of RPR is among students who are in 4th year at 79%. This could be due to senior students having more experience and still fresh with how the game is played.

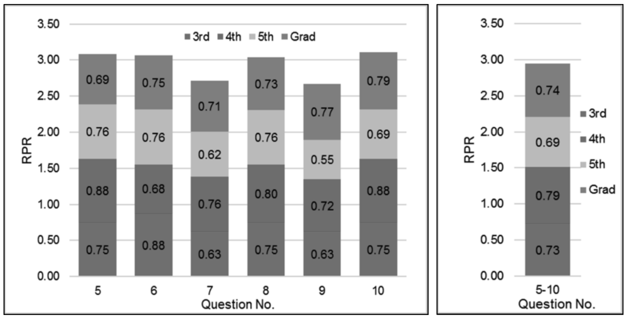


Fig. 9 : Responses based on college year for Category 3 (Game usability). Left: detailed RPR by question No., right: averaged RPR of all 5-10 questions.

The results of category 4 (Game effectiveness) based on Male/Female responses are depicted in Figure 10 for detailed and averaged RPRs left and right charts respectively. The Figure shows that there is no significant difference between female and male RPRs at 84% and 83% respectively.

The responses based on major for category 4 are shown in Figure 11 which has a left chart that shows detailed RPRs per question and a right chart that shows the averaged RPR for questions of this category. It is observed that all RPRs are satisfactory and so far, the highest compared to other categories ranging from 80% - 100%. This category is a key element in the study as the main question of the study is to investigate the effectiveness of GBL in our class of interest – Software Engineering.

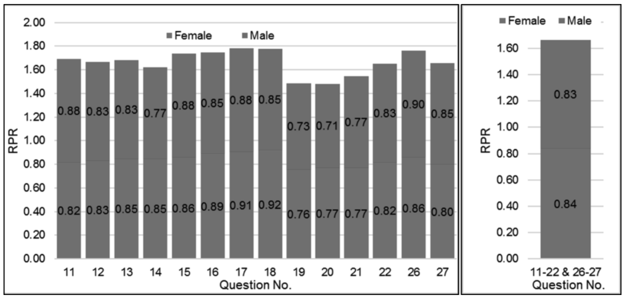


Fig. 10 : Responses based on Male/Female for Category 4 (Game effectiveness). Left: detailed RPR by question No., right: averaged RPR of all 11-22&26-27 questions.

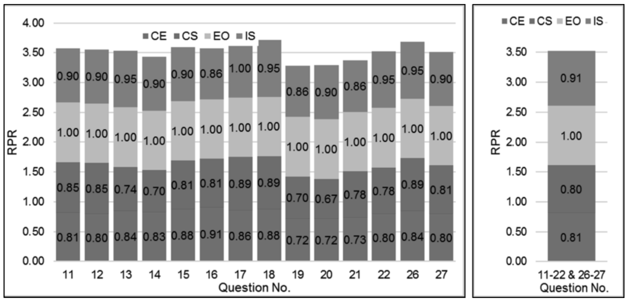


Fig. 11 : Responses based on Major for Category 4 (Game effectiveness). Left: detailed RPR by question No., right: averaged RPR of all 11-22&26-27 questions.

The responses on category 4 based on college year is shown in Figure 12 with chart on the left illustrating detailed RPRs of individual questions and the one on the right illustrating the average RPRs of the questions of this category. It can be seen that students in the 4th year have the highest RPR at 87% compared to other college years.

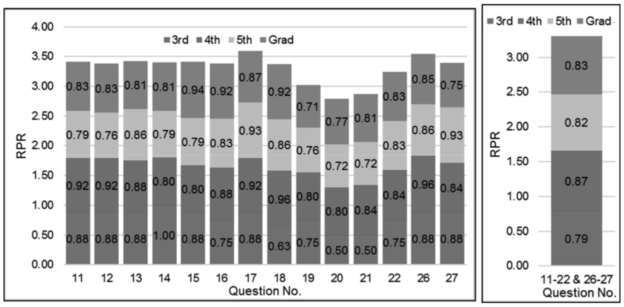


Fig. 12 : Responses based on College year for Category 4 (Game effectiveness). Left: detailed RPR by question No., right: averaged RPR of all 11-22&26-27 questions.

5.4 Results of category 5 (Class structure supporting the Game activities)

The results of category 5 based on female/male responses are shown in Figure 13 with left chart depicting detailed RPRs and right chart depicting averaged RPRs for questions of this category. It is observed that RPRs for both female and male have no significant difference and both are satisfactory at 84% and 83% respectively.

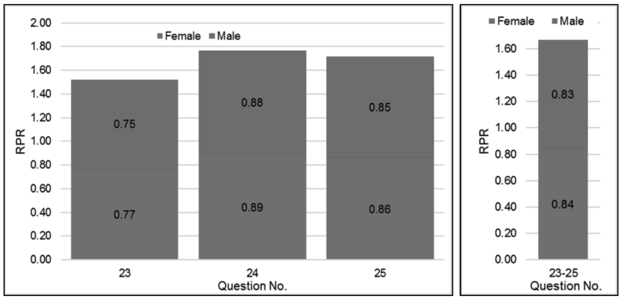


Fig. 13 : Responses based on Female/male for Category 5 (Class structure). Left: detailed RPR by question No., right: averaged RPR of all 23-25 questions.

