JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 39, 999-1022 (2023)
DOI: 10.6688/JISE.202309_39(5).0001

An Educational Card Game Approach to Motivating the Learning of Software Engineering

BAO-AN NGUYEN[2,+], HOANG-THANH DUONG[1],
LING-LING TSAO[1] AND HSI-MIN CHEN[1]
[1]*Department of Information Engineering and Computer Science*
Feng Chia University
Taichung, 407 Taiwan
E-mail: thanh.duongrsm@gmail.com; hsiminc@fcu.edu.tw; eva5784891@mail.fcu.edu.tw
[2]*Department of Information Technology*
Tra Vinh University
Tra Vinh, 8700 Viet Nam
E-mail: annb@tvu.edu.vn

With the rapid expansion of the software sector in recent decades, companies' standards for new employees become more stringent as well. Specifically, they are often unsatisfied with the insufficient competence of students in handling complex assignments in the software development process. To address these issues as well as to help students become acquainted with the actual development process in software engineering, we developed a card game that simulates concepts, roles, and tasks of the actual scenarios for software engineering education. To test the effectiveness of the game, we experimented with two groups of 42 students and measure the results using a post-test and a post-questionnaire. Experimental results show that our approach increased students' learning motivation and help students better understand knowledge in software engineering lessons. These potential results make a call for the use of game-based learning in software engineering education to increase students' learning engagement and outcomes.

Keywords: software engineering education, educational games, gamification, UTAUT, card games, learning analytics

1. INTRODUCTION

The software industry plays a vital role in today's digital era, especially during and after the Covid-19 pandemic. With the continuous development of software engineering, reducing the gap between the academic environment and the industrial environment in software engineering becomes more and more important [1]. As the range of relevant content continues to expand and the complexity of the curriculum grows, it has become increasingly difficult to reconcile the relevant knowledge to be taught in academic institutions. Therefore, both educators and industrial enterprises believe that students should not only be taught on paper or in lectures but also need to engage with the problems, they will encounter in the real workplace and the actual development process [2]. However, the industry is generally dissatisfied with the level of preparation of new students entering the workplace [3]. To get them on track quickly, companies often need to pay additional costs to retrain the newcomers with knowledge and skills that were not fulfilled at school.

The main cause of these problems seems to be the way software engineering is taught today. It is insufficient for software engineering topics to be conveyed in traditional lec-

Received September 30, 2022; revised November 14, 2022; accepted December 22, 2022.
Communicated by YungYu Zhuang.

tures with abstract concepts that are difficult for students to comprehend. Course instructors often introduce theories and concepts in lectures and textbooks, and require students to meet certain requirements, complete small projects according to a specific framework or process, and try to apply what they learn in lectures or discover new tools in practice. Practical and core knowledge is necessary for software engineers, however, the fact that teachers impose knowledge on students may cause students active learning skills [4]. Consequently, students may rely too much on their instructors and may not think by themselves when any problem arises, thus limiting their ability to take the initiative in solving problems. Meanwhile, this approach may raise students' passive and negative attitudes toward learning and even a sense of timidity or disgust.

Among non-traditional education approaches, game-based learning or gamification is employed to arouse students' interest in learning, alleviate their anxiety in learning, enhance students' learning motivation, and help them better understand the teaching objectives [5-7]. In this study, we propose a card game for software engineering teaching, which combines related knowledge of an actual software development process into a card game. The game simulates concepts, roles, and scenarios encountered by different members at different positions in the iterative development process. To analyze the benefits and ideas of the students in using this card game, the following research questions were posed:

RQ1: Is the card game help students improve their understanding of the software development process?
RQ2: Does adding the card game to support a traditional course affect students' learning motivation in software engineering?
RQ3: What are the advantages and disadvantages of this game card for teaching software engineering?

To address the research questions, we conducted an experiment on two groups of students, proposed a set of hypotheses based on the modified and extended version of the Unified Theory of Acceptance and Use of Technology (UTAUT2) [8] model, and used the PLS-SEM model [9] to test the hypotheses. The remainder of the paper is structured as follows. In section 2, we discuss recent studies applying educational games and their related hypothesis model. In Section 3, we introduce our card game for software engineering courses. The experiment and result are presented in Section 4. Implications and a brief conclusion are presented at the end of the paper.

2. RELATED WORK

2.1 Research Related to Educational Games

Many educators have substituted traditional teaching methods with game-based learning (GBL) to motivate software engineering students during the learning process [10, 11]. Card games, with visual features and competition mode, can effectively boost students' engagement in learning-by-playing activities. Sung *et al.* [13] combined Mindtool to develop a game-based collaborative learning game for experimentation in elementary school science classes to facilitate students to share and organize their knowledge during the game and to examine students' learning attitudes, motivation, *etc.*

With the acknowledgment that the visual appearance of cards can convey software

engineering principles, the author of [14] proposed *Problems and programmers*, a card game in which each student takes on the role of a product manager with a common product goal, and the first one who completes the production project is the winner. To complete the product project, players need to manage their budgets and meet the needs of their customers to produce high-quality software. To facilitate student practice in testing environments, the author of [15] proposed an experimental card game related to software testing. Depending on the content of the card, students can decide whether they want to solve the card by themselves or together. The solution to the task and activity is written on the back of the card, and after the player attempts to solve the task, the group will decide whether the player will receive the reward. Bonus points are recorded on the white sheet, and those who achieve 30 points win.

Card games are usually used to teach the Agile development process in the classroom. Gabriel *et al*. [16] utilized the MEEGA+ model to evaluate students' perceptions of using the Scrum card game to learn Agile methodologies. According to the findings of the analysis, students enjoy card games because of their product benefits. The authors of [17] introduced a new game card that helped learners comprehend vendor evaluation factors. After completing the course, students are be able to select a software development company and determine the cost-to-time ratio. To familiarize students with the risk management process, the authors of [18] proposed a risk game that encouraged students to discuss the hazard and its solutions. In overall, card games are considered as effective tools in learning the software engineering concepts and enhance students' learning motivation. The proposed card-game in this paper is in line with mention studies.

2.2 Theoretical Framework for the Experiment

In 2003 [19], Venkatesh *et al.* validated the UTAUT model to determine the most effective strategies for fostering user behavior and the use of technology for educational advancement. The model utilizes the four primary predictors of intention to use technology Performance Expectancy (PE), Effort Expectancy (EE), Social Influence (SI), and Facilitating Conditions (FC). Studies examining technology adoption in higher education contexts have successfully used the UTAUT model [14, 15]. The UTAUT2 model, an expanded version of UTAUT, was subsequently developed for research in numerous fields. Hedonic Motivation (HM), Price Value (PV), and Habit are three additional constructs that the new model adds (HT). G. Baptista and T. Oliveira used UTAUT2 to demonstrate the impact of gamification on mobile banking and the significant correlation between gamification and action intention to use the service [22]. In several different academic situations, including those involving technical education, engineering, and management, UTAUT2 has also received widespread acceptance and testing [23]. To propose a new model in this study, we used a modified and expanded version of UTAUT2 and added 2 new constructs. As shown in Fig. 10, the new constructs include gamification and active learning.

Gamification (GM) Gamification or game-based learning (GBL) are terms that are frequently used as synonyms to describe the use of games or game elements in education [24]. Gamification has become popular in the educational environment over the last few decades. The benefits of gamification for education were extensively discussed by educators and researchers [19-21]. We incorporated gamification into our software engineering courses to better assist students in learning.

