Gamification in Software Engineering Education

G. Ivanova*, V. Kozov* and P. Zlatarov*
* University of Ruse/Department of Computing, Ruse, Bulgaria
givanova@ecs.uni-ruse.bg

***Abstract* – Gamification has proven to be an adequate approach in various educational environments, from kindergarten and elementary school to higher education classrooms. While it has been extremely effective with school students and standard subjects, such as STEM, when applied correctly, gamification and game-based learning can be applied with older students, even in the information technology and software engineering field. The paper outlines the results based on the experience of applying gamification of education to attract and stimulate student motivation and engagement in class. An approach for including games to teach software engineering methods and concepts is described. Different styles of software engineering games are examined and several of the most appropriate and relevant to the students' field of study are integrated in the Software engineering course. Descriptions of the games and the planned student progression through them are presented. A collaborative agile team-based approach is implemented and described. A suitable number of student groups have been selected, and the results from surveys held among those students are shown. Conclusions about the improvement of the process are discussed.**

Keywords - gamification, game based learning, education, software engineering, higher education

I. INTRODUCTION

Games are an innovative educational tool for actively engaging the learners and improving their motivation [1]. Games accelerate the acquisition of specialized knowledge of both elementary and intricate interconnections and actions in specialized fields. Educational games are used with most success in pre-school education, where most of the learning process is game-based. Games are also becoming a part of the curriculum of high school and university students in recent years [2, 3, 4]. In different studies, the game-based approach for adults is referred to as "gamification" [5] or "serious games" [6]. The term "serious games" is used to stress that these games are not meant to only entertain, but also achieve a certain educational result. The term "gamification" has entered the area of higher education as a description of the game-based approach used as a tool to increase student motivation and improve their engagement with the educational process [7].

As using games and simulations in the area of higher education is still in its early stages of development, experimental actions are necessary in order to achieve the desired improvement in all scientific fields. Gamification methods used in the course of Software engineering will be described in this report. Software engineering is a complex subject, the course is aiming to teach the students the different practices for creating software projects and solutions in both the practical and theoretical aspect. Software engineering is a wholesome systematic approach for analyzing, designing, creating, testing, documenting and supporting software projects by using different tools and methods for professional development and team management. The fast pace with which Software engineering has been evolving leads to the increasing necessity to teach students using the latest educational tools and keeping up with the newest trends in order to stay relevant to the business field. Fig. 1 presents a survey about the usability of the software project development practices in the industry [8]. The survey is conducted with developers in 2018. The results attest that the most widely used approaches are the agile methodologies: Agile, Scrum and Kanban. In the survey more than 58 981 participants have taken part.

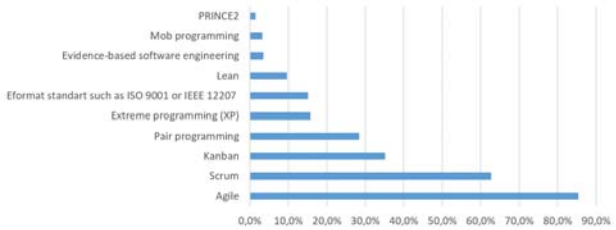

Figure 1. StackOverflow Developer Survey Results in 2018 about software methodologies

This paper will present a gamification method used in the learning process in the Software engineering course in the education of both Computer systems and technologies engineering students, and their peers from Telecommunications technologies bachelor degree courses in the University of Ruse. Some of newest and most relevant methodologies and technologies for designing different parts of a software project, as well as approaches for keeping track of and controlling the entire process - are being taught to the students. In the course workshops the learners go through the whole software process (Fig. 2) – from the problem to the end product, going through all the necessary stages - experiencing planning, how to deal with constraints on resources, teamwork communication and time management issues, all that has to be taken into account when designing and finishing a Software product.

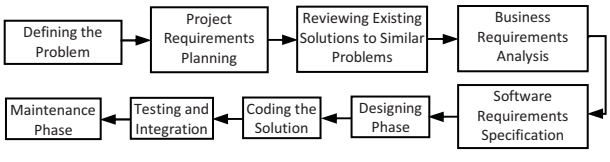

Figure 2. Software Engineering Process

The paper outlines the results based on the experience of applying gamification to the course. An approach for including games to teach software engineering methods and concepts is going to be described.