Results of category 5 based on student major are presented in Figure 14 that shows a left chart for detailed RPRs and a right chart for the averages of all questions in this category. It can be seen that RPRs of majors of CE, CS and IS (at 80%, 91% and 87% respectively) are satisfactory and relatively much higher than that of major EO at 67%.

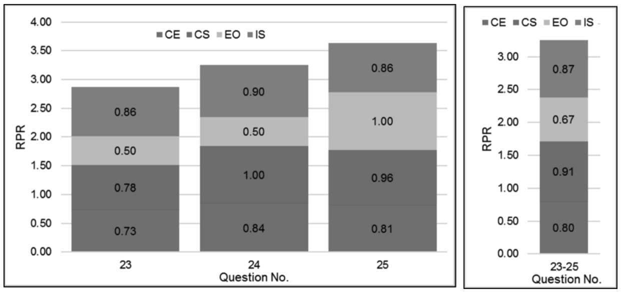


Fig. 14 : Responses based on major for Category 5 (Class structure). Left: detailed RPR by question No., right: averaged RPR of all 23-25 questions.

The results of category 5 based on college year are shown in Figure 15. The left chart shows detailed RPRs for individual questions in the category while the chart on the right shows the average RPRs of all questions in this category. It can be seen that all RPRs for this category based on all college years are satisfactory ranging between 79% - 89%. The result of this category is a good feedback to us as instructors of the class that confirms the supportive structure of the class to the GBL learning component.

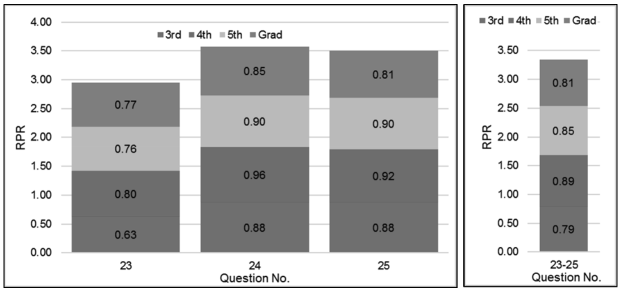


Fig. 15 : Responses based on college year for Category 5 (Class structure). Left: detailed RPR by question No., right: averaged RPR of all 23-25 questions.

5.5Results of category 6 (Student final conclusion and future recommendation) The results of category 6 based on female/male responses are shown in Figure 16 where, the left chart depicts detailed RPRs of the questions and the right depicts the average RPRs of the questions of this category.

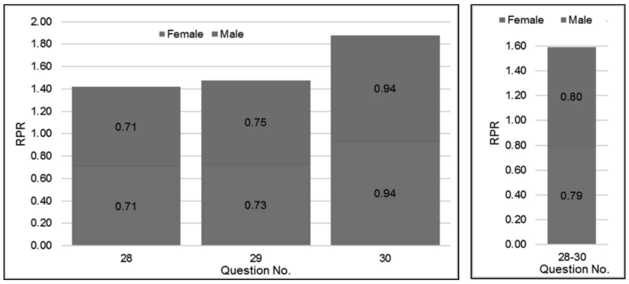


Fig. 16 : Responses based on female/male for Category 6 (conclusion and future recommendation). Left: detailed RPR by question No., right: averaged RPR of all 28-30

The results in Figure 16 shows that the RPRs for both genders are satisfactory and there is no significant difference between them at 79%-80% respectively.

The results of category 6 based on student majors are presented in Figure 17. The left chart shows detailed RPRs per question and the chart on the left shows the average RPRs of all questions per major. It can be seen that RPRs ranges between 74%-100%. This shows that students appreciate the GBL concept, enjoyed it and would recommend it for other classes. Although the averages are all satisfactory but it can be observed that major "EO" has an impressive RPR of 100% while CE major showed the lowest RPR at 74%.

The results of category 6 based on college year are presented in Figure 18. The Figure shows a left chart with detailed RPRs per question and a right chart showing the average of all questions per college year.

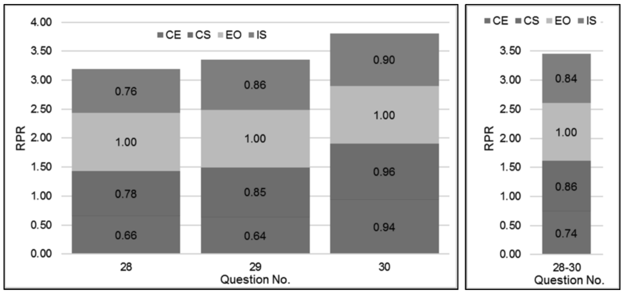
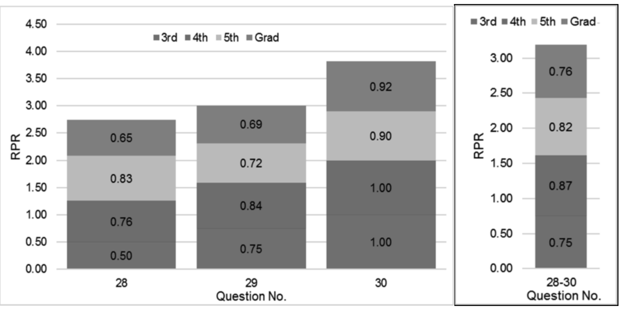


Fig. 17 : Responses based on major for Category 6 (conclusion and future recommendation). Left: detailed RPR by question No., right: averaged RPR of all 28-30.