Active Learning (AL) Activities that involve student participation are considered to be active learning. This indicates that the students are engaged in an activity and reflecting on it [4, 22]. The efficacy and relationships between active learning and learning outcome are presented in [23, 24]. Specifically, students will retain more information from the brief activities that follow the lecture.

3. THE SOFTWARE ENGINEERING CARD GAME

3.1 Purpose of the Card Game

This study proposes an educational card game to introduce realistic software practice, which is a combination of real-world concepts software development processes, and knowledge cards. The card game aims to overcome difficulties in presenting knowledge related to software engineering and improve students' motivation and self-study ability [8].
The card game provides a sequential approach to a typical software development process, simulating the iterative process of actual software development and what kinds of problems are encountered by all parties (developers, quality assurance, and operation and maintenance), and how they are solved. Specifically, students can think independently when playing with the cards, for example, what kind of problems they encounter and how to solve them. It is also a good idea to think about what strategies can be used to better solve the problem at hand and to use physical cards to give students more memory points.

3.2 Card Design and Display

The game is designed as a game-based learning approach, not just a game for entertainment, but the game emphasized the educational value brought by competitiveness and fun. It is designed as a competitive game and adopts a one-on-one group battle method of conduct. Simulating the environment of the software development process, students will play as a small team in the company, in which team members have different roles with different abilities. The card game is intended to be played in small teams, so the number of players in each group is limited to three to four. There are 108 cards categorized as project cards, character cards, question cards, and concept cards. Whereas project cards are used for the game initial and character cards determine the player's roles, the teams use question cards as the challenges to the opponents, and concept cards are used as addressing actions. Pairs of question-concepts cards are listed and explained in Appendix 1. Descriptions of the proposed cards are presented below.

Project Cards: Project cards are used to initiate the game. As each project has many iterations, the text under the picture on the card represents the number of rounds to play in each competition, as shown in Fig. 1.

Character Cards: A characters denote a role of a specific person in the project, so all popular job positions might be included in the character cards. Since characters differ in abilities and characteristics, the ability of each role card can only be used once in a game. As shown in Fig. 2, the cards are decorated with three different colors to distinguish between different roles: the orange card represents developers, the green card represents quality assurance engineers, and the blue card represents operations and maintenance workers.

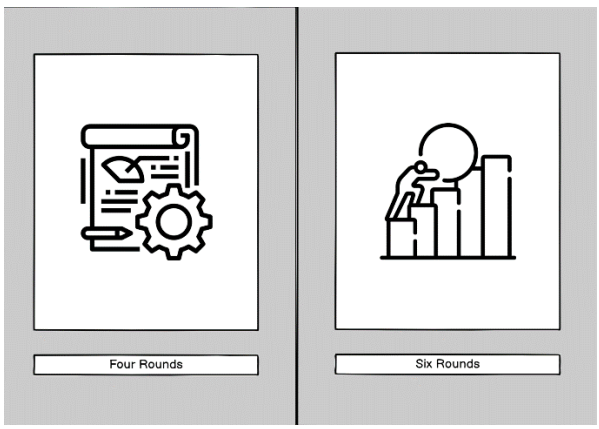

Fig. 1. Project cards.

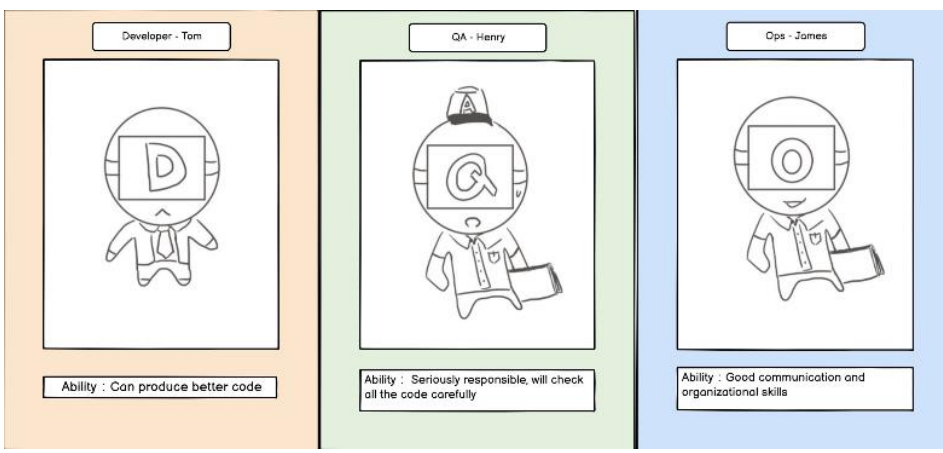

Fig. 2. Character cards.

Question Cards: The card game is played in competition mode where students throw out question cards to the opposing team, and the opposing team thinks about which card they need to play to solve the problem, correspondingly to the current phase of the game. We propose a unique design for the appearance of the card as follows. The top left of the card represents the kind of problems, which are technical problems (T), communication problems (C), and environmental problems (E). The top right corner shows the awarded points for solving the problem. The cards are decorated with pictures illustrating the problem. A description of the problem is shown at the footer of the card. The bottom left corner shows the amount of time it will take to finish the problem. As shown in Fig. 3, question cards are colored in three colors: orange, green, and blue which respectively represent project program problems, test problems, and operation and maintenance problems.

(a) Project Program Card: These cards represent problems related to programs that developers may encounter in the software development process.

(b) Test Card: These cards represent problems that quality assurance engineers will encounter in the software development process.
(c) Operation Card: These cards illustrate problems that operations and maintenance will encounter in the software development process.

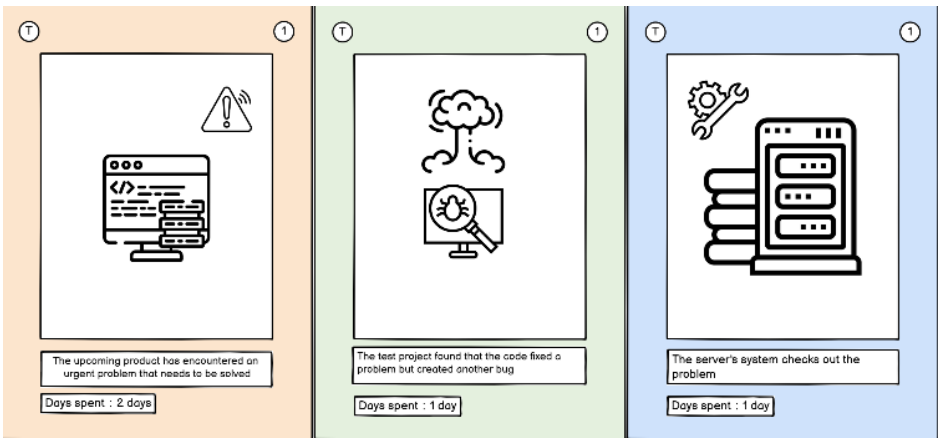

Fig. 3. Project program card, test card, and operation card.

Concept Cards: In the card game, the students will throw the question cards to the opponent's team according to the current phase of the game. The opponent then tries to find the concept cards to solve the problem, according to the problem type and the corresponding phase. As shown in Fig. 4, we use three colors, orange, green, and blue, to distinguish the types of concept cards, which are development, test, and operation, respectively. The picture in the middle of the card is an illustration of the solution, and the text below the picture is a description of the corresponding solution to the concept program card. Accordingly, to phases of the software development process, we propose three categories of concept cards: (i) Development Concept Card; (ii) Test Concept Card; and (iii) Operation Concept Card.

