Towards a Serious Game to Teach ISO/IEC 12207 Software Lifecycle Process: An interactive learning approach

Conference Paper *in* Communications in Computer and Information Science · June 2015

DOI: 10.1007/978-3-319-19860-6_17

Murat Yilmaz
Gazi University

Rory V. O'Connor
Dublin City University

Towards a Serious Game to Teach ISO/IEC 12207 Software Lifecycle Process: An interactive learning approach

Ufuk Aydan[1,2] and Murat Yilmaz[1,2] and Rory V. O'Connor[3]

[1] Çankaya University, Game Research & Development Laboratory, Turkey
aydanufuk@gmail.com
[2] Çankaya University, Department of Computer Engineering, Turkey
myilmaz@cankaya.edu.tr
[3] School of Computing, Dublin City University, Ireland
roconnor@computing.dcu.ie

Abstract. ISO/IEC 12207 training is a key element to provide an ability to software development organizations for selecting a set of required processes, measuring the performance of these processes, and continuously improving them. Traditionally, such training is either performed by an expert individual to the software quality management personnel most likely in form of a seminar in a classroom environment. This may also be given by a qualified professional, such as a registered auditor. However, software requirements are usually subject to change, and therefore such training is not enough to teach the substantial details of the entire standard. This has led to increased reports of complications, which demotivates organization to use this standard. To improve the quality of traditional training, a 3D serious game was proposed. The preliminary idea here is that the training is utilized as a game that employs 3D office landscape to provide a realistic virtual environment for ensuring that the training will be based in a real-world-like environment. Before building a prototype for our serious game, we consulted five industrial experts whose works are related with ISO standards. To give these practitioners an opportunity to explore the conceptual design and raise some potential problems, the semi-structured interview method was used. Based on the suggestions of experts, proposed model of the serious game were revised. Taken together, initial results suggest that a serious game for teaching ISO/IEC 12207 should be useful for individuals who are interested in learning more about the standard.

1 Introduction

ISO/IEC 12207 is an international software engineering standard that defines the software engineering processes and activities, which are associated with software life cycle process from conception to the end of product [1]. This standard defines a set of suitable roles for software practitioners and follows the plan-do-check-act cycle for improving the quality of the product [2].

ISO/IEC 12207 is based on *the qualitative definitions of the processes*, and therefore there are no implementation details of defined the tasks and activities [3]. Moreover, it does not measure the quality, it does not define specifically how to do activities and tasks, and it does not prescribe to specific methods, practices or tools. Its modular structure is suitable for tailoring. Therefore, an organization can customize the necessary parts of the standard that are planned to be used based on the requirements of a software project [4]. Because of the high modularity of the standard, it is more easily to deal with factors that are affecting the software development such as complexity, schedule, cost, etc. In addition to that ISO/IEC 12207 can act as an *inventory* of processes, which give different perspectives to specific parts of the software life cycle process. These processes are categorized as organizational processes (i.e. management, infrastructure, improvement, training), supporting processes (i.e. documentation, configuration management, quality assurance, verification, validation, joint review, audit, problem resolution), and primary processes (i.e. acquisition, supply, development, operation, maintenance) [5].

ISO/IEC 12207 is a guideline based on a set of process descriptions for providing a base for adopting a role, which defines a set of constraints for selecting process, activities or tasks that are required for software development. The standard proposes a set of views that can be used to label the processes connected to a role. To this end, it offers five different viewpoints as follows: (i) contract, (ii) engineering, (iii) operating, (iv) quality management, and (v) management views [4]. Firstly, there is a contract view that includes an acquisition process (i.e. for the acquirer) and supply process (i.e. for the supplier). Secondly, there is an engineering view which has a development process for product development and a maintenance process for upkeeping the software. Thirdly, the operating view with the operation process that provides a guideline for operating the software. Fourthly, a quality management view that has six processes; (i) joint review, audit, verification, validation, quality, and problem resolution processes [4]. ISO/IEC 12207 Software Lifecycle processes can be maintained by 7 main phases by any organization which have capability to support the standard's views and ability to handle software engineering requirements. These main phases are; (i) requirements analysis, (ii) specification, (iii) design, (iv) coding, (v) verification & validation, (vi) installation, (vii) maintenance & support [6].

