Co-Created Design of a Serious Game Investigation into Developer-Centred Security

Manuel Maarek[1[0000-0001-6233-6341]], Sandy Louchart[2], Léon McGregor[1[0000-0002-4550-8292]] and Ross McMenemy[2]

[1] Heriot-Watt University, Edinburgh, UK
{M.Maarek,L.McGregor}@hw.ac.uk
[2] Glasgow School of Art, Glasgow, UK
S.Louchart@gsa.ac.uk,R.McMenemy1@student.gsa.ac.uk

Abstract. The cyber security context requires to better understand how developers write (in)secure code and to assist them in their software developments. We have developed a secure coding experiment and serious game intervention. In this paper, we report on the design of a serious game to investigate developer-centred security. We used a combination of approaches to shape discussions and support the serious game co-creation.

Keywords: Serious Game, Serious Game Design, Cyber Security, Software Security, Developer-Centred Security.

1 Introduction

Cyber security is a growing concern in a world of ever-increasing connectivity. Our activities and lives depend on software systems that are vulnerable as recent attacks have shown (e.g. impact on the UK health service of the WannaCry worldwide cyberattack). This results in a need to raise the awareness of software security issues and to train the developers of software systems, whether they are professionals or hobbyists.

As part of a project funded by the Research Institute in Science of Cyber Security (RISCS) in association with the National Cyber Security Centre (NCSC) and the EPSRC, we started to investigate how serious games could impact developer-centred security. In this paper, we report on the co-design of a serious game for code security. We used different serious game approaches to shape the discussions and the exploration for a serious game intervention. We focus on the steps we took to facilitate dialogue and the type of support we put in place in order to design a solution that fits the purpose of the experimentation. We present the process we implemented and provide a contribution to other serious game designers as to how a number of different methodologies can be used in order to provide an ad-hoc solution to a domain problem.

Plan. In Section 2, we give the background and motivation for the overall cyber security research project. Section 3 and Section 4 focus on the serious game design and implementation. In Section 5, we present the experimental setting surrounding the serious game. Finally, Section 6 concludes and draws future perspectives of this work.

2 Serious Game Investigation into Developer-Centred Security

While software systems become more ubiquitous and we are increasingly reliant on their use in our everyday lives, accessibility of mobile and cloud programming platforms makes software development and deployment more democratically available. This combination creates security challenges as software code could be deployed without having the security vetting or level of scrutiny that the end-users expect. The project in which this work takes place, proposes to investigate how serious game could play a role in evaluating and training the security skills of *the masses* of software developers. This project is part of the RISCS community for *developer-centred security* research. In our work, we focus particularly on serious games for secure software development and aim at identifying activities that could effectively persuade developers to improve their cyber security skills and increase the security of their code.

Background. A recent study has explored the software development security skills of GitHub users [1]. In this online experiment, the participants were invited to undertake three secure programming exercises in Python, their programming was then evaluated with regards to security properties. The experiment revealed that the self-reported security knowledge level or the professional or student status of the participants was not statistically related to the security grading of their programming solutions.

Serious Game Intervention Project. We chose to build our investigation as an extension to this base study, giving us some grounds to compare our results. We framed the experiment as an embedding of the programming exercises of the base study inside a game. The base study [1] evaluated the participants' ability to write secure code using three Python script tasks (URL Shortener, Credential Storage and String Encryption). A secure solution would for instance involve using a strong encryption algorithm or would prevent code injection. We complemented these exercises with three additions which have no obvious security focus (Image Analysis, Time Tool and Search & Replace), with the intention to compare the specific impact gamification has on security.

The base study [1] has found that developers often program insecure solutions regardless of their background. Developers will often have knowledge of security related concepts but fail to implement them or use an outdated or insecure standard. A game could be used to remind and raise awareness of security, while not necessarily instruct on the concepts or methods. As our primary purpose is to evaluate developers' security, the game intervention should replicate the base study with a control group. The game should motivate players to perform the programming tasks well. The key developer-centred security issues that the game could target are therefore: lack of awareness, outdated knowledge of standards, lack of experience, motivation or reasoning to implement proper security. To target these issues, the game could put the following potential processes in use: presenting an in-game context and motivation to build secure solutions, providing information on secure standards, challenging players with the effects of insecure solutions while maintaining neutrality to meet the evaluation aim.