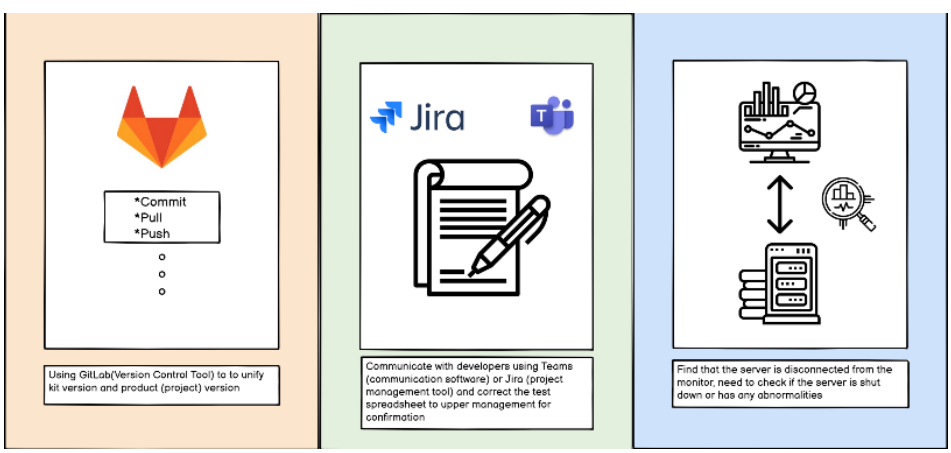

Fig. 4. Examples of development concept card, test concept card, and operation concept card.

3.3 Game Process

The game is played in competition mode so that multiple students can participate in the game as competitors. To be the winner, each team must try to achieve the most awarded points or the least number of days completing the problems. We expect that competition makes students want to compete in the game, and via that, it enhances students' learning motivation. The game flow is shown in Fig. 5 and the steps of how to play the game are shown in Figs. 6 and 7.

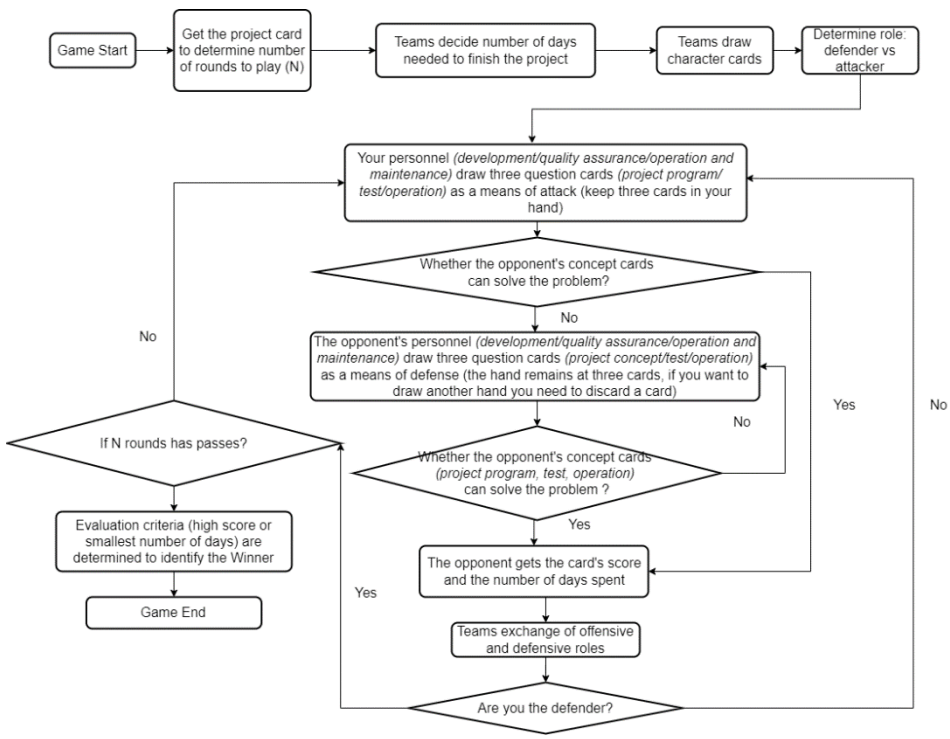

Fig. 5. The card game flow chart.

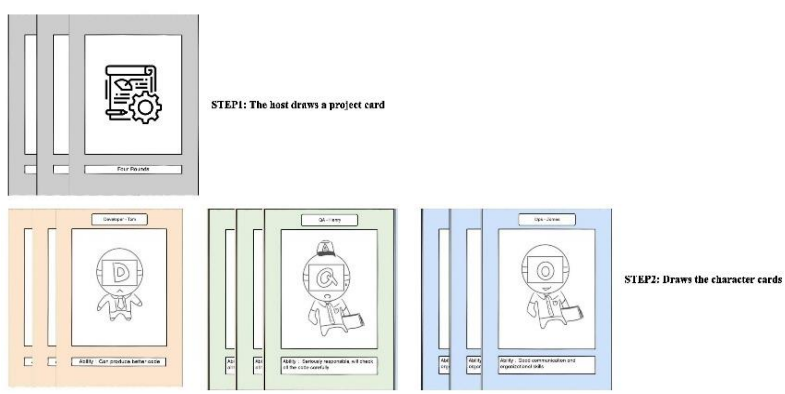

Fig. 6. Card game steps 1-2.

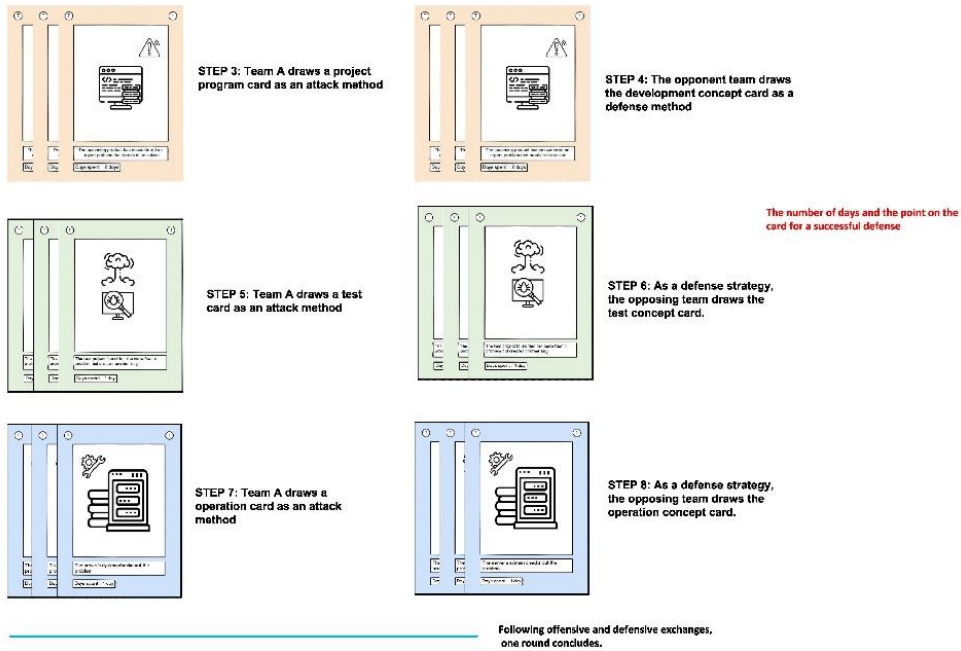

Fig. 7. Card game steps 3-8.