Despite the fact that ISO/IEC 12207 is a well-structured and detailed technical text on a complex subject, many professionals find it difficult to gain substantial information regarding to the standard. Games are found to be effective learning tools especially for teaching complex subjects. In particular, serious games are a kind of interactive computer application (i.e. computer simulations of real-life situations or processes) designed for educational purposes. As a serious game is designed to include an educational aspect [7], the learners can be challenged with possible scenarios that may found similar to a real-world problem. However, a well-designed serious game should include game playing and a set of serious aspects (e.g. teaching, learning, communication, information, etc.) where such combination should be based on an utilitarian goals [8]. In fact, this scenario should be aligned with gaming objectives that implements the dramatic elements of a game such as story, sound, rules, graphics, etc.

In light of this remarks, the goal of the study is to investigate the possibilities of a game that is designed for teaching the primary concepts of ISO/IEC 12207. The rest of the paper is organized as follows. Section 2 presents the background of the study. It details of the ISO/IEC 12207 standard and the notion of serious games. Section 3 includes a discussion about the customization of the standard. Further, it details the applications of serious games in software engineering. Next, we discuss the tentative plans for an ideal game. Lastly, paper concludes with conclusion and future work.

2 Background

2.1 ISO/IEC 12207

Similar to the definition of ISO/IEC 12207, definition a software life cycle starts with a requirement analysis based on a need and eventually the life cycle ends with the retirement of a product [9]. The standard has an architecture, which is built by set of interrelated processes, which are consequent to modularity and responsibility. While defining modularity under the conceptualization of the standard is about being unique with every processes and availability of being capable enough to handle all types of projects. The processes that are designed for the standard have a modular structure. From the practical point of view, the modules have the maximum cohesion and minimum coupling where each process supports unique functionalities as possible [5]. To clarify every part which is associated with the life cycle has specific and well defined responsibility to take care. However, from a traditional point of view, the modules of a life cycle should be studied distinctively.

To understand the basics of the standard, the definition of organization and party should also be elaborated. The terms organization and part are required to highlight different viewpoints that can be acquired using the standard. The term organization defines a group of persons (or authorities) with a set of responsibilities who are organized for a particular objective. However, the party defines an organization that enters into a contract, which can be either from an organization or more. The name that is given to a part is usually correlates with the name of the process it performs (e.g. an acquirer is involved with the acquisition process). There are several roles, which can be directly related with the process names from the standard such as acquirer, supplier, implementer, maintainer, and operator, etc [6].

The ISO/IEC 12207 processes can be organized into three main categories: primary processes; supporting processes; and organizational processes (see Figure 1).

Primary processes described by the standard are (i) acquisition, (ii) supply processes, (iii) development processes, (iv) operation processes, and (v) maintenance processes. The goal of a supporting process is to support other processes while fulfilling a function. The supporting processes are identified as (i) documentation processes, (ii) configuration management processes, (iii) quality assurance processes, (iv) joint review processes, (v) audit processes, (vi) verification processes, (vii) validation processes, and (viii) problem resolution processes. The organization processes are employed by an organization to manage, control and improve the life cycle processes.

There are seven process groups that are defined by ISO/IEC 12207 which can be accomplished during the life cycle of a software system. Each of the life cycle processes within those groups can be defined with respect to its goals and expected outcomes. Figure 2 shows the activities and tasks that should be carry out to accomplish these outcomes [6].