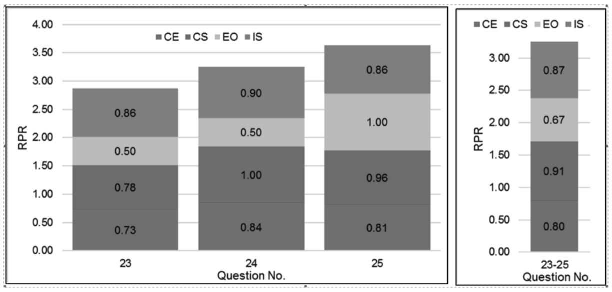


Fig. 18 : Responses based on college year for Category 6 (conclusion and future recommendation). Left: detailed RPR by question No., right: averaged RPR of all 28-30.

It can be seen that all RPRs are satisfactory ranging between 76%-87% with the highest RPR shown by students of 4th year and the lowest shown by students of 3rd year.

Since questions 5-30 are designed with Likert scale, it is of interest to us to see how the aggregated results of all questions in terms of RPR. Questions 5-30 plotted in Figure 19 below. Please note that: blue=strongly agree, red=agree, yellow=neutral, green=disagree and orange=strongly disagree. It is can be observed that the two positive responses strongly-agree and agree dominate the chart.

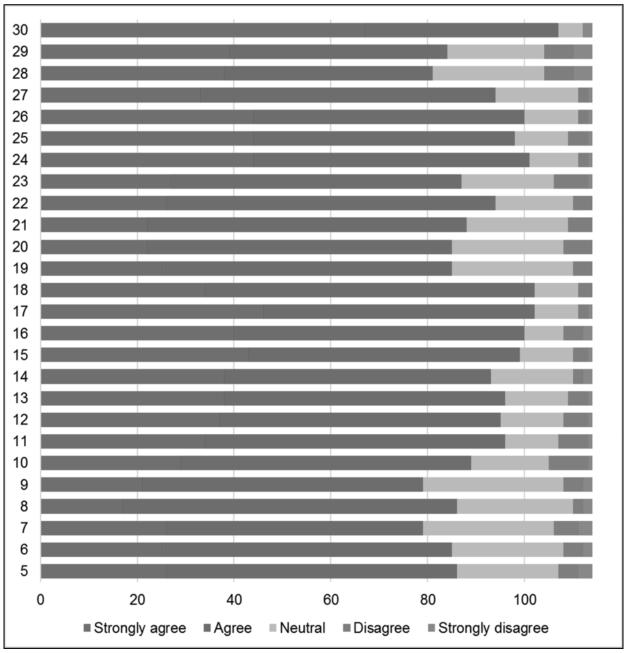


Fig. 19 : Graphical representation of Questions results 5-30: dark green=strongly agree, light green=agree, yellow=neutral, orange=disagree and red=strongly disagree.

To display the questions according to their RPR in an ordered fashion, the data of total positive responses ratios RPRs are plotted in Figure 20 below. Please note that questions appear on the x-axis according to their descending order of positive response and not according to their natural sequence as they were sorted to make it convenient to compare between all questions at a glance. We computed four statistical metrics of the total positive percentages as follows:

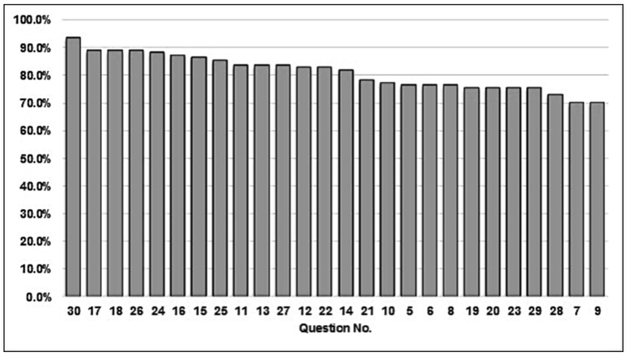


Fig. 20 : Percentage of positive responses (agree + strongly agree) vs. question number. Note that questions appear on the x-axis according to their descending order of positive response.

Minimum = 70.3%, Maximum = 93.7%, Mean = 81.2% and standard deviation = 6.4%.

Based on these results, we can infer that responses from the conducted survey were positive with an average of 81.2% and relatively low standard deviation of 6.4% that indicates consistency in the responses. These positive results indeed confirm the literature we have reviewed and discussed in Section 1. Furthermore, the results of the study are encouraging to keep gaming as a complementary component of our SE course and may be promote it in other computing courses.