The students were given different roles in the software development process. In this card game, there are three roles: Developer, Quality Assurance Engineer, and Operations and Maintenance, as shown in Fig. 2. Each character card has different skills that can be used in the game, and the skill can only be used once in a game to avoid breaking the game balance too much.

To begin, the host of the card game randomly selects one project card for each team in the match. The project cards show how many rounds the two groups need to play in this game. There are three phases in the software development process, corresponding to three phases of the game: development, QA, and operation.

At each phase, the offensive team tries to attack by throwing three question cards (the cards must relevant to the current phase) to the opponent. The defensive team then tries defense by finding concept cards with characteristics and abilities that can solve the posed problems in the least duration. If they succeed in defense, they can earn the award points for the problem, and the day spent is recorded. Otherwise, no award is given, the teams exchange their roles, and the former defensive team will be the offensive team and vice versa. The team continues playing one more round of attacking defense. Then, the current phase finish, and the game moves to the next phase.

At the finish, the host summarizes the total of awarded points and spent days with each team to determine the winner.

Regarding the design of games, this study adopted the seven principles of game design proposed by Liu *et al.* [31, 32]. Those principles were originally introduced in digital game design [33], then they were utilized by their authors in educational card games later [34, 35]. The seven design principles were transferred into the proposed game as follows:

Analyze learner: Learners are students in software development practice courses, we ensure that they have the same level of knowledge related to software engineering before engaging in the card game, so that, the game was delivered after the students had spent eight weeks of lectures.

Set an educational goal and choose an appropriate game object: The game's objective is to assist students in learning with knowledge derived from the actual software development process, thus, the question cards and concept cards were designed based on popular roles, problems, and actions in software engineering.

Integrate game content with an educational goal: The actual problems encountered by the developer, quality assurance engineer, and operation and maintenance are the main challenges of software teams. How to solve these problems is the main axis to design the card game. The game flow is arranged according to the actual development process so that students can early comprehend actual situations that may appear in software companies.

Education as the purpose and games as the motivation to strengthen students: With the entertainment of the games and their competitive nature, students can be more motivated to learn than in traditional lectures. Since the software development processes are difficult to be mimicked in the classroom, the card games can simulate partly the flow, the problems, and improve students' capabilities in decision-making [35].

Make the most of the game's features: Competitive card games provide students with a sense of competition and motivation for the game itself, allowing them to participate more actively and at the same time increase their motivation for learning. Also, of the physical nature of the card game itself, students can easily memorize specific things in their minds.

Let students enjoy learning while they learn: Using a game-based learning approach allows students to acquire knowledge entertainingly and enjoyably, allowing them to learn new knowledge while having fun during the card game.

Evaluate learning outcomes, and improve teaching methods: Students' responses were observed during the card game and a questionnaire was given to students after the game to assess learning.

This study also refers to the five evaluation indicators of game design proposed by Liu *et al*. [33], which are:

The message of the game: The game is mainly designed for students of software engineering practice courses, learners themselves need to have a certain degree of understanding of software engineering-related knowledge. The sequence of the game process conveys the sequence of the actual software development process and fully conveys the problems encountered in the software development process in different capacities and their solutions. The game graphics are designed to provide pictures according to the problems or solutions. The cards related to the developers, quality assurance engineer, and operation and maintenance each have different colors to facilitate differentiation, and when the operation and maintenance are dealing with server-related problems, the corresponding pictures will have the server pictures.

The game mode: The card games are played in competitive mode to give all students a

sense of participation, and the group battle mode can also enhance everyone's centripetal force, cooperation ability, and sense of competition for the group.

The design structure:

a) Sequential Reasonability: Design the game flow through the actual software development process.
b) Game specificity: The card game promotes interpersonal communication and competitiveness, and the physical nature of the cards makes it easier for students to remember the content.
c) What the game can enhance: To enhance the student's sense of participation in the course and the competitive mindset of group confrontation, to increase students' motivation to learn.

The game content: The card game allows students to learn about the practical application of the software development process. The main content is to learn the process of software development and different identities will encounter different problems and their related solutions, and the card game is played in a turn-based competition.

Feedback: User responses or feedback on the game after playing the card game. Feedback was obtained by using verbal comments and reactions given by students during the game and by conducting a survey of students after the card game.

In this study, knowledge acquired through play was classified into the following categories based on Anderson and Krathwohl's revision of Bloom's taxonomy of educational objectives.

Remember-Factual Knowledge: Students can put themselves in different roles to understand the problems encountered by the developers, quality assurance engineers, or operation and maintenance during the game.

Remember-Conceptual Knowledge: Using the different color markers on the cards makes it easier to remember the problems encountered by the characters in different positions.

Understand:

a) The project program cards are marked with the number of days so that students know the cost of the time it takes to solve the problem when it occurs.
b) Students can understand the sequence of the development process and what problems will be encountered in which capacity from the flow of the cards when playing.
c) The character cards will have personality traits on them, allowing students to discover the advantages of each character when they have different traits while playing.

Apply: The concept cards put real-life encounters into the cards so that students may be able to use the solutions taught in the cards to tackle problems in the development process in the future.

Analyze: It is possible to analyze what kind of problems people face in different capacities so that we can understand the differences in the work content of different capacities.

Evaluate: Evaluate what problem solution is the most beneficial and use the most appropriate solution.

4. THE EXPERIMENT

To verify the applicability of the proposed game in supporting software engineering education, we conducted a quasi-experiment with two groups of students, called the experimental group and the control group. Pre-test and post-test were used to compare the difference between the two groups. At the end of the experiment, a post-questionnaire was delivered to collect responses from students in the experimental group, via that, answers to the research questions can be determined.

4.1 Experimental Environment

The experiment was conducted in the Software Development Practicum course taught at the Department of Computer Science and Information Engineering, Feng Chia University Taiwan in the first semester of the academic year 2021-2022. The 42 students were randomly divided into an experimental group and a control group. All of them were undergraduates between the age of 19-21, including 14 females and 28 males, who are participating in a software development project in their course.

4.2 Experimental Process

The experiment was implemented in three weeks. Before that, the students had studied for 8 weeks with lecturers covering processes, problems encountered and solutions, DevOps, and tools used in software development. We utilized the mid-term exam as the pre-test on software engineering-related contents to determine whether the knowledge of the control group and experimental group was on a uniform benchmark before the experiment. In the following week of the pre-test, the experimental group played the game before the lecture, and the control group studied by lecture as usual. Post-tests are scheduled in the next week of the course to give students time to absorb the knowledge. The detail of the experiment process is presented in Fig. 9.

In the experimental group session, students were first instructed how to play the card game and then randomly divided into groups to play against each other and freely learn and absorb the educational elements of the card game. Fig. 8 shows students playing the card game and discussing it with their group members.

Fig. 8. Students play card games in class.

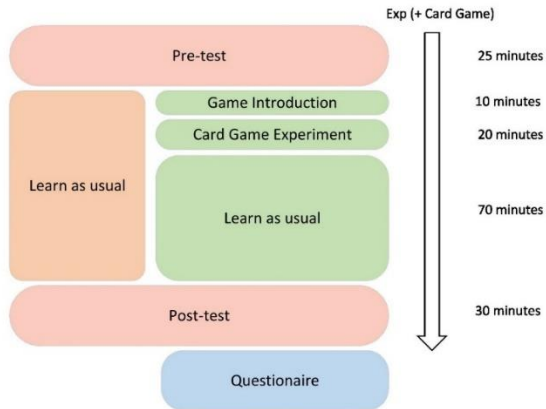

Fig. 9. The flow chart of the experiment.