Related Research. Recent surveys of game-based cyber security training [2, 3] show a growing interest in designing games for security. In [4], the authors show the benefits

of security exercises and competitions in cyber security training based on a survey of experiments. In [5], the authors used a game to study security decisions. Coding games such as *Code Hunt* are being adapted for secure coding [6], while secure coding competitions such as *Build It, Break It, Fix It* are organised as a *Catch The Flag* game [7]. Another coding game, *Code Defenders* [8], was primarily designed for crowed-sourcing purposes but also served in training [9]. In [10, 11], the authors advocate the use of dialectics and games for raising developers' security. Gamifications within software engineering has been studied and used [12], for example as a means to incite developers to remove compilers warning [13]. However our research aims at going beyond gamification, as other researchers also suggested in [14].

3 Design Methodology

To handle the complexity of cyber security and software programming, we chose to combine the LM-GM (Learning Mechanics – Game Mechanics) framework [15] with the Triadic Game design approach [16] to facilitate domain and serious game experts' discussions. The Triadic game design approach was initially used to explore the domain and range of potential intervention areas. The LM-GM framework was used to support the concrete definition of meaningful gameplay loops when determining the meaning of the intervention and the exact nature of the learning outcomes and gameplay experience. This section details how we used these approaches for the game co-creation.

3.1 Reflection on the Triadic Game Design Approach

The Triadic game design [16] is an approach to designing serious games through balancing three constituent parts. The worlds of; *Reality*, *Meaning* and *Play* are put forward as these parts and reflect balancing between serious aspects and gameplay. *Reality* describes the context behind the game and rely heavily on the domain expert to map out potential issues and problems within the target domain, often demanding some real-world features be reflected in the game. Secondly, *Meaning* is the phase of the design that focuses on exploring one or several intervention areas and lead to the identification of clearly intended outcomes for the player. intended outcomes of the game. We used the LM-GM framework in order to establish a clear link between learning and gameplay outcomes during this phase of the design. Finally, *Play* is a measure of interaction and fun within the game and represents the phase of the design when the game is actually designed according to pedagogies in line with the intervention's learning outcomes. Gameplay loops are created to match and map onto identified pedagogies and design the game aspects of the intervention. Balancing theses aspects can help a game to execute on its intended outcome, while being both believable and enjoyable to play. In our context, the *Reality* aspect of the game design framework had already been explored through the base study [1] and overseen by the cyber security co-creator of the game. As such, this phase of the design had already been completed and discussions focused on establishing a dialogue between serious games and cyber security researchers on the precise nature and meaning of the intervention.

3.2 Exploring Meaning Through the LM-GM Framework

LM-GM [15] is a game analysis and design model that allows for game mechanics to be studied and discussed in parallel with learning mechanics. We chose it for its simplicity and focus on semantics. The model helps to relate a set of standardised learning mechanics to another set of standard game mechanics. It allows for designers to investigate how the mechanics interact and to ensure that a game is grounded from a pedagogical and entertainment standpoint. Finally, it allows the definition of contextualised Serious Game Mechanics (SGM) that bridge the blending of leaning outcomes and gameplay elements. In our context, we used the framework to determine the nature of learning outcomes, feeding into the *Meaning* part of the Triadic game design approach.

Learning Mechanics. Game verbs [17] are a method of mapping learning and game mechanics by quantifying player actions. Table 1 gives the verbs used and their implementation. The aim of this exercise is to identify verbs relevant to the domain and tasks to frame a range of possible interventions based on pre-determined learning outcomes. The list of verbs with their meaning has been compiled from established learning theories (e.g. Piaget, Bloom's revised Taxonomy etc.) [17]. We decided to focus on using verbs that add potential for a motivational as well as learning impact.

Game Mechanics. The LM-GM framework proposes a set of standardised game mechanics for use with analytics, drawn from the SCVNGR set [18]. There are however many other useful game mechanic resources available and, from a purely design perspective, this exercise served as a framing device for design discussions and engaging stakeholders from disparate disciplines to discuss the overall game approach. As such, any game mechanics set can be used for this stage of the process. We later designed the game through the use of game bricks [19]. Table 2 gives the game mechanics we chose.

LM-GM Relationships. Fig. 1 displays the LM-GM diagram which describes how each set of mechanics relates to the game flow. It describes the specific game mechanics varying with game progression and how they relate to the LM and GM and the coverage of learning outcomes within the overall serious game approach. We see that both the macro and meta gameplay loops have roughly equal number of mechanics. Having sufficient mechanics relating to the tasks helps integrate task performance into the game.