5. Conclusion

We have presented our efforts and experience in adopting gaming technology in our Software Engineering (SE) course. We found that the literature in Game-based Learning (GBL) is supportive and encouraging to use computer games in education in general and in SE. We adopted a computer game called SimSE (Navarro, 2004) which is created specifically to support SE education. Although the Game supports different elements of SE course structures such as: software (SW) process models, requirements engineering, system design, implementation, testing, evolution, project planning and project management but, our investigation showed that there are three main elements that the Game effectively supports and provides good opportunity for students to improve their skills in them these are: process models, system testing and project management. In our SE, students have to go through enough material to understand SE concepts before they can play the game. Students play in class and off class several scenarios according to the identified SE process models we assign them. These process models are all supported in the Game and they are: Waterfall, Incremental, Prototype, Extreme Programming, Inspection and Rational Unified Process models. Our experience with SimSE has been positive and rewarding to our students. SimSE supports many educational opportunities as we discussed in details in section 2, 3 & 4. Critical thinking is enabled in several places in the Game as students are required to stay alert throughout the game, analyze situations, evaluate alternative and take informed decisions. We created a student feedback survey to investigate their experiences in GBL in SE course. We collected 114 responses from 8 consecutive semesters from Spring 2016 through Fall 2019. The questioner of the survey was designed to include 6 different categories: demographics, confidence in own major, game usability, game effectiveness in student learning, class structure supporting the Game activities and student final conclusion and future recommendation. The results of all these categories were examined using the demographics category to understand the influence of gender, college year and major on the results of all the 5 other categories in the survey. It was found that the results of all of the categories were positive and encouraging. As an attempt to see the entire survey at a glance including all questions regardless of the category, all the data was combined and plotted and basic statistical metrics were computed. The mean of positive response = 81.2% and the standard deviation of promotive responses = 6.1%. A positive result of 81.2% is encouraging to us to keep GBL as a component in our course and try to promote it in other computing course as well. Finally, we would like SimSE to be updated with more recent developments in the area of Agile SW development such as Scrum and Kanban (Shamshurin & Saltz, 2019; Saleh et al., 2019) as it would help students to practice these contemporary models in the game as well and it would add a good value to the game.

References

[1] Afzal, Hasan (2010). A Study of University Students' Motivation and Its Relationship with Their Academic Performance. International Journal of Business and Management Vol. 5, No. 4, pp. 80-88.

[2] Rababaah, A and Rabaa'I A (2017) 'Utilization of Robotics as Contemporary Technology and an Effective Tool in Teaching Computer Programming', Modern Management and Technology Institute (MTMI), Virginia Beach, VA, USA, Sep 2017.

[3] Liu, Allison S., Schunn, Christian D., Flot, Jesse & Shoop, Robin (2013) 'Computer Science Education (2013): The role of physicality in rich programming environments', Computer Science Education, DOI: 10.1080/08993408.2013.847165

[4] Rababaah, Aaron R. & Rabaa'I, Ahmad A. (2018) Enhancing Programming Learning Environment with Physical Computing And Robotics: A Case Study of the American University of Kuwait, Int. J. Teaching and Case Studies, Vol. 9, No. 4, 2018. Pp. 323-346.

[5] Eguchi, Amy (2014) 'Robotics as a Learning Tool for Educational Transformation', Proceedings of

4th International Workshop Teaching Robotics, Teaching with Robotics & 5th International Conference Robotics in Education, Padova (Italy) July 18, 2014, pp. 27-34.

[6] Tomoko Yoshida, Toshiyuki Kamada, Ryota Nakamura and Toshio Matsuura (2015) 'Development and Use of a Programming Environment for Learning the Mechanism of Measurement and Control by Programs', proceeding of The Society of Digital Information and Wireless Communications (SDIWC) conference, September 2015.

[7] Usher, A (2019, April). Student Motivation—An Overlooked Piece of School Reform, Retrieved from https://files.eric.ed.gov/fulltext/ED532666.pdf.

[8] Gozcu, E & Caganaga C. (2016). The importance of using games in EFL classrooms. Cypriot Journal of Educational Science. 11(3), pp. 126-135.

[9] Özüorçun, Nilcan Çiftçi and Bicen, Huseyin (2017) 'Does the Inclusion of Robots Affect Engineering Students Achievement in Computer Programming Courses?', EURASIA Journal of Mathematics Science and Technology Education, July 2017.

[10] Papadakis, S. (2018) 'The use of computer games in classroom environment', Int. J. Teaching and Case Studies, Vol. 9, No. 1, pp.1–25.

[11] Stathakis, Rebekah (2019, April). Reasons to Use Games in the Classroom. Retrieved from https://www.educationworld.com/a_curr/reasons-to-play-games-in-the-classroom.shtml.

[12] IEAB (2019, April). Learning in the 21st Century: Teaching Today's Students on Their Terms. Retrieved from https://www.certiport.com/Portal/Common/DocumentLibrary/IEAB_Whitepaper040808.pdf

[13] Koivisto, J. & Hamari J. (2019). The rise of motivational information systems: A review of gamification research. International Journal of Information Management, Volume 45(1), pp. 191–210.

[14] Greipl, Simon; Moeller, Korbinian & Ninaus, Manuel. (2020). Potential and Limits of Game-Based Learning. Int. J. Technology Enhanced Learning. Vol.12 No.1, pp. 1-20.

[15] Bahadoorsingh, Sanjay; Ronald Dyer, Chandrabhan Sharma (2016). Integrating serious games into the engineering curriculum - a game-based learning approach to power systems analysis. International Journal of Computational Vision and Robotics. Vol. 6, Issue 3, pp. 276-289.

[16] IBM (2020, April). Invo8 Game: CityOne Game. Retrieved from https://www.ibm.com/developerworks/library/ws-bpm-innov8/.

[17] Al-Sharafat, Safa & AbuSeileek, Farhan (2012). The effectiveness of vocabulary learning website games on English language learners' communication skills. International Journal of Learning Technology. Vol. 7, Issue 2, pp. 192-211.

[18] Minovic, Miroslav; Milos Milovanovic; Dusan Starcevic; Mladan Jovanovic (2010). Learning objects in educational games. International Journal of Technology Enhanced Learning. Vol. 2, Issue 4, pp. 336-346.

[19] Tsai, Chia Hui, Ching-Hsue Cheng, Duen-Yian Yeh, Shih-Yun Lin (2017). Satisfaction of high school students with a mobile game-based English learning system. International Journal of Mobile Learning and Organization. Vol. 11, Issue 2, pp. 131-154.