After the card game, the same sessions were taught. The lesson explained what benefits the card content brings to the learners, with the corresponding physical cards and the experience of playing so that the students can better understand and absorb the lesson content. The post-test lasted for 30 mins with 20 questions covering topics related to the actual process, DevOps, and automation tools.

At the end of the post-test, students in the experimental group were asked to fill out a survey about the card game, with a total of 18 items in five categories. The questionnaire was administered on a five-point Likert scale, with five levels of options: 1 (strongly disagree) to 5 (strongly agree). In addition, there are three open-ended questions about the suggestions of students for the game. The questionnaire content can be found in Appendix 2.

4.3 Data Analysis

The questionnaire was derived from the UTUAT2 model, as shown in Fig. 10, and was adapted and extended by introducing elements such as performance expectancy, he-

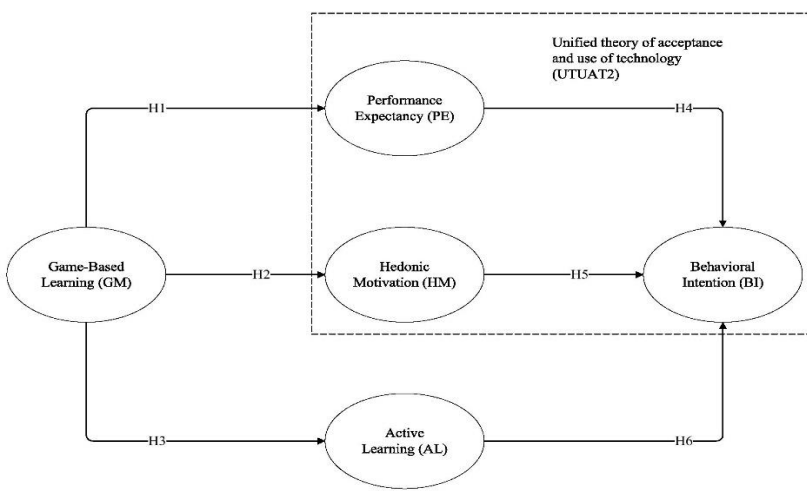

Fig. 10. Questionnaire conceptual framework.

donistic motivation, and behavioral intention. Gamification and active learning were introduced as new elements of the questionnaire. Since structural equation modeling (SEM) can validate multiple relationship hypotheses about observed and latent variables [36] and requires a low sample size [9], which is suitable for the analysis of small sample size in this study, PLS-SEM was used to detect the observed and latent variables by measurement model and structural model.

5. ANALYSIS RESULTS

5.1 Pre-test Result

The pre-test was designed with 15 multiple-choice questions based on the content of the software engineering course to test whether the knowledge of the two student groups is at the same level. Descriptive statistics of the pre-test results are shown in Table 1.

Table 1. Descriptive statistics of the pre-test results.

Variable Grouping	Count	Mean	Median	Minimum	Maximum	Std. dev.
Experimental Group	21	76.9524	80	24	120	21.0867
Control Group	21	76.1905	80	48	120	18.5246

An independent sample *t*-test was used to compare the mean score of the experimental group with that of the control group. The mean test score of 21 samples in the experimental group was 76.952, while that of 21 samples in the control group was 76.19. In the homogeneity test for variance, the value of the test statistic *f* was 1.2957 and the *p*-value was 1.4322, indicating that there was no significant difference between the variance of the two samples. In the independent sample *t*-test, the t-value of the validation statistic was 0.1244 and the *p*-value of the odds ratio was 0.9016. Therefore, the null hypothesis that the mean scores of the two groups are equal could not be rejected. Such data results indicate that there is no significant difference in software engineering knowledge between the experimental group and the control group.

5.2 Post-test Result

In the post-test, we used 15 single-choice questions and 5 multiple-choice questions on the course content and concepts communicated during the experiment. The purpose of this test was to examine whether there were differences between the experiment and control groups in the test. Since some students did not take the test, 36 students' data were available. The descriptive statistics table of the post-test is shown in Table 2.

Table 2. Descriptive statistics of the post-test.

Grouping Variable	Count	Mean	Median	Minimum	Maximum	Std. dev.
Experimental Group	18	90.7222	94.5	69	106	11.6811
Control Group	18	83.7219	89	52	106	15.4036

With the mean values, we can see that the experimental group with the aid of the card game did achieve better scores than the control group, with a mean difference of 7.0003

points, and in the experimental group, the standard deviation of the post-test results was much more stable than in the pre-test. The minimum score was also seen to be higher than that of the pre-test.

5.3 Assessment of Measurement Model: Reliability and Validity Test

We employed PLS to analyze the questionnaire results, with the model shown in Fig. 11. To ensure the validation of analysis results, the reliability, internal consistency, and convergent validity of the questionnaire categories and items must be calculated in advance. If the indicators are highly correlated and exchangeable, they are considered to be reflexive and should be checked for reliability and validity [37]. For this purpose, of our questionnaire, the two items related to social influence (SI-1, SI-2) and one related to gamification (GM-2) were deleted.

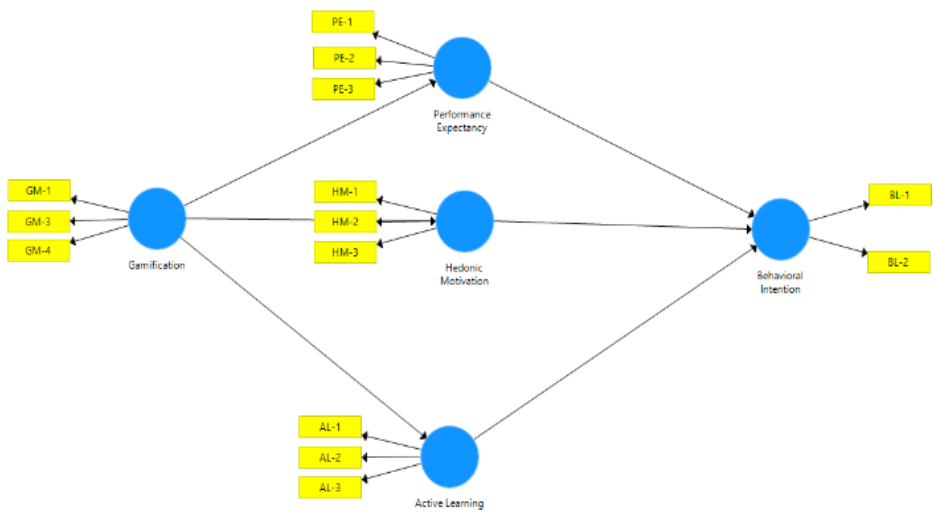

Fig. 11. Questionnaire analysis model.

Table 3. Construct reliability and validity.

	AVE	CR	Cronbach's Alpha	AL	BI	GM	HM	PE
AL	0.796	0.921	0.871	0.892				
BI	0.916	0.956	0.909	0.804	0.957			
GM	0.839	0.939	0.904	0.853	0.915	0.916		
HM	0.712	0.880	0.804	0.583	0.779	0.739	0.844	
PE	0.830	0.936	0.897	0.633	0.534	0.675	0.718	0.911

Note: GM = Gamification; PE = Performance Expectancy; HM = Hedonic Motivation; AL = Active Learning; BI = Behavioral Intentions

According to Ken, the reliability and validity of these indicators are considered acceptable if the Cross-Loading (CL), Cronbach's alpha (α), and Composite Reliability(CR) values are equal or greater than 0.7, and the AVE value is greater than 0.5 [38]. The final analyzed results were presented in Table 3 and Appendix 2. All CL values were greater

than 0.760 and α values were greater than 0.870, which indicated that the reliability of the measurement model is good and all CR values are above 0.880. The average variance extracted (AVE) is greater than 0.871, indicating a high degree of convergence calibration. The next significant step is to observe the discriminant validity of the Fornell-Larcker criterion analysis, in which the results are considered acceptable when the diagonal values are higher than the cross-loading values of the constructs. It means the model can have good discriminant validity indicating the structures has sufficient convergence calibration.