Table 1. Learning verbs associated with the development of the serious game intervention

Verb	Gameplay Mechanic	Implementation	Task Usage/Motivation	
Instruct	Tutorial	Teach player the game	Context for tasks	A
Respond	Feedback	Provide feedback on a mistake	Give players contextual hints towards successful task completion	B
Act	Intervene	Allow players to react to situations where they lose control	Reflect methods of security prevention/response	C
Choose	Strategize	Task completion order enables gameplay strategy	Motivate players to complete task they perceive as impactful	D

Construct	Task Embodiment	Players will be able to interact with objects that are analogous to a task's process	Provide an in-game context and motivation for why the task should be performed	E
Present	Text	Tasks related information displayed within the game	Show hints towards a secure task	F
Situate	Provide Context	Have tasks completion affect the players abilities in game	Show how good and bad implementations of the task affect the game	G
Reward	Rewards	Give in-game rewards for good play and task performance	Motivate through extrinsic rewards	H

Table 2. Game mechanics of the game

Game mechanic	Description	
Selecting Collecting	The player scores and collects points based on their performance	1
Tokens	Players earn tokens to track their task progression and show the task-to-game mechanical interaction	2
Infinite Gameplay	The game is designed to infinitely loop replaying a level, in order to facilitate hi-score chasing and illustrate player progression	3
Strategy	Players can pick and choose how they want to play the game by utilizing and upgrading certain mechanics over others	4
Resource Management	Player actions have associated resources that must be balanced to make effective use of all mechanics	5
Eliminate	The game presents security threats that must be eliminated to perform	6
Time Pressure	The main level has a set number of waves, which appear at certain time intervals	7
Meta-Game	The tasks are integrated within the game through a meta element that links an out of game mechanism to the game	8
Tutorials	The gameplay is taught through an instructional tutorial	9
Competitions	A fake leader board and core systems encourage hi-score chasing and game replaying	10
Rewards Penalty	Players are rewarded based on their in-game and task performances	11

3.3 Conceptualising Play through Gameplay Loops

We can restructure the gameplay loops [20] to also take LM-GM into account, see Fig. 2. This should help illustrate what mechanics are most important to the serious game intervention and how game mechanics are linked to the learning mechanics. Here we can see that strategizing is an important aspect for the macro gameplay loop (left diagram of Fig. 2), which is completely overlooked in the game map (Fig. 1).

If a new threat was to be added, it must be highly relevant to the security concerns, such as to target Act (C) and Situate (H). In addition, it should have certain strategies and resources needed to eliminate it. Security relevance is important to the loop, mainly

through Construct (E) which presents text information. However, this game is intentionally limited to allow for a control group. If the game was to be a learning tool, this element could be strengthened through increased presence and specificity.

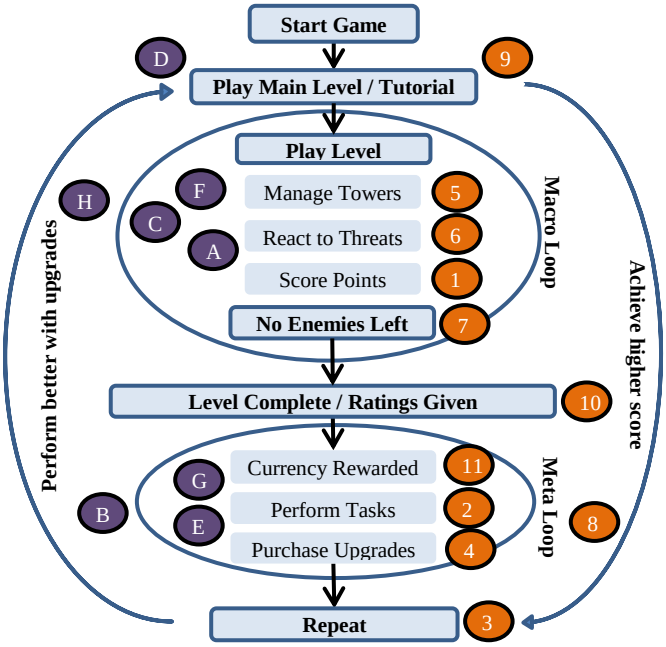

Fig. 1. Game mechanics map

The goal of having players complete programming tasks relies on the idea of a meta-game. In this context, the meta-game takes on two of its common meanings: having external elements (the tasks) affect the game state and enabling strategic trends to develop by allowing player to choose what task to perform. To strengthen the tasks, the Construct (E) learning mechanics could increase the immersive and motivational factors to complete tasks. This would be by having highly relevant and contextual upgrades presented. Another method would be to increase the play mechanics of the meta-game by adding game elements to the tasks or more strategic depth to the upgrades.