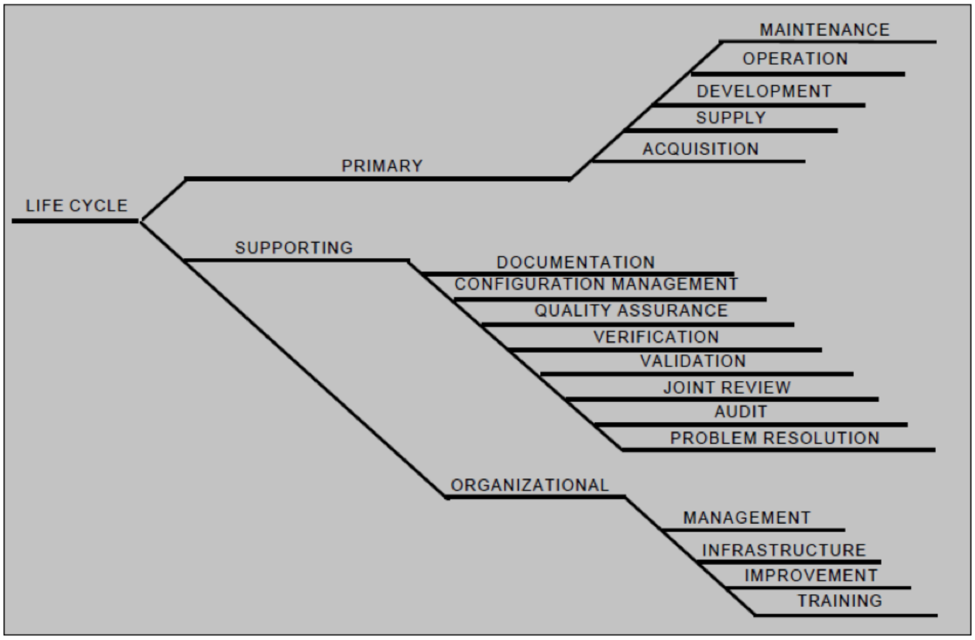

Fig. 1: *The Life Cycle Processes of ISO/IEC 12207 [5]*

(a) Agreement Processes (two processes) (subclauses 5.2.2.1.1 and 6.1)
(b) Organizational Project-Enabling Processes (five processes) (subclauses 5.2.2.1.2 and 6.2)
(c) Project Processes (seven processes) (subclauses 5.2.2.1.3 and 6.3)
(d) Technical Processes (eleven processes) (subclauses 5.2.2.1.4 and 6.4)
(e) Software Implementation Processes (seven processes) (subclauses 5.2.2.2.1and 7.1)
(f) Software Support Processes (eight processes) (subclauses 5.2.2.2.2 and 7.2)
(g) Software Reuse Processes (three processes) (subclauses 5.2.2.2.3 and 7.3).

Basically, the design of ISO/IEC 12207 software life cycle process was constituted with a set of complementary components [6]. For instance, each process has its own activities that cover cohesive tasks where tasks have necessary actions [5]. A task takes several type of inputs (e.g. data, information) and generate outputs (e.g. data, information). A set of verbs such as *will* (for declaration of purpose), *shall* (for binding provision), *should* (for recommendation) is used to express requirements, recommendations or acceptable actions.

The standard utilizes the fundamentals of quality management techniques which are the integral and indispensable parts of the total life cycle. Therefore, each process is a basic implementation of plan-do-check-act (PDCA) cycle. To implement ISO/IEC 12207 properly, it is important to know that each process and individuals who are in charge of related processes must be aware of their particular roles and responsibilities in all processes. Based on the assigned roles, evaluations of particular tasks have to be carried out properly within the software development organization [10].