II. SOFTWARE ENGINEERING GAMES

A. *Collaborative Role Playing Game*

An experiment has been conducted with the participation of the students in the Software engineering course. It consists of using role playing as a game [9], where the students take on the roles of different positions taken by people in a normal software company. They have their own company and positions, each with its' different characteristics, which the students have to understand and apply.

Following the most modern trends in software engineering for using agile technologies [10], the students that are being taught in the course are given the task to create a software project. They are divided in groups of 5-6 people. Managing the project requires the creation of a special social construct – a project team, which has a temporary and strictly defined goal – to complete the project successfully. The teams have to create an imaginary company and define their budget for the project in the beginning of the semester, and the choices they have made are sealed with a virtual contract with a client.

Every member of the team has a main and secondary role. The roles in the software teams are as follows: Manager, Business Analyst, Programmer, Quality Assurance, Data Base Specialist, and Graphics Designer. The Manager is responsible for the overall control of the human resources and their tasks, with the goal being the successful delivery of the software product on time, based on the schedule, and its adherence to the clients' requirements. The Manager is necessary as all software projects need to comply with their budget and time constraints, which is an important aspect that has to be managed and understood.

The teams distribute their roles and responsibilities for the project by using a Gantt time chart. By using the Gantt chart the teams can visually present the time allocated for each task in the software project. The tasks are distributed horizontally – for each task there is one line, and each vertical line is a moment in time. Every task is a rectangle with a width proportional to the time it will take to complete it on the timetable. The plan details the tasks and activities that are needed for the completion of the work on the project; Mid-level expected results are defined; Control points are also defined – they are called milestones; A timetable with dates of beginning and completion for each task is designed; The time and resources necessary for the realization of the project are calculated; The resources – both human and material – are defined; The overall strategy for the quality management of the project is decided on; The personal responsibilities for each task is distributed to the members of the team, every person having tasks allocated on the chart. Fig. 3 presents a Gantt chart, created by a team of Erasmus exchange students, who were taught in the Software engineering course during their stay at the University of ruse in 2018.

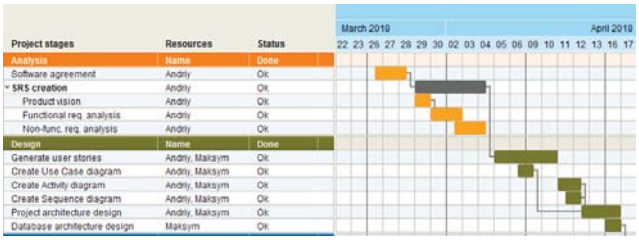

Figure 3. An example for Gantt Chart

The teams have to write all software requirements – including functional and non-functional ones – necessary for the successful project completion, with the guidance of Software Requirements Specification that they are given as an example. The Specification helps keep track of the main points that have to be completed for the project, introduces a shared vision of the completion of the project – for both the client and the company, as well as instructions on how to distribute each of the user roles as well as description of how each function of the software product should behave.

The teams are given the opportunity to play the role of Scrum or Kanban team. The agile methodologies are contemporary methods for creating, developing and supporting software products and services. Agile methodologies have the potential to meet the challenges in front of the modern software companies, functioning in an ever-changing and rapidly developing highly competitive scene [10]. In this business setting, the deciding factor between success and failure is managing time frames, quality assurance and pricing of the software products. These parts of the projects' lifetime are critical to the survival and evolution of a software company.

Scrum is an agile method for guiding the process of creating a software product [11], while giving the ability to react to different changes in the environment such as: product requirements, time for development, availability of resources and technologies; as all of these may be subject to change at the course of the project. Scrum allows higher levels of personal freedom to the teams and improves the flexibility of the whole process and its' ability to respond to change effectively. In scrum the teams have three main roles: product owner, team and scrum master. The team is self-governed and has a high level of independence and self-accountability. The Product owner is responsible for maximizing the profit gained for every investment. The owner achieves this by defining the product characteristics, thus creating a product catalogue, prioritizing each of the properties for the next sprint and constantly reevaluates and improves this catalogue. The Scrum master helps the product group with learning and applying the Scrum practices to the development process, and steers the team to success.