5.5 Analysis Results of the Research Questions

Based on the conceptual framework of the questionnaire and the analysis model, we listed six hypotheses and analyzed them separately.

Hypothesis 1: Gamification has a strong effect on Performance expectancy in software engineering practice courses.

Hypothesis 2: Gamification has a positive effect on students' hedonic motivation toward software engineering practice courses.

Hypothesis 3: Gamification has a significant effect on students' active learning.

Hypothesis 4: Performance expectations have a strong effect on students' behavioral intentions.

Hypothesis 5: Students' hedonic motivation has a positive influence on students' behavioral intentions to use gamification in the software engineering course.

Hypothesis 6: Students' active learning has a significant impact on students' behavioral intentions to use gamification in the software engineering course.

Table 4 shows the results of the research hypotheses. The first hypothesis showed a positive relationship between Gamification and Performance expectancy (β = 0.675; *P*-value = 3.923; *T*-value = 0.00*). which indicated that Gamification has a meaningful impact on Performance expectancy; thus, H1 is supported.

The second hypothesis outline the association between Gamification and Hedonic Motivation (β = 0.739; *P*-value = 4.996; *T*-value = 0.00*) which demonstrates that Gamification has a positive effect on students' hedonic motivation; thus, H2 is supported.

The third hypothesis presented the connection between Gamification and Active Learning (β = 0.853; *P*-value = 11.427; *T*-value = 0.00*) which shows that Gamification has a significant effect on Active learning; thus, H3 is supported.

The fifth revealed a correlation between Hedonic Motivation and Behavioral Intentions (β = 0.648; *P*-value = 2.909; *T*-value = 0.004*) which demonstrated that student's hedonic motivation has a positive influence on students' behavioral intentions to use gamification in the software engineering course; thus, H5 is supported.

The sixth hypothesis shows the relationship between Active Learning and Behavioral Intentions (β = 0.639; *P*-value = 2.987; *T*-value = 0.07*) which indicated that students' active learning has a significant impact on students' behavioral intentions to use gamification in the software engineering course; thus, H6 is supported. As a result, H1, H2, H3, H5, and

H6 are supported and demonstrated that Gamification and Active learning have a signify-cant effect on the student's motivation and behavioral intentions in software engineering class.

In contrast, the relationship between Performance Expectancy and Behavioral Intention is less obvious, and the variable cannot be predicted for Behavioral Intention, so H4 is not supported. From the results, we can see that the educational approach of the card game is indeed effective in influencing motivation and the benefits of learning. This effect is not so obvious compared to the effect of hedonistic motivation and active learning, so H4 is not supported, which shows that the main factor that affects behavioral intention is not the effectiveness of learning but the motivation of learning.

Table 4. Results of testing study hypotheses related to direct effects.

Hypotheses	Relation	β	T-Value	P-Value	Result
H1	GM → PE	0.675	3.923^{***}	0.000	Supported
H2	GM → HM	0.739	4.996^{***}	0.000	Supported
H3	GM → AL	0.853	11.427^{***}	0.000	Supported
H4	PE → BI	–0.336	1.219	0.224	Unsupported
H5	HM → BI	0.648	2.909^{**}	0.004	Supported
H6	AL → BI	0.639	2.987^{**}	0.003	Supported

Note: GM = Gamification; PE = Performance Expectancy; HM = Hedonic Motivation; AL = Active Learning; BI = Behavioral Intentions; $^{*}p < 0.05$; $^{**}p < 0.01$; $^{***}p < 0.001$

6. DISCUSSIONS

RQ1: Is the card game help students improve their understanding of the software development process?

Data obtained from the post-test showed that the experimental group performed better than the control group after engaging with the game, with mean scores were 90.72 (*sd* = 11.68) and 83.72 (*sd* = 15.40), and median scores were 94.5 and 89, respectively. Compared to the pre-test, the standard deviation of the experimental group decreased by 9.4 and the control group decreased by 3.1. The findings showed that, in comparison to the control group, the performance of students in the experimental group was relatively compact. In the questionnaire, students were asked whether they thought that adding card games had improved their understanding of software development. 72.2% of students strongly agreed (5) and 16.7% agreed (4). It demonstrates that the student's understanding of the software development process has improved as a result of the card game's inclusion, as confirmed by both test results and students' opinions.

RQ2: Does adding the card game to support a traditional course affect students' learning motivation in software engineering education?

The analysis results are in Section 5.5. has stated that hypotheses H1, H2, and H3 are supported by the results of the survey questionnaire. Based on the results, the inclusion that card games effectively affect performance expectations is held; (i) Card games improve understanding of software development knowledge, making it easier to remember software engineering development knowledge than traditional educational methods, making it easier to understand actual knowledge of software development; (ii) Hedonistic

motivation: playing with cards is enjoyable, card games are more motivating than traditional learning methods, card games are more competitive than traditional learning methods; (iii) Active learning: students can think about how to solve problems on their own and what game strategy should be used to get better competitive results. Generally, the students were able to better understand the educational objectives, felt more engaged in their learning, and were more motivated to learn, which had a positive impact on the learning content related to the software engineering practice course.

RQ3: What are the advantages and disadvantages of this game card for teaching software engineering?

We consolidated the students' responses to OEQ1, OEQ2, OEQ3, and consolidated similar feedback, and sorted out the benefits and drawbacks of the card game.

Advantages mentioned in OEQ1 and OEQ2 feedback:

a) Less boring and more motivated to learn (32%)
b) Increase interaction with group members through games (28%)
c) Easier to remember course content (25%)
d) Understand the process of software development (15%)

Students were motivated to learn the knowledge taught in the classroom, which can enhance the motivation to learn and increase the interaction with the group through the game. Some students even mentioned in their questions that they usually do not feel involved in the classroom, so they often feel bored and lack the motivation to learn.

Some students also reported that the card game helped them to remember the course content better. We believe that this is due to the physical nature of the cards, which makes the educational element a physical object that students can remember better. Through the feedback, we can tell that the students felt that the card game helped them to better understand the actual software development process.

Drawbacks mentioned in the OEQ3 feedback that could be improved:

a) The game mode is relatively single (15%)
b) Need to adjust the score allocation (30%)
c) Hope the game adds random elements (24%)
d) Hope to give more time to play (20%)

Although most of the responses were positive, some of the game's design still needs to be improved. For example, some students would like to see more variety in the game modes, while more students want to adjust the number of points earned in the game. It is also hoped that the game can be made more interesting by adding random factors such as chance or fate cards to the game chart. Some students also mentioned that they would like to have more time for card games in the course so that they can understand the course content better.