The map and loops show that an upgrade could be used to strengthen motivation through a meta loop element – as all in-game mechanics strengthen only macro or micro abilities. An upgrade that targets either the currency or task completion tokens could support the motivational factors provided by Reward (G) in a virtuous cycle, by motivating players to increase their own motivational incentives.

Finally, adding threats can be expanded to consider how its relevant upgrades are developed. In this case, mechanics can be either security or non-security relevant, provided that its integration into the game share strong relations with the corresponding tasks. The upgrade must enable or strengthen a mechanic, but also have a clear comparative strength to other upgrades. These enable the strategic game mechanic, but also the meta-game strategy of choosing the strongest upgrades to purchase. Weak upgrades need to exist to enable the meta-game and players discovering dominant strategies.

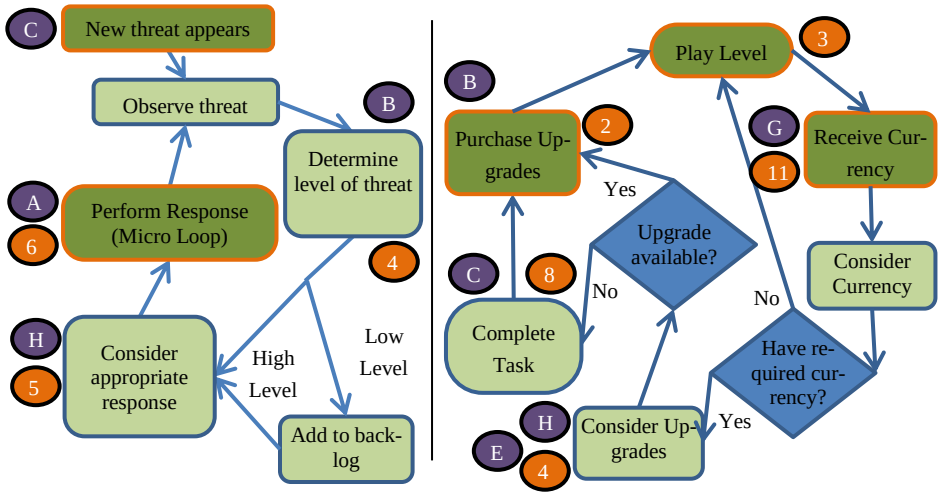

Fig. 2. Macro loop (left) and Micro loop (right)

Dark green represents in-game mechanics while light green represents an out-game cognitive response.

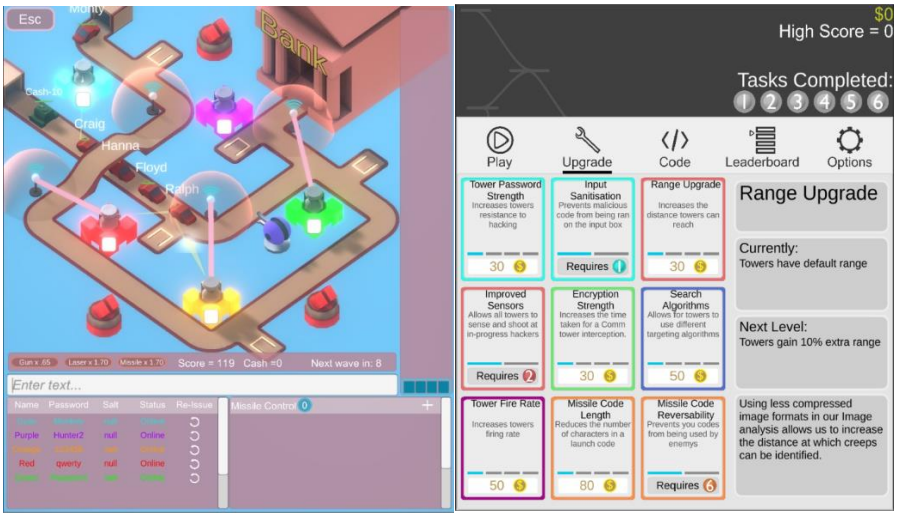

Fig. 3. The game showing standard gameplay (left) and the upgrade system for the game (right)

4 Game Implementation

The games basic genre is that of a *Tower Defence*, sub-genre of strategy games and well known for being approachable and having players improve their abilities over time. The player manages static defences and traps to defend from an invading enemy. The game takes place in a situation where the player is defending a bank from cars and trucks that are attempting to rob it. Before playing the next level, the player spends bank currencies to improve the security systems.