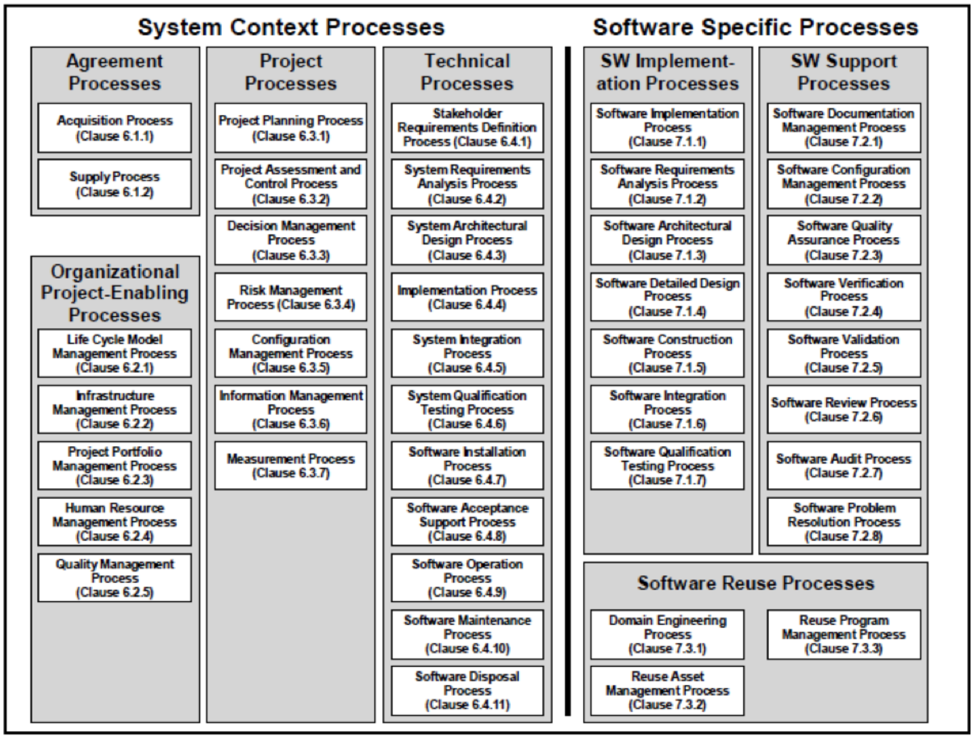
System Context Processes
Software Specific Processes
Agreement Processes
Acquisition Process (Clause 6.1.1)
Supply Process (Clause 6.1.2)
Organizational Project-Enabling Processes
Life Cycle Model Management Process (Clause 6.2.1)
Infrastructure Management Process (Clause 6.2.2)
Project Portfolio Management Process (Clause 6.2.3)
Human Resource Management Process (Clause 6.2.4)
Quality Management Process (Clause 6.2.5)
Project Processes
Project Planning Process (Clause 6.3.1)
Project Assessment and Control Process (Clause 6.3.2)
Decision Management Process (Clause 6.3.3)
Risk Management Process (Clause 6.3.4)
Configuration Management Process (Clause 6.3.5)
Information Management Process (Clause 6.3.6)
Measurement Process (Clause 6.3.7)
Technical Processes
Stakeholder Requirements Definition Process (Clause 6.4.1)
System Requirements Analysis Process (Clause 6.4.2)
System Architectural Design Process (Clause 6.4.3)
Implementation Process (Clause 6.4.4)
System Integration Process (Clause 6.4.5)
System Qualification Testing Process (Clause 6.4.6)
Software Installation Process (Clause 6.4.7)
Software Acceptance Support Process (Clause 6.4.8)
Software Operation Process (Clause 6.4.9)
Software Maintenance Process (Clause 6.4.10)
Software Disposal Process (Clause 6.4.11)
SW Implementation Processes
Software Implementation Process (Clause 7.1.1)
Software Requirements Analysis Process (Clause 7.1.2)
Software Architectural Design Process (Clause 7.1.3)
Software Detailed Design Process (Clause 7.1.4)
Software Construction Process (Clause 7.1.5)
Software Integration Process (Clause 7.1.6)
Software Qualification Testing Process (Clause 7.1.7)
SW Support Processes
Software Documentation Management Process (Clause 7.2.1)
Software Configuration Management Process (Clause 7.2.2)
Software Quality Assurance Process (Clause 7.2.3)
Software Verification Process (Clause 7.2.4)
Software Validation Process (Clause 7.2.5)
Software Review Process (Clause 7.2.6)
Software Audit Process (Clause 7.2.7)
Software Problem Resolution Process (Clause 7.2.8)
Software Reuse Processes
Domain Engineering Process (Clause 7.3.1)
Reuse Program Management Process (Clause 7.3.3)
Reuse Asset Management Process (Clause 7.3.2)