[20] Dreyer, Adriaan M.F., Nicole Dodd, Wayne O. Dalton (2019). Applying game-based learning at the South African Military Academy: an experimental study. International Journal of Technology Enhanced Learning. Vol. 11, Issue 4, pp. 380-397.

[21] Fellnhofer, Katharina (2018). Game-based entrepreneurship education: impact on attitudes, behaviours and intentions. World Review of Entrepreneurship, Management and Sustainable Development. Vol. 14, Issue 1-2, pp. 205-228

[22] Thomas, Connolly (2007). An application of games-based learning within software

engineering. British Journal of Educational Technology - BJET 38 (2007): 416-428.

[23] Campolina, Pedro; Maurício Souza; Eduardo Figueiredo (2018). Games and Gamification in Software Engineering Education: A Survey with Educators. Proceedings of Frontiers in Education. October 2018, San Jose, California, USA.

[24] Taran, G. (2007). Using Games in Software Engineering Education to Teach Risk Management. 20th Conference on Software Engineering Education & Training (CSEET'07), 211-220.

[25] Claypool, Kajal T. & Mark Claypool (2005). Teaching software engineering through game design. ACM SIGCSE Bulletin. 37(3):123-127.

[26] Navarro, Emily & Andre Hoek (2004). SIMSE: An Interactive Simulation Game for Software Engineering Education. Proceedings of the 7th IASTED International Conference on Computers and Advanced Technology in Education, August 16-18, 2004, Kauai, Hawaii, USA.

[27] SIMSE (2020, April). Simulation Game for Software Engineering Education. Retrieved from https://www.ics.uci.edu/~emilyo/SimSE/.

[28] Connolly T.M., Stansfield M., Hainey T. (2008) Using Games-Based Learning to Teach Software Engineering. In: Filipe J., Cordeiro J. (eds) Web Information Systems and Technologies. WEBIST 2007. Lecture Notes in Business Information Processing, vol 8. Springer, Berlin, Heidelberg.

[29] Ye, En; Chang Liu & Jennifer A. Polack-Wahl (). Enhancing Software Engineering Education Using Teaching Aids in 3-D Online Virtual Worlds. Proceeding of ASEE/IEEE Frontiers in Education, October 10 – 13, 2007, Milwaukee, WI, USA.

[30] Sommerville, Ian. (2016). Software Engineering. ISBN: 978-1-292-09613-1. Pearson, Essex CM20 2JE, England.

[31] Shelly, Gary B. & Rosenblatt, Harry J. (2011). Systems Analysis and Design. ISBN: 978-0-538-48161-8. Course Technology Press, 25 Thompson Pl., Boston, MA, United States.

[32] IBM (2020, April). Rational Unified Process - Best Practices for Software Development Teams. Retrieved from https://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf

[33] OMG (2020, April). Unified Modeling Language for Software Engineering. Retrieved from https://www.omg.org/spec/UML/2.5.1/PDF

[34] Navarro, Emily (2020, April). SimSE game player's manual. Retrieved from https://www.ics.uci.edu/~emilyo/teaching/info43f2014/misc/SimSEPlayersManual.pdf

[35] Pressman, Roger S. & Bruce Maxim (2014). Software Engineering: A Practitioner's Approach. ISBN: 978-0078022128. McGraw-Hill, 2 Pen Plaza, New York, NY, USA.

[36] Stephens, Rod (2015). Beginning Software Engineering. ISBN: 978-1118969144. Wrox, Birmingham, England.

[37] Google (2020, April). Google survey service. Retrieved from https://www.google.com/forms/about/

[38] Shamshurin, Ivan & Saltz, Jeffrey S. (2019). Using a coach to improve team performance when the team uses a Kanban process methodology. International Journal of Information Systems and Project Management, Vol. 7, No. 2, 2019, 61-77.

[39] Saleh, Sabbir M.; Syed, Maruful & Mohammed Ashikur (2019). Comparative Study within Scrum, Kanban, XP Focused on Their Practices. International Conference on Electrical, Computer and Communication Engineering (ECCE). April, 2019, Cox's Bazar, Bangladesh.

Appendixes

Appendix A – Student Survey (Questions)