7. CONCLUSIONS AND FUTURE WORKS

Understanding the actual software engineering development process and its content

is important for students who will be working as software engineers in a near future. The complexity of the software engineering field has resulted in a lot of content in traditional courses that cannot be realized on paper. Some studies have integrated video games into their educational content, but fewer studies have used card games to integrate card games. Therefore, this study developed a serious game to educate the knowledge of the actual software development process with educational card games. A research experiment was also designed to evaluate the effectiveness of using educational card games in software engineering education. The research model was designed based on the UTUAT2 model, and the data from the experimental questionnaire was tested and analyzed through PLS-SEM.

Based on the experimental results, almost elements in the model had positive effects on behavioral intention. In addition, gamification has obvious positive effects on learning motivations and can indirectly influence students' behavioral intentions. Therefore, we believe the card game proposed in this study can have a positive impact on students' learning and software development practices.

In future work, we are going to take students' advice on the card game to change the content, add random elements, adjust the card score, and even add more educational objectives to enrich the game so that students can have more fun, learn more and become more competitive while playing. We also introduce more concepts related to the software development process and give students more time to play so that they can be more familiar with the knowledge and understanding.

REFERENCES

1. M. Shaw, "Software engineering education: A roadmap," in *Proceedings of Conference on Future of Software Engineering*, 2000, pp. 371-380.
2. S. Bylund, *The Road to the Unified Software Development Process*, Cambridge University Press, England, 2000.
3. D. Callahan and B. Pedigo, "Educating experienced IT professionals by addressing industry's needs," *IEEE Software*, Vol. 19, 2002, pp. 57-62.
4. C. C. Bonwell, and J. A. Eison, "Active learning: Creating excitement in the classroom," ASHE ERIC Higher, Education Report No. 1, School of Education and Human Development, George Washington University, Washington DC, 1991, pp. 1-6.
5. C. Caulfield, J. C. Xia, D. Veal, and S. P. Maj, "A systematic survey of games used for software engineering education," *Modern Applied Science*, Vol. 5, 2011, pp. 28-43.
6. C. G. von Wangenheim, R. Savi, and A. F. Borgatto, "SCRUMIA – An educational game for teaching SCRUM in computing courses," *Journal of Systems and Software*, Vol. 86, 2013, pp. 2675-2687.
7. J. Schell, *Tenth Anniversary: The Art of Game Design: A Book of Lenses*, CRC Press, NY, 2019.
8. V. Venkatesh, J. Y. L. Thong, and X. Xu, "Consumer acceptance and use of information technology: Extending the unified theory of acceptance and use of technology," *MIS Quarterly: Management Information Systems*, Vol. 36, 2012, pp. 157-178.
9. H. M. Lin, M. H. Lee, J. C. Liang, H. Y. Chang, P. Huang, and C. C. Tsai, "A review

of using partial least square structural equation modeling in e-learning research," *British Journal of Educational Technology*, Vol. 51, 2020, pp. 1354-1372.
10. M. R. de A. Souza, L. Veado, R. T. Moreira, E. Figueiredo, and H. Costa, "A systematic mapping study on game-related methods for software engineering education," *Information and Software Technology*, Vol. 95, 2018, pp. 201-218.
11. A. All, E. N. P. Castellar, and J. van Looy, "Digital game-based learning effectiveness assessment: Reflections on study design," *Computers & Education*, Vol. 167, 2021, p. 104160.
12. D. Lopez-Fernandez, A. Gordillo, P. P. Alarcon, and E. Tovar, "Comparing traditional teaching and game-based learning using teacher-authored games on computer science education," *IEEE Transactions on Education*, Vol. 64, 2021, pp. 367-373.
13. H. Y. Sung and G. J. Hwang, "A collaborative game-based learning approach to improving students' learning performance in science courses," *Computers & Education*, Vol. 63, 2013, pp. 43-51.
14. A. Baker, E. O. Navarro, and A. van der Hoek, "Problems and programmers: An educational software engineering card game," in *Proceedings of International Conference on Software Engineering*, 2003, pp. 614-619.
15. A. Soska, J. Mottok, and C. Wolff, "An experimental card game for software testing: Development, design and evaluation of a physical card game to deepen the knowledge of students in academic software testing education," *IEEE Global Engineering Education Conference*, Vol. 10, 2016, pp. 576-584.
16. G. G. Moreira and A. B. dos Santos Marques, "Evaluating the students' experience with the scrum card game: An experience report in a software engineering course," in *Proceedings of ACM International Conference Proceeding Series*, 2018, pp. 344-353.
17. G. V. Maturana-Gonzalez, C. M. Zapata-Jaramillo, J. D. Mosquera-Tobon, D. J. Medina-Gonzalez, and J. S. Arias-Velasquez, "Software project pool: A game for learning software project bidding," in *Proceedings of Annual ABSEL Conference on Developments in Business Simulation and Experiential Learning:* Vol. 46, 2019, pp. 304-310.
18. S. Santos, F. Carvalho, Y. Costa, D. Viana, and L. Rivero, "Risking: A game for teaching risk management in software projects," in *Proceedings of the XVIII Brazilian Symposium on Software Quality*, 2019.
19. V. Venkatesh, M. G. Morris, G. B. Davis, and F. D. Davis, "User acceptance of information technology: Toward a unified view," *MIS Quarterly: Management Information Systems*, Vol. 27, 2003, pp. 425-478.
20. B. A. Kumar and S. S. Chand, "Mobile learning adoption: A systematic review," *Education and Information Technologies*, Vol. 24, 2019, pp. 471-487.
21. J. A. Kumar and B. Bervell, "Google Classroom for mobile learning in higher education: Modelling the initial perceptions of students," *Education and Information Technologies*, Vol. 24, 2019, pp. 1793-1817.
22. G. Baptista and T. Oliveira, "Why so serious? Gamification impact in the acceptance of mobile banking services," *Internet Research*, Vol. 27, 2017, pp. 118-139.
23. P. Jakkaew and S. Hemrungrote, "The use of UTAUT2 model for understanding student perceptions using Google Classroom: A case study of Introduction to Information Technology course," in *Proceedings of the 2nd Joint International Conference on Digital Arts, Media and Technology: Digital Economy for Sustainable Growth*, 2017,

pp. 205-209.
24. J. Martí-Parreño, E. Méndez-Ibáñez, and A. Alonso-Arroyo, "The use of gamification in education: a bibliometric and text mining analysis," *Journal of Computer Assisted Learning*, Vol. 32, 2016, pp. 663-676.
25. M. Papastergiou, "Exploring the potential of computer and video games for health and physical education: A literature review," *Computers & Education*, Vol. 53, 2009, pp. 603-622.
26. R. T. Hays, "The effectiveness of instructional games: a literature review and discussion," *Research Gat*e, 2005, No. 235113027.
27. J. M. Randel, B. A. Morris, C. D. Wetzel, and B. V. Whitehill, "The effectiveness of games for educational purposes: A review of recent research," *Simulation & Gaming*, Vol. 23, 1992, pp. 261-276.
28. K. Wolfe, "Active learning," *Journal of Teaching in Travel & Tourism*, Vol. 6, 2008, pp. 77-82.
29. S. Hartikainen, H. Rintala, L. Pylväs, and P. Nokelainen, "The concept of active learning and the measurement of learning outcomes: A review of research in engineering higher education," *Education Sciences*, Vol. 9, 2019, p. 276.
30. M. Prince, "Does active learning work? A review of the research," *Journal of Engineering Education*, Vol. 93, 2004, pp. 223-231.
31. T. Lee, J. B. Lee, and H. P. In, "A study of different coding styles affecting code readability," *International Journal of Software Engineering and Its Applications*, Vol. 7, 2013, pp. 413-422.
32. E. Z. F. Liu, "Avoiding internet addiction when integrating digital games into teaching," *Social Behavior and Personality*, Vol. 39, 2011, pp. 1325-1336.
33. E. Zhi, F. Liu, and C. H. Lin, "Colloquium developing evaluative indicators for educational computer games," *British Journal of Educational Technology*, Vol. 40, 2009, pp. 174-178.
34. E. Z. F. Liu and P.-K. Chen, "The effect of game-based learning on students' learning performance in science learning – A case of 'Conveyance Go,'" *Procedia – Social and Behavioral Sciences*, Vol. 103, 2013, pp. 1044-1051.
35. J. M. Fernandes and S. M. Sousa, "Playscrum-a card game to learn the scrum agile method," in *Proceedings of the 2nd International Conference on Games and Virtual Worlds for Serious Applications*, 2010, pp. 52-59.
36. L. J. Williams, N. Hartman, and F. Cavazotte, "Method variance and marker variables: A review and comprehensive CFA marker technique," *Organizational Research Methods*, Vol. 13, 2010, pp. 477-514.
37. J. F. Hair, C. M. Ringle, and M. Sarstedt, "Partial least squares structural equation modeling: Rigorous applications, better results and higher acceptance," *Long Range Planning*, Vol. 46, 2013, pp. 1-12.
38. K. K.-K. Wong, "Partial least squares structural equation modeling (PLS-SEM) techniques using SmartPLS," *Marketing Bulletin*, Vol. 24, 2013, pp. 1-32.