Context and Integration. In order to situate a player for tasks performance, core game mechanics and tasks share some context and interaction methods. Game text is written primarily to strengthen these shared contexts than present a narrative. Typing is used as the main game interaction method to draw parallels with programming. All tasks have in game counterparts, presenting reasons for development and example motivations. Similarly, several common security concerns are represented through the effects they have on the game mechanics – giving players a contextual reasoning for developing tasks with a security mind-set. The game features an upgrade system, in which several gameplay elements can be upgraded – improving and expanding on a player's ability to play the game. However, each of these elements is locked behind the completion of a corresponding programming task. Upon upgrading, the player's in-game abilities are noticeably improved – allowing them to score higher and play better. The motivational drive to continue to achieve higher scores and explore new gameplay options is directed into the completion of tasks.

Gameplay Mechanics. Mechanics follow similar structure, interaction methods and appearance, such that a player can quickly learn and understand a new mechanics after being introduced to a related one. The player is working to defend the bank from *Creeps* of several kind that are trying to reach it. The creeps have to travel along a winding road to reach the bank. Under the players control are serval *Towers* which will damage and destroy the creeps by shooting them. Between levels the player is able to improve their in-game abilities through an upgrade system, which increases the effectiveness of the towers or increases defences to a threat. Table 3 gives the four kinds of towers of the game and their relations with the programming tasks and motivations.

Table 3. Table relating the towers to tasks and task motivations.

Tower	Player Interaction Blocks	Task Integration	Integration Mechanics	Task Completion Motivation
Standard	Write, Match, Select, Manage	Credential Shortener, Time Tool, Search & Replace, Image Analysis	Creep hacking and password resetting	To decrease how often the player has to manage the tower
Communication	Select, Create, Manage	String Encryption	Encryption laser communication	To decrease how often the player has to manage the tower
Laser	Write, Match, Select	Credential Storage	SQL codes having adverse effects	To allow player to select tanks without having to weigh the pros and cons of entering SQL
Missile	Write, Match, Select, Shoot	URL Shortener	Generating codes and shorten them	To shorten the time taken to write codes To prevent the code entering time from being wasted

Game Heuristics. While a tutorial is in place, the layouts of the game elements instruct players how to play [21]. Directional heuristics help direct a player to specific elements: the player needs to read potentially moving words and write them into a text box. Fig.

3 shows the game being played, and the upgrade system. Positional heuristics give the player feedback: for example, a bar shows progress when a tower is being hacked.

5 Design of the Experiment

Our choice has been for an online experiment. The experimental platform consists of a Web-browser extension, the game, a Web-API server and an online questionnaire. Participants are invited to log into the GitHub website and install the extension which guides them through the experiment and provides a programming environment for the tasks. We chose to base the interface on the GitHub website as it is a natural developers' codebase. Participants can follow the tutorial to familiarise themselves with the game. Within the game, participants would choose to take on the programming tasks. Completing an individual task activates the corresponding upgrade. After playing the game the participants are invited to fill out an online questionnaire that combines the questionnaire used by the base study [1] with game-related questions inspired by [22]. A Web-API server performs software tests on the participant solutions to the programming tasks and collects gameplay data for analysis. The platform includes a no-game mode to be able to replicate the base study with a control group. After an early pilot of the experiment we improved and explained better to the participants the flow between the game and the programming environment. We are in the process of inviting participants for the experiment, so we cannot report results at this stage.

6 Conclusion

We have presented the combination of serious game design approaches we used in the co-creation of a serious game to investigate developer-centred security. We have also described the mechanics and gameplay loops of the serious game we have developed and how we integrated the programming tasks within the game. We have given an overview of the experimental platform which we have built for this ongoing experiment. This work is an initial step which shows the potential that serious game for software engineering and secure coding could offer to assist and engage developers with a specific concern such as cyber security.

References

1. Acar, Y., Stransky, C., Wermke, D., Mazurek, M.L., Fahl, S.: Security Developer Studies with GitHub Users: Exploring a Convenience Sample. In: Symposium on Usable Privacy and Security (SOUPS) (2017).
2. Tioh, J.N., Mina, M., Jacobson, D.W.: Cyber security training a survey of serious games in cyber security. In: IEEE Frontiers in Education Conference (FIE). pp. 1–5 (2017).
3. Hendrix, M., Al-Sherbaz, A., Bloom, V.: Game Based Cyber Security Training: are Serious Games suitable for cyber security training? International Journal of Serious Games. 3, (2016).
4. Sommestad, T., Hallberg, J.: Cyber Security Exercises and Competitions as a Platform for Cyber Security Experiments. In: Secure IT Systems. pp. 47–60. (2012).