The teams which choose the Kanban agile methodology for process management [12], have the

opportunity to have a better view on the work process, as well as identification and vision on how tasks and process are handled on a Kanboard. Kanban's leading quality is transparency – the whole process is tracked, and each member of the team is given an equal amount of workload [13]. Kanban uses a visual control mechanism for tracking the work flow during the implementation of the different stages of realization. The Kanban methodology takes into account the number of started and unfinished tasks (WIP – work in progress tasks) and restricts starting more tasks until the previous ones have been completed, or until the next stage of development is nearing initiation. This approach allows the software developers to take tasks from a so-called "queue" with different sections - to do, backlog, planned, etc.; where all tasks are prioritized, categorized, described and the necessary restrictions have been placed.

Table I shows comparison between the Kanban and Scrum methodologies, which is given to the students to help them decide which methodology their teams would follow.

TABLE I. KANBAN VS SCRUM METHODOLOGIES

Scrum	Kanban
The time limited iterations are obligatory.	Time limitated iterations are optional.
The team promises to finish a work process in every iteration.	Work process finalization in every iteration is not obligatory.
Tempo is used as a metric for planning and improving the process.	Time for completion is used as a metric for planning and improving the process.
It is imperative that multifunctional teams are present.	Multifunctional teams are optional. Specialized teams are permitted.
Work needs to be separated in small parts in order to be finished within	There are no constraints on the size of tasks and their length.
It is necessary to use a chart for the remaining work (Burndown chart).	A work chart is not needed.
Indirectly limiting the incomplete tasks (for a sprint).	Directly limiting the incomplete tasks (during work flow).
Rating and prioritizing tasks is obligatory.	Rating and prioritizing tasks is not obligatory.
Three roles are mandatory - Product owner, Scrum master and team.	There are no mandatory roles.

After completing the specification, one of the first tasks for each team is to write user stories based on the specification. The list with user stories is a flexible method for managing the project requirements. User stories are short tasks, which define the specific goals for actors; which can be developed by the team in a short period of time - for example - in one sprint. For the creation and management of all user stories the students use the team management system – Jira.

Each team has access to Confluence and Jira (Fig. 4), deployed on a special cloud server for the university's needs. Confluence is used as a documentation version management tool, which helps keep track of the project's main priorities and requirements, while Jira is the team management tool for the work flow and tasks of the teams. Jira is used as a project management system; an especially useful component is the project board, which students use to augment learning Agile methodologies such as Scrum and Kanban, while also improving their task and time management skills. Both these applications are immensely useful in understanding the whole process of creating and managing a software project as a team and company.

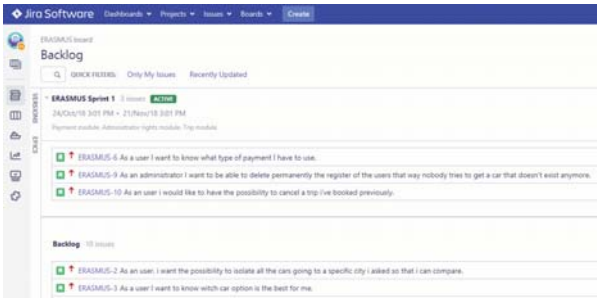

Figure 4. Jira Software used by students from the University of Ruse

B. Prioritization. Agile Estimating and Planning Poker Game

Software development teams can use different techniques for prioritizing the requirements of the project – organizing the tasks in level of importance is one of the most important responsibilities of the team. Even when projects start with stellar success, there comes a moment when a decision must be made on which requirements have to be completed, and which have to be postponed, in order for the product to be delivered according to the time schedule.

In the analysis of the requirements phase, every team decides the relevance and importance of each requirement by answering feasible questions. Time-space constraints have to be taken into account, resources, expenses and the possibilities for error must be well understood and calculated for. Another important major problem must always be planned for – clients frequently change their requirements even in an advanced stage of project development. Usually the clients have only an idea in the beginning, and the details are forged afterwards. This must be avoided, as it leads to incredible complications further down the development process. This is also known as the "metastasis of possibilities" – where the application features continuously grows to epic proportions.