Fig. 2: *Life Cycle Process Groups [6]*

ISO/IEC 12207 requires outputs and these outputs have to be documented, but there is no specific or predefined format for any types of output to be documented. The organization can use their documentation methods also get benefit from standards. In addition to documentation base-lining is important issue. The standard requires the baselines of software related tasks and activities such as software requirements, software design, and coding. Baselining is a process in which quality and effectiveness of a method assessed by comparing before and after a change. The idea of *baselining* can be considered as an effective method to mitigate risks that establish certainty in milestones, to control costs and deadlines by prohibiting unnecessary (i.e. unplanned or open) changes in all parts of the software development life cycle [5]. In particular, baselining can happen while joint review (or an audit process) so as to clarify and acquirer-supplier understandings. However, it is not necessary for projects to perform baselining, which is the responsibility of the Development Process. It is not related with Configuration Management Process and it is not a must. Consequently, ISO/IEC 12207 covers the total software development life cycle and it is relatively a complex process especially when it is based on a viewpoint of variety of stakeholders who are working together in the same software development project.

However, the standard should be examined in the context of organizational objectives where the requirements of a project may hinder a solid interpretation. To avoid improper usage of the standard, these prerequisites below shall be met [5]:

(a) The requirement of qualified personnel;
(b) The requirement of understanding the organization's policies;
(c) Experience with the project's environment;
(d) Develop an understanding of the standard.

2.2 Customization of ISO/IEC 12207

Although ISO/IEC 12207 software lifecycle standard shows a set of agreements of experts on some procedures for software development, there is no one-size-fits-all type of selection and tailoring of processes. However, the responsibilities of both acquirer and supplier are considerably important in the tailoring process. According to the *acquisition process* defined by ISO/IEC 12207, *"success of the supplier depends largely on explicit definition of the acquirer's expectations, in terms of system requirements and with respect to the software development process" [11].*

There is a common point of view to the quality of any software product largely depends on the quality of the software development process [12]. In fact, all successful companies in any size should follow well-defined activities and tasks. Another factor that increase the success of further development and reduce the current project's problems is receiving adequate feedback from both user and prior projects. To accomplish these factors and to improve the development process, companies should invest on software process improvement activities, however a number of companies still rely on the success of the employment of ad-hoc processes, which relies on individual's skills. Such processes are very difficult to reuse. In addition, they may have an adverse effect on the quality, maintainability, cost of a product [11]. Moreover, too many reasons can be stated for a failure of process because of the absence of enough knowledge, lack of customer feedback and market related problems.

To overcome risks, problems, and inadequate development process and to improve the quality, there is a strong need to follow some standardized methods. Due to these reasons ISO/IEC 12207 is one of the recent and valid to use standard that includes the steps that should be followed by contractors. Tailoring process shows itself here more clearly at the acquisition process and acquirer has a chance to tailor ISO/IEC 12207. This process is totally good for providing guidance for quality related activities by mitigating risks and ultimately crucial to project success. Under tailoring circumstances such as novelty, size, budget, risks, technology, time etc. shall be inspected.