Bao-An Nguyen received the BS in Hanoi University of Science and Technology, Vietnam, in 2005, and the MS and Ph.D. degrees in Information Engineering and Computer Science from Feng Chia University, Taiwan, in 2011 and 2021, respectively. He is currently a faculty of the Department of Information Technology, School of Engineering and Technology, Tra Vinh University, Vietnam. His research interests include data mining, software engineering, and education technology.

Hoang-Thanh Duong received a BS in Information and Technology from Tra Vinh University, Vietnam, in 2018, and an MS degree in Information Engineering and Computer Science from Feng Chia University, Taiwan, in 2021. He is currently a Ph.D. student in the same department at Feng Chia University. His research interests include deep learning, data mining, software engineering, and education technology.

Ling-Ling Tsao received the BS and MS degrees in Information Engineering and Computer Science from Feng Chia University, Taiwan, in 2021 and 2022, respectively. Her research interests include software engineering, computer vision, and object-oriented technologies.

Hsi-Min Chen received BS and Ph.D. degrees in Computer Science and Information Engineering from National Central University, Taiwan, in 2000 and 2010, respectively. He is currently a Professor with the Department of Information Engineering and Computer Science, Feng Chia University, Taiwan. His research interests include software engineering, programming education, object-oriented technology, service computing, and distributed computing.

Appendix 1. The Question cards for attacking (A) and the corresponding concept cards for defense (D).

Project Program Cards (A)	Development Concept Cards (D)
An upcoming product has a major issue. The product was coded with a logical error.	Engineers address product issues promptly or use Pair Programming to reduce program defects.
When writing the product's unit test, the coverage rate of the unit test cannot exceed a certain threshold. Discrepancies in discussions with product stakeholders regarding the required functionality for the project.	Teams will schedule project discussion meetings to enhance precisely unit test coverage or usage.
During product processing, different package versions frequently	Employ a unified development environment for each developer to avoid potential writing conflicts.
Due to the use of different environmental systems during project processing, some program data cannot be read or used.	Continuous delivery using Jenkins can unify environment variables and versioning.
Due to the different versions of the environment, there will be errors in the compilation of the program during the processing of the product.	Using GitLab for version control of a program can unify the package and product versions.
Test Cards (A)	**Test Concept Cards (D)**
The testing project discovered that the code patched one bug but introduced another. When the developer's code passes testing and is determined to be valid, the area is distinct, posing testing challenges.	Using Teams or Jira, communicate with developers regarding testing defects.
It is discovered during product testing that the provided product testing requirements are flawed. There are distinctions between testing product requirements and product development	Teams or Jira can be used to discuss product specifications with developers.
The test environment system is different from the developers, so some data cannot be read or utilized.	Please configure the environments of quality assurance personnel and developers in a
Using a different software version on the test causes errors during product launches due to inconsistencies.	Continuous delivery using Jenkins can unify environment variables and versioning.
Using a different environment version on test results in errors when launching production due to variations.	Using GitLab for version control of a program can unify the package and product versions.
Operation Cards (A)	**Operation Concept Cards (D)**
The server was compromised by outsiders, who stole crucial data. The SQL database in the background is corrupt, users cannot use SQL commands to retrieve data correctly, and some data are lost.	Emergency backup and setting the data as read-only to repair information security vulnerabilities or actively backing up the database during normal business hours; when the website is inaccessible, close the original data and enable users to access the backup data.

The server's system check is malfunctioning. There is no standardization in the online product support process. Errors require re-adjustment of environmental parameters.	Utilize Teams or Jira to communicate with developers and report issues.
The configuration version of the production environment differs from that of the developer, causing an error when executing the product	Please standardize the environments of developers and operations and maintenance personnel to prevent unidentified errors.
The production environment uses a different suite than the suite used by the developer.	Continuous delivery using Jenkins can unify environment variables and versioning.
Due to the difference in environment version, there will be errors in the program compilation during the processing of the product.	Using GitLab for version control of a program can unify the package and product versions.
There is a problem with the monitoring server's management, and the	The server is discovered to be disconnected from the monitor; therefore, it is necessary to determine whether the server is shut down or abnormal.

Appendix 2. The questionnaire and corresponding factor loading values.

Performance Expectancy（PE）		Factor Loading
PE-1	I think using card games improves my understanding of software development concepts	0.906
PE-2	I think using card games makes it easier to remember software engineering-related knowledge than traditional teaching methods	0.947
PE-3	I think using card games can help me understand the actual process of software development.	0.878
Hedonic Motivation (HM)		
HM-1	Learning with card games is enjoyable.	0.863
HM-2	Using card games gives me more motivation to learn than traditional teaching methods	0.900
HM-3	The competitiveness of using card games is enough to make me more active in my studies.	0.760
Active Learning (AL)		
AL-1	Through the card game, I think that I know how to solve the problem.	0.863
AL-2	Through the card game, I think about the meaning of the arrangement of the game flow.	0.858
AL-3	When I'm playing cards, I consider what strategies I can employ to improve my results.	0.953
Social Influence (SI) *		
SI-1*	The course's demands compelled me to participate in the card game.	0.841
SI-2*	My classmates actively participated in the card game, so I did as well.	0.884
Gamification (GM)		
GM-1	Gamification of educational objectives allows me to understand the educational content better than traditional text-based lessons	0.803

GM-2*	The gamification of the educational content made me feel more involved in the course learning	0.955
GM-3	The integration of the actual software development process through card games is helpful to me	0.960
GM-4	Physical card games are easier to remember than books or handouts	0.957
Behavioral Intention (BI)		
BI-1	I prefer to conduct the course with the aid of games rather than traditional teaching.	0.954
BI-2	If other courses are offered in the future (software quality testing), I would be more willing to participate if they are taught in a game format.	0.960
Open-Ended Question (OEQ)		
OEQ-1	How do you think the card game assistance will help in the actual course?	
OEQ-2	If you had to choose between a traditional class or a game as a course aid, which would you choose and why?	
OEQ-3	What improvements do you think the software engineering card game could make?	

* deleted items