One of the preferred methods which students use in the course is – MoSCoW [14]. This method is applicable to every software project, independent on the chosen control methodology. The "Moscow method" is a way to categorize requirements by importance, so as to decide which is actually important for the client, and which is not. The abbreviation MoSCoW means:

- Must have: requirements and functionalities, without which the main goal of the project would not be reached. They are the backbone of the system, the absolute minimum necessary for the project to have any meaning;

- Should have: features which are not critical to the products' core functionalities, but are incredibly important to the client;
- Could have: features which are useful to have in the application, but only if they do not require a large amount of work to be completed. They are the first ones that would be delayed when the project scope narrows due to incoming deadlines;
- Won't have: requirements which are absolutely out of the current scope of the project and do not enter the management plan. They are planned as future improvements or features to be added later in next versions of the product.

A mistake which students often make is to have all the project features and requirements marked as Must have, but this almost always results in a failed project.

Every task must be estimated before prioritization. The grading of the tasks is achieved by playing Poker Planning. Each of the teams starts the game and loads their user stories to be assessed. The grading is done through poker cards, where every card has a numeric value – weight. The team manager gives the team start game, and every member of the team has to personally estimate every user story, after which the game analyses all user grades and calculates the average result. Fig. 5 shows the online game (planitpoker.com), which is used by the students during class.

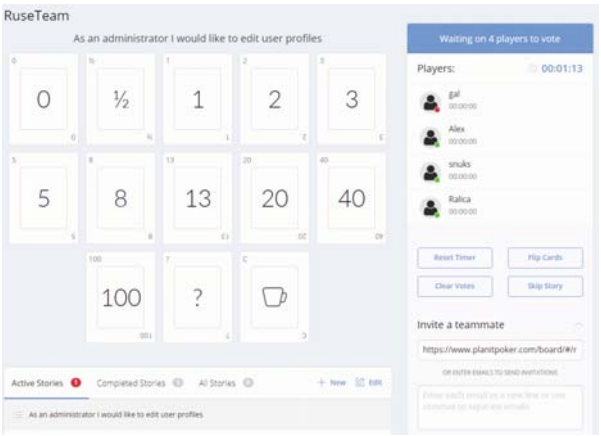

Figure 5. Poker Planning Game played by students during a lecture

Poker planning, also known as Scrum Poker [15], is a technique for rating efforts based on consensus. The members of the team judge user stories by playing with virtual numbered cards that other teammates cannot see. The results of the voting for each user story is revealed after everyone has finished choosing. Using such a game helps bypass the psychological effect anchoring bias, which occurs during voting by voice – as the risk of influencing each other's opinions beforehand is present. This effect can lead to incorrect grading and prioritizing tasks.

C. *Code Combat Game*

Various gamification techniques can also be applied during the code implementation phase. One example of a game geared towards teaching students to code is CodeCombat (Fig. 6). The system is, in its essence, a simple coding environment, disguised as a game. Its main objective is to teach students (mainly K-12) the basics of coding [16] in either one of two of the most popular programming languages – JavaScript and Python. Players get to control a variety of magical and/or hero characters through a multitude of levels, including mazes, obstacle avoidance and combat levels.

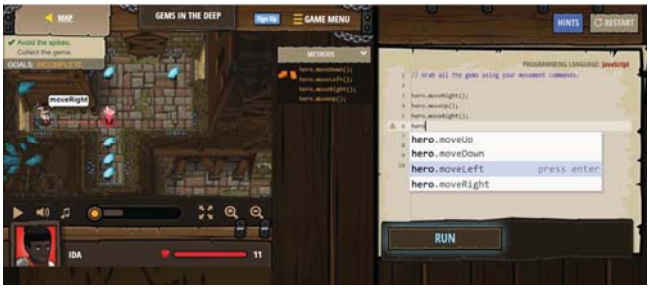

Figure 6. Code Combat Game played during a Software Engineering workshop

While this type of educational software might be intended for use with younger students, it can also be applied in higher education. It is especially interesting for computer engineering students who are less experienced with programming and coding.

The game has a dedicated classroom mode, which allows educators to track the whole class' overall progress, as well as generate reports and statistics on how the class or a particular student is doing (Fig. 7).