3 Concept of Serious Games and Examples

A serious game is an interactive approach, which is designed for a purpose other than pure entertainment [13]. A goal of a serious game is usually improving an educational aspect where participants certainly attend such activities with such an expectation. These interactive application are widely preferred in training and education for medical and military personnel. Recently, serious games become more popular and therefore they are now found in any size, complexity and platform similar to casual games. The education aspects of these interactive applications are heavily depend on the notion of play which is an importnat factor for individuals development and learning [7].

In addition, serious games are kind of simulations of real-world events or processes that are addressed to comprise particular problems. Therefore, they can be considered as serious activities such as exploring, training or advertising. However, they still can be entertaining, if their main purpose covers game elements well. Substantially, games have many attributes which have been seen in the case of different examples. For instance, serious games allow user to experience different learning tasks by using the elements of fun. Another example of attributes is stating how actions affect the context. Players can create artifacts or complete tasks within in the orders of a serious game serves and without the effects of real world problems and stress. This can be interpreted to resembling sand box type games. Moreover, serious games allow users have an active participation while accomplishing the main goal. In fact, games are powerful tools, because they have the ability to change human behavior [14]. Furthermore, games can help users with repetitive actions while learning certain subjects. Because particular tasks and clearly stated objectives of serious games make player easier to follow certain pathways and play their role for a set of planned behaviors. Such planned behaviors can be easily linked to the learning process where gaming may assist and ultimately create a user-oriented learning experience.

There are numerous works which are related about serious games and applications. However in the literature there are only several serious games which are related to software project management. These are; Problems and Programmers [15], SIMSOFT [16], SimSE [17], SESAM [18], DELIVER [19], ProDec [20].

Problems and Programmers [15], is an educational (serious) card game, which simulates several software engineering processes. It is designed as a teaching tool to improve the students' understanding of the software processes. The goal is to teach software processes and issues that are encountered in software development, which may not sufficiently highlighted by software engineering lectures and projects. Among computer based digital variants of serious games, Problems and Programmers is created as a non-digital card game where two or more player are trying to finish a software project. By using the benefits of card games, the game state and action of the players are always visible. The game is designed as a competitive game where participants may have a chance to learn from each other [21]. These attributes strengthen the main goal of the game and reveal the games powerful sides such as (i) covering proper use of software engineering techniques, (ii) providing clear and instant feedback of players choices and decisions towards phases, and (iii) encouraging interactions among different players and evaluating their perspectives. According to Baker, *"Because different players follow different strategies, more than one strategy is exposed per game. This allows players to not only evaluate their own strategy but to also discuss and compare strategies followed by others. As a result, players learn from each other, which enhances the educational value of Problems and Programmers"* [21].

SimSE [17] is a single player serious game that is design to software engineering in an interactive (graphical) environment. The participants take on the role of a project manager who is responsible from a team of software developers. The game allows players to manage the software engineering process by recruiting engineers, assign them to a set of tasks, monitor their progress and ultimately gain (virtual) experience on managing a software development project. A significant reason for creating educational software engineering applications is that during the courses, computer science and engineering students from are only exposed to the theoretical concepts of software engineering. There is no sufficient or relatively sufficient project which converts lectures into practices. SimSEs main goal is fulfilling this absence by providing 2D graphical virtual office where software engineering processes take place. This office scene includes many office staff such as computers, desks, employees, and artifacts. These approaches clarify the players' actions with the support of selection of the available moves and steps the player will proceed. Perhaps the most important feature of SimSE (among other variants) is that it has a model builder which allows user to create employees, artifacts, tools, projects, and customers without any need of programming skills. Bu using such a modeler, SimSE supports customization of the software processes based on real-world scenarios about software projects.