5. Frey, S., Rashid, A., Anthonysamy, P., Pinto-Albuquerque, M., Naqvi, S.A.: The Good, the Bad and the Ugly: A Study of Security Decisions in a Cyber-Physical Systems Game. IEEE Transactions on Software Engineering. (2018).
6. Xie, T., Bishop, J., Tillmann, N., de Halleux, J.: Gamifying Software Security Education and Training via Secure Coding Duels in Code Hunt. In: Symposium and Bootcamp on the Science of Security. pp. 26:1–26:2. ACM (2015).
7. Ruef, A., Hicks, M., Parker, J., Levin, D., Mazurek, M.L., Mardziel, P.: Build It, Break It, Fix It: Contesting Secure Development. In: ACM SIGSAC Conference on Computer and Communications Security. pp. 690–703. (2016).
8. Rojas, J.M., White, T.D., Clegg, B.S., Fraser, G.: Code Defenders: Crowdsourcing Effective Tests and Subtle Mutants with a Mutation Testing Game. In: International Conference on Software Engineering. pp. 677–688. IEEE (2017).
9. Rojas, J.M., Fraser, G.: Code Defenders: A Mutation Testing Game. In: Intl. Conference on Software Testing, Verification and Validation Workshops (ICSTW). pp. 162–167 (2016).
10. Weir, C., Rashid, A., Noble, J.: Reaching the Masses: A New Subdiscipline of App Programmer Education. In: ACM SIGSOFT International Symposium on Foundations of Software Engineering. pp. 936–939. (2016).
11. Weir, C., Rashid, A., Noble, J.: I'd Like to Have an Argument, Please : Using Dialectic for Effective App Security. In: European Workshop on Usable Security (EuroUSEC). (2017).
12. Pedreira, O., García, F., Brisaboa, N., Piattini, M.: Gamification in software engineering – A systematic mapping. Information and Software Technology. 57, 157–168 (2015).
13. Arai, S., Sakamoto, K., Washizaki, H., Fukazawa, Y.: A Gamified Tool for Motivating Developers to Remove Warnings of Bug Pattern Tools. In: International Workshop on Empirical Software Engineering in Practice. pp. 37–42. IEEE (2014).
14. Barik, T., Murphy-Hill, E., Zimmermann, T.: A perspective on blending programming environments and games: Beyond points, badges, and leaderboards. In: IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC). pp. 134–142 (2016).
15. Lim, T., Louchart, S., Suttie, N., Hauge, J.B., Earp, J., Ott, M., Arnab, S., Brown, D., Stanescu, I.A., Bellotti, F., Carvalho, M.: Serious Game Mechanics, Workshop on the Ludo-Pedagogical Mechanism. In: Games and Learning Alliance. pp. 174–183. (2014).
16. Harteveld, C.: Triadic Game Design: Balancing Reality, Meaning and Play. Springer (2011).
17. Arnab, S., Lim, T., Carvalho, M.B., Bellotti, F., Freitas, S., Louchart, S., Neil, S., Berta, R., De Gloria Alessandro: Mapping learning and game mechanics for serious games analysis. British Journal of Educational Technology. 46, 391–411 (2014).
18. Schonfeld, E.: SCVNGR's Secret Game Mechanics Playdeck, http://social.techcrunch.com/2010/08/25/scvngr-game-mechanics/, (2010).
19. Djaouti, D., Alvarez, J., Jessel, J.-P., Methel, G., Molinier, P.: A Gameplay Definition Through Videogame Classification. Int. J. Comput. Games Technol. 2008, 4:1–4:7 (2008).
20. Guardiola, E.: The Gameplay Loop: A Player Activity Model for Game Design and Analysis. In: ACM Intl. Conference on Advances in Computer Entertainment Technology. (2016).
21. Fullerton, T.: Game Design Workshop: A Playcentric Approach to Creating Innovative Games, Third Edition. A K Peters/CRC Press (2014).
22. IJsselsteijn, W.A., Kort, Y.A.W. de, Poels, K.: The Game Experience Questionnaire. Technische Universiteit Eindhoven, European Community - New and Emerging Science and Technology (NEST) programme (2013).