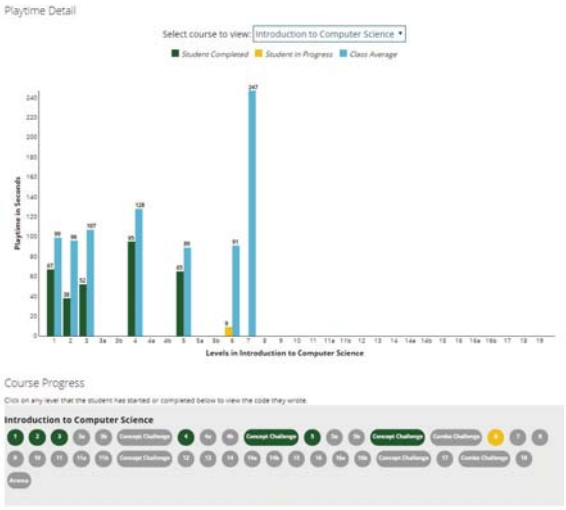

Figure 7. CodeCombat Playtime and Progress Stats

D. *Alphabet Brainstorming*

Alphabet Brainstorming [17] is a game which helps students generate connections between ideas and key words beginning with the letters from the alphabet. The game session was conducted with the students during the software engineering course, where they had to generate

as many words with key terms from the course lecture materials. Every student was given a sheet of paper with 30 letters from the Bulgarian alphabet. The goal was to determine at least one term with every letter of the alphabet. At the end of the game the generated terms for every letter were presented for discussion to all the students. Fig. 8 shows the overall student results from 2018. The results have been summarized and the most used key terms from the course have been picked out in the brainstorming game. An analysis on the student activity shows that 79% of the learners have succeeded in generating terms for more than 20 letters of the alphabet. The generated terms can be used as a reference to be included in the lecture material among more key words, which are usually put at the beginning of each lecture.

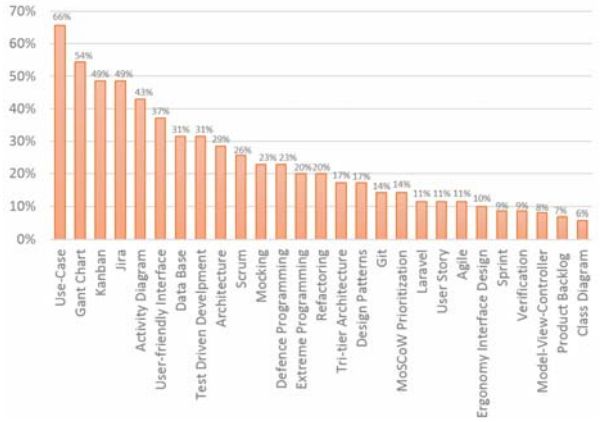

Figure 8. The terms used by students in the game Alphabet Brainstorming, from most to least

The educational goal achieved by using this type of game is improving knowledge acquisition skills in the learners, helping them synthesize connections between terminology and key words, and understanding them. The game is a great approach for reviewing the students' knowledge on the subject at the end of the semester.

E. *Kahoot Game*

The game Kahoot is an innovative method for actively engaging the students during class [18]. It can be used as a quiz game with a prize, for subject discussion or research on a topic from the lecture material. In the Software engineering course, Kahoot was used as a quiz game at the end of the semester during a lecture. The idea was to find out excellent students who had paid attention in class and quickly revise the course material. For the game there were 15 questions presented, which the students could view on their mobile devices using a generated code and logging in on the game's website. The game was received exceptionally well, as more than 90% of the attending students played the game and answered the questions in the allocated time frame. A competition formed about giving the most correct answer in the shortest time. After every question, the interactive board in the lecture room visualized the results in a player rank ladder with the students' points. At the end of the game there were three winners who received a golden, a silver and a bronze medal corresponding to their results.

On Fig. 9 the process for the creation of the quiz using kahoot.com website is shown.