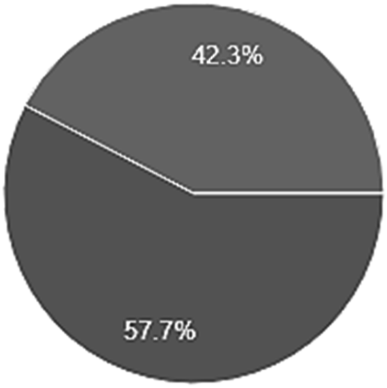
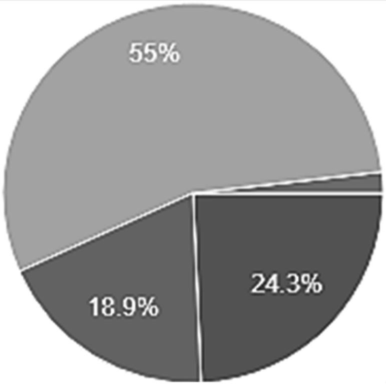
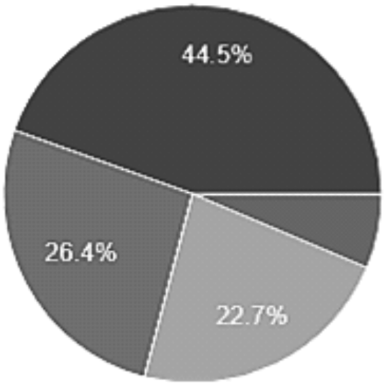
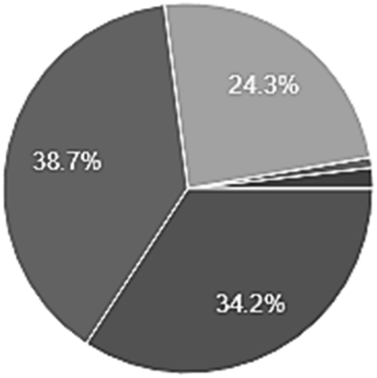
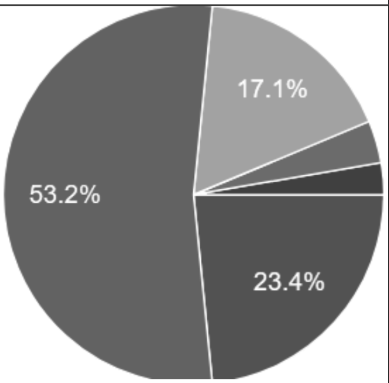
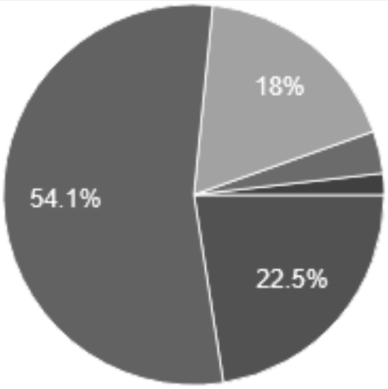
1. Gender M/F
2. My major is: Computer Science, Information Systems, Computer Engineering, Others.
3. I am in my ... year: 2nd, 3rd, 4th, 5th, Graduated.
4. I believe I am in the right major
5. CSIS 330 - Software Engineering gave me my first experience to learn via video games
6. The Game interface is friendly
7. The Game was easy to learn
8. The Game interaction time was efficient
9. The Game process was flexible
10. The Game Interface contains relevant and helpful information
11. The Game helped me to understand Software Engineering ACTIVITIES
12. The Game helped me to understand Software Engineering PROCESS MODELS
13. The Game helped me appreciate the NEED for Software Engineering
14. Team work was possible with the Game
15. The Game helped me appreciate the need for PROJECT MANAGEMENT
16. The Game helped me appreciate PROJECT RESOURCES
17. The Game helped me realize consequences of my decisions in Software Engineering
18. The Game included skills required in Software Engineering and Project Management
19. The Game input VARIABLES made sense
20. The Game RESULTS made sense
21. The Game SCORING CRITERIA made sense
22. The Game was capable of simulating several Software Engineering process models
23. I learned enough theory of Software Engineering before I practice the Game
24. The Game assignments were graded and contributed to my class grade
25. I was informed that Games will contribute to my class grade
26. The Game is suitable for Software Engineering topics
27. The Game made me critically think and it was not trivial
28. I Like the Game
29. I would recommend the Game for other students
30. I would recommend Educational Gaming for other classes

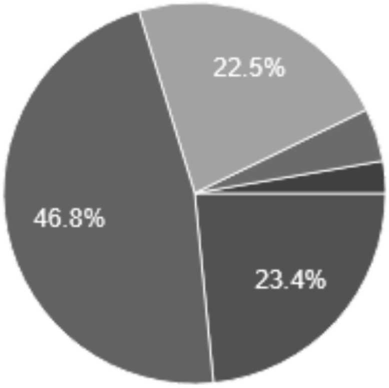
Appendix B – Student Survey (Detailed Responses)

Key for the Response Charts

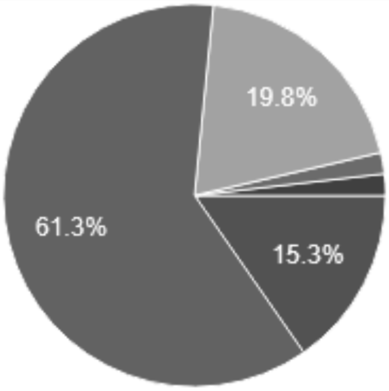
● Female ● Male Legend for Question 1	● Computer Science ● Information Systems ● Computer Engineering ● Engineering (Other) Legend for Question 2
● 2nd ● 3rd ● 4rth ● 5th ● Already graduated Legend for Question 3	● Strongly agree ● Agree ● Neutral ● Disagree ● Strongly disagree Legend for Questions 4-30

Individual question charts

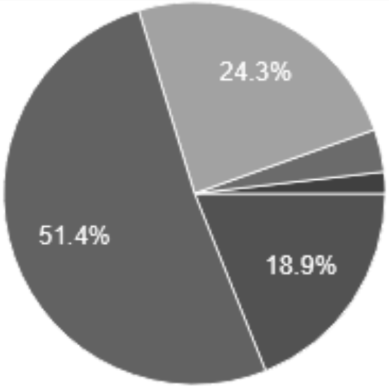
 Question 1	 Question 2	 Question 3
 Question 4	 Question 5	 Question 6