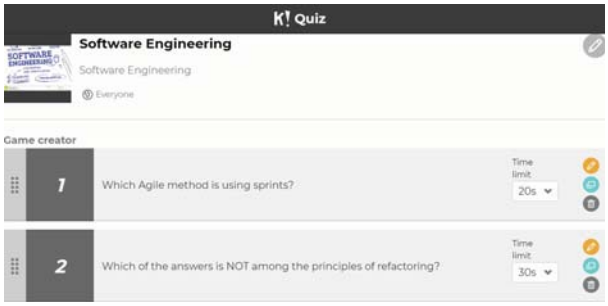

Figure 9. Kahoot Game Creator for Software Engineering

F. *The Millionaire Game*

The Millionaire game (Software Engineering Edition) was also used to review the course material at the end of the semester. Questions with different degree of difficulty were formed, and six difficulty levels were created. The students could also use jokers for help: having help from the audience (who were voting for the answer by lifting their hands); and call a friend. For the second joker, the students actually chose to call the assistant professor from the course, who was their workshop teacher. On Fig. 10 a question from the game is shown. It is about the Cumulative Flow Diagram, used in Kanban for tracking team progress and keeping information about tasks: Work in Progress (WIP), Cycle Time and Throughput.

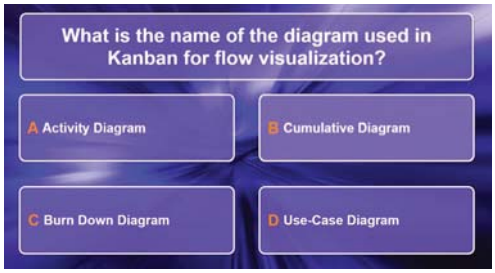

Figure 10. The Millionaire Game (Software Engineering Edition)

G. *Students Survey*

In 2018 a survey was held for full-time and part-time students' opinion on the software engineering course. The survey was done by 78 students who had finished the course. Fig.11 shows the results from the question where students had to evaluate the different games used in the course. The results on Fig. 11 show that the most interesting game for the students was the Brainstorming game, which had 65.5% of the participants voting for it. The role playing game was evaluated as interesting by 62.1% of the students, and in third place the survey ranks the Millionaire Game with 58,6%.

Results from the study show that all the students who have taken the survey approve the use of games in the learning process for the course. 79.3% of the respondents chose the highest option in the Likert Scale question "How would you rate the use of games in the Software engineering course?". The other 20.7% all selected the second best answer.

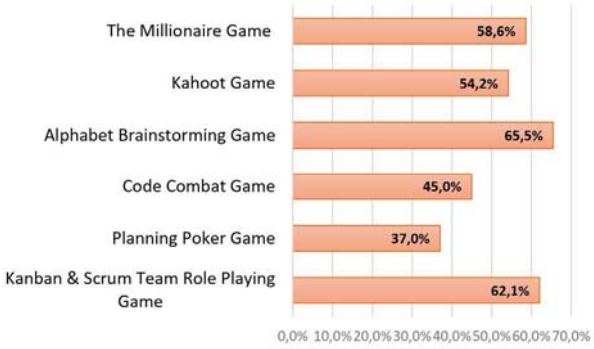

Figure 11. Survey Results - Software Engineering Games Evaluation

During the study, students have revealed that their interest and motivation to learn for the course has improved considerably after participating in the gaming activities. Notable are the answers to the question of whether the results achieved by the students themselves were above what they expected, and 44.8% of the learners attest that they have accomplished more in the course than they expected to achieve. The other 39.5% answered that they are satisfied (Fig. 12).

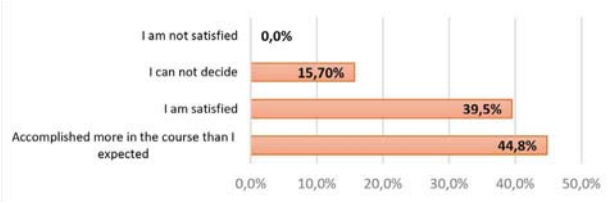

Figure 12. Survey Results

III. CONCLUSION

Software engineering is a mandatory subject included in the curriculum of computer engineers which aims to give them knowledge on complex matters which will be necessary for them for their future work in any software company. The course has only 60 hours of workload – 30 lecture hours and 30 workshop hours, which are not even close enough for the entire theoretical and practical material to be taught. The team that developed the study materials decided to use gamification as an innovative teaching approach for engaging student attention and heightening motivation for the learning process, with the goal of improving the students' ability to absorb and understand knowledge faster and more efficiently.

The educational games used in the course help demonstrate the use of software methodologies in a fun and efficient way. The results from the experiment show the positive approach students take to using gamification. The used gaming strategy in the learning process shows that this method is effective as it is being used for the third year. In the survey from 2018 a majority of the students showed that they approve of games as a viable strategy to improve learning, and prefer studying while playing.

ACKNOWLEDGMENT

The study was supported by contract of University of Ruse "Angel Kanchev", № BG05M2OP001-2.009-0011-C01, „Support for the development of human resources for research and innovation at the University of Ruse "Angel Kanchev". The project is funded with support from the Operational Program "Science and Education for Smart Growth 2014 – 2020" financed by the European Social Fund of the European Union.

REFERENCES

[1] S. Kim, K. Song, B. Lockee, J. Burton, "Gamification in Learning and Education," Advances in Game-Based Learning, ISBN 978-3-319-47282-9, Springer International Publishing, 2018.

[2] J. Doncheva, I. Ilieva "Practical and Applied Aspect of Motivation Role for Game and Physical Activity in Transition between Preschool and School Age," Activities in Physical Education & Sport 5.2, 2015.

[3] A. Juan, B. Loch, T. Daradoumis, S. Ventura, "Games and simulation in higher education," International Journal of Educational Technology in Higher Education, 2017

[4] D. Baeva, "The interactive multimedia as a language learning resource on the initial level of language acquisition," Paradigmata poznani, vol. 3, pp. 90-92, 2014.

[5] M. Ibáñez, A. Di-Serio, C. Delgado-Kloos, "Gamification for engaging computer science students in learning activities: A case study," IEEE Transactions on learning technologies 7.3, pp. 291-301, 2014.

[6] G. Baptista, G., T. Oliveira, T. "Gamification and serious games: A literature meta-analysis and integrative model." Computers in Human Behavior, vol. 92, March 2019, Pages 306-315, 2018.

[7] I. Caponetto, J. Earp, M. Ott. "Gamification and education: A literature review," European Conference on Games Based Learning, Academic Conferences International, vol. 1, 2014.

[8] StackOverflow Developer Survey Results 2018 - https://insights.stackoverflow.com/survey/2018/

[9] A. Amory, R. Seagram, "Educational game models: conceptualization and evaluation: the practice of higher education," South African Journal of Higher Education, 17.2, pp. 206-217, 2003.

[10] M. Fowler, J. Highsmith, "The agile manifesto," Software Development, 9(8), pp. 28-35, 2001.

[11] Rodriguez, Guillermo, Álvaro Soria, and Marcelo Campo. "Virtual Scrum: A teaching aid to introduce undergraduate software engineering students to scrum." Computer Applications in Engineering Education 23.1 (2015): 147-156.…

[12] Yacoub, Maha Khaled, Mostafa Abdel Athim Mostafa, and Ahmed Bahaa Farid. "A New Approach for Distributed Software Engineering Teams Based on Kanban Method for Reducing Dependency." JSW 11.12 (2016): 1231-1241.

[13] Anderson, David J., et al. "A comparative study of Scrum and Kanban approaches on a real case study using simulation." International Conference on Agile Software Development. Springer, Berlin, Heidelberg, 2012.

[14] J. Khan, I. Rehman, Y. Khan, I. Khan, S. Rashid, "Comparison of requirement prioritization techniques to find best prioritization technique," International Journal of Modern Education and Computer Science, 7(11), 53, 2015.

[15] M. Jaatun, I. Tondel, "Playing Protection Poker for Practical Software Security," International Conference on Product-Focused Software Process Improvement. Springer, Cham, 2016.

[16] D. Kumar, "Digital playgrounds for early computing education," ACM Inroads, 5.1, pp. 20-21, 2014.

[17] D. Buehl, "Classroom strategies for interactive learning,", Stenhouse Publishers, 2017.

[18] S. Licorish, et al. "Students' perception of Kahoot!'s influence on teaching and learning," Research and Practice in Technology Enhanced Learning, 13.1, 9, 2018.