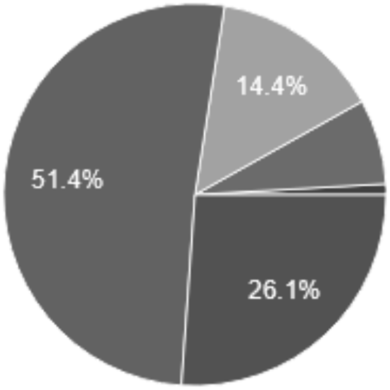
Question 7



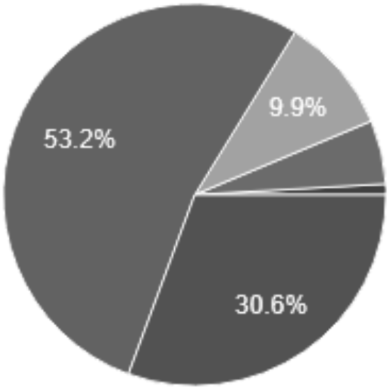
Question 8



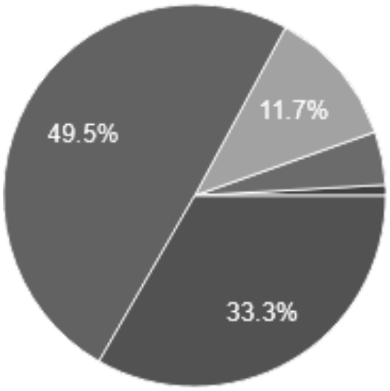
Question 9



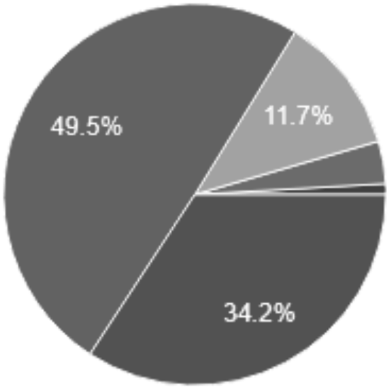
Question 10



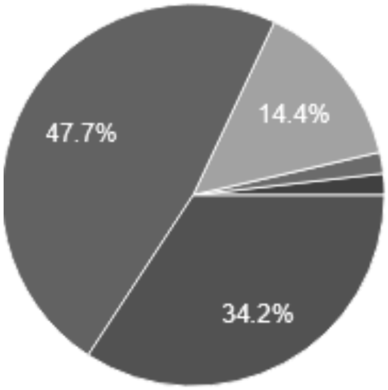
Question 11



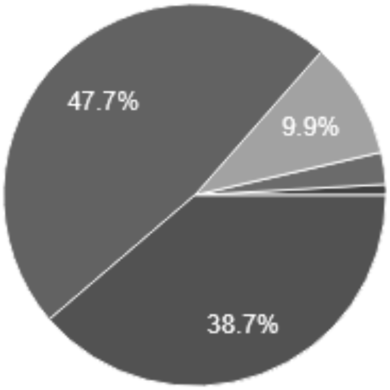
Question 12



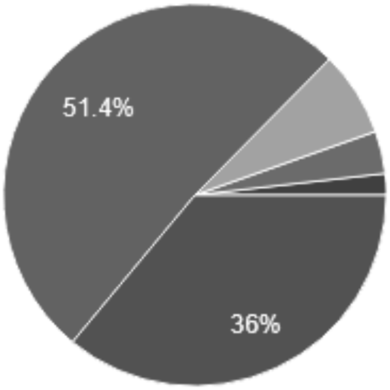
Question 13



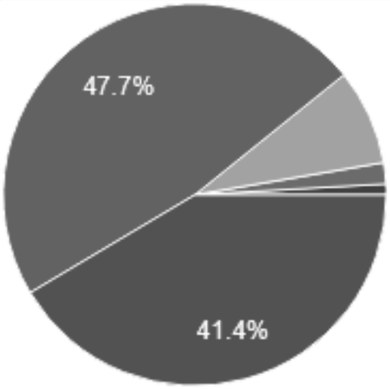
Question 14



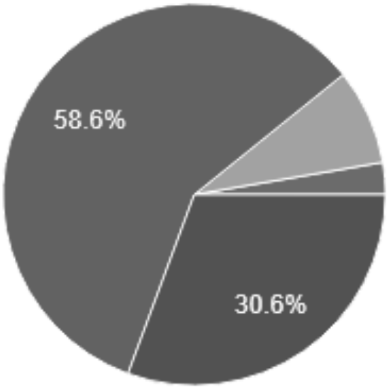
Question 15



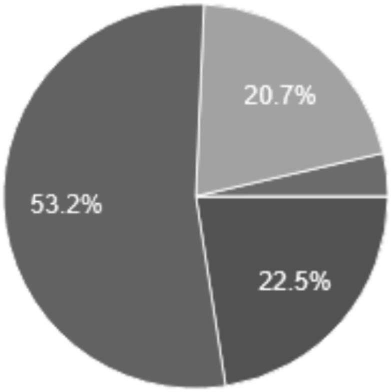
Question 16



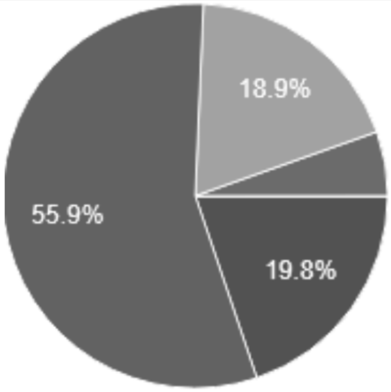
Question 17



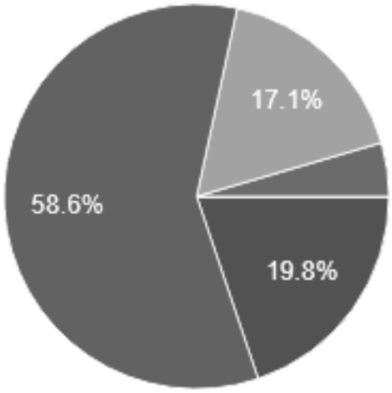
Question 18



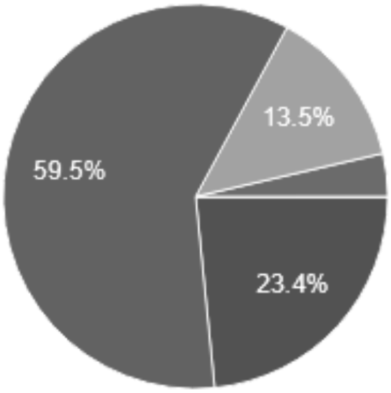
Question 19



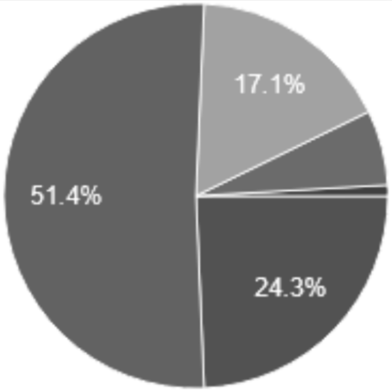
Question 20



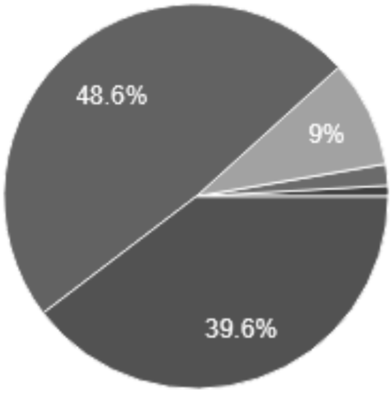
Question 21



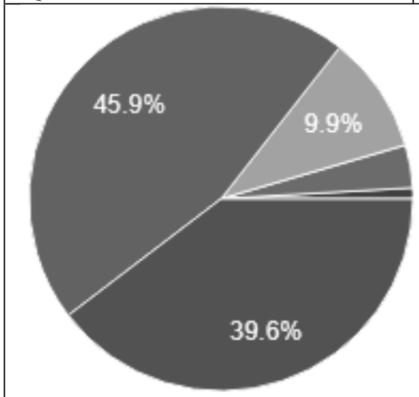
Question 22



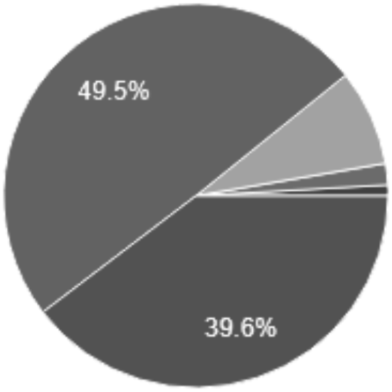
Question 23



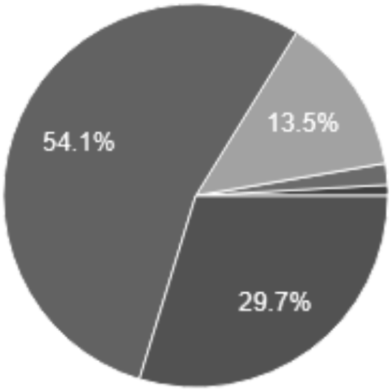
Question 24



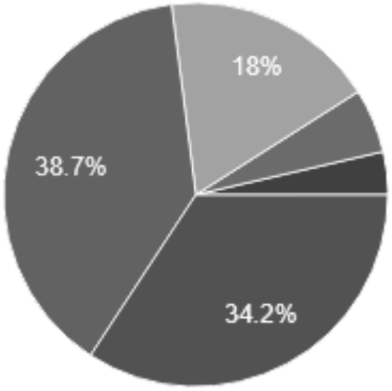
Question 25



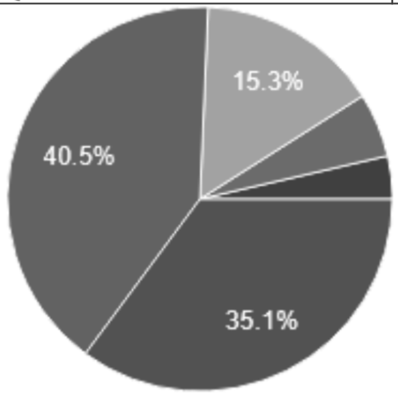
Question 26



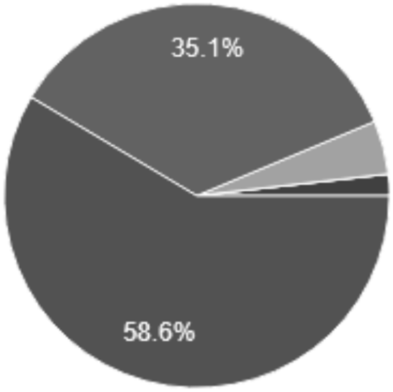
Question 27



Question 28



Question 29



Question 30