Project Decision (ProDec) is a simulation-based serious game created with the intention to train and assess students in software project management [20]. The main objective is to take advantage of the engaging nature of games to place the learners in a virtual organization where they can manage software projects and solve real-life problems in a risk-free environment [20]. The main goal of the game is to remain aware of planning, controlling, and managing a software project. The game is over to the extent permitted by the amount of budget the players have and allocated time for the project. While gameplay players need to plan to deal with obstacles which are created by unplanned events. ProDec is intended to be a collaborative game, that is, it is a game to be played by teams of players [20]. This means that the group of players works collaboratively to win the game not to compete among them [20]. After any game play, ProDec offers a complete report including the logs representing every decision the players made and the result of applying the assessment criteria provided by the trainer at the beginning of the game play [20].

Simsoft [16] is a kind of serious game which consists of two game boards, a printed board and a digital board. A printed board is used to gather participants to discuss the actual state of the game project and decide their strategies. The staff is represented by plastic counters while game chips as virtual money, which is used for resources such as project budget. The players make a set of meaningful decisions and spend their resources during the game. In addition to a printed board there is a digital game board which shows several game stats, player's history and reports. The goal is to complete a small software development project in a given set of time frame. The participants are grouped into small set of teams to manage the software development from start-up to final delivery [16]. The players have the opportunity to observe the difficulties of resource management where the skills of the hired personnel affects the outputs of a software project.

In addition, SESAM is a natural language based serious game which motivates players for learning software project management techniques. SESAMs environment consists of natural language interface, records about program and statistics about project.

Lastly, DELIVER is another type of serious game which consists of a printed board. It helps students to develop controlling projects performances. Its main ambition is totally motivate students in their learning progression.

Defining a solution of improving the managerial skills in software project practitioners by using a game environment may somehow lead disagreement. Some researchers may claim that the concepts of software engineering could be hard to be thought by using a game system. However, Gee [22] suggest that *"good video games incorporate good learning principles, principles supported by current research in cognitive science."* In fact, a game may encourage the process of deep learning [23] in which participants may improve their thinking skills about a complex situation by becoming successful in a serious game. In other words, if a cycle of learning has a chance to improve itself in a more enjoyable way, the interaction process become easier to everyone. In particular, a goal of a serious game is to make people think that they can shorten their learning time in a complex topic by putting some effort into improving their skills in a video game.

4 Discussion on planning the ideal game

Typical software engineering methods and principles are not quite easy to learn and implement because of the lack of motivation and absence of fine learning period in lectures. In addition to this, technical developments and evolving industry demands change rapidly and there is a need for learning broader concepts about software engineering and management tasks, rather than simply getting general knowledge that was given in every introductory level software engineering course. The problem is not simple as being only hard-to-learn. The main point is being unable to make easier to understandable for everyone either managerial or technical levels of software engineering topics in the market. Due to this no consistency of an adequate education there are too many projects have been struggling to go further.

Undoubtedly, getting the big picture of today's current circumstances and state of the software engineering and project management are difficult, but obvious thing is the failures and problems are becoming more visible for everyone to recognize. To reduce failures and problems there are crucial jobs which are waiting for project managers to accomplish. Because project management skills are the key factors which have potential to overcome risks in development environment, and this skills may lead the way of engineers, developers and even students who are the future responsible of life cycle of software. However, current view of project life cycles in any kind of software development implies that managerial skills need more practical experience to become mature enough to produce more stable projects with less risks and problems.

From the student side, this problem can be restated with similar reasons. Although being the future engineers and managers there are too many inconsistencies between market and syllabuses. In fact, software development organization desires to see better skills from university graduates while there is a lack of software engineering skills taught at university level [24]. Apparently, many software engineering concepts are given to students in a more theoretical sense. In contrast, this should be more practical with supporting projects which resemble to real life scenarios. Even they come together there is a need for in depth procedure to follow. Navarro et al. [24] states that "*In particular, lectures allow only passive learning, and the size and scope of class projects are too constrained by the academic setting to exhibit many of the fundamental characteristics of real-world software engineering processes [24]"*.

To deal with these problems one possible and feasible way is using serious game which is suitable for specific concept. Serious games are designed to teach especially educate players about desired topics in a well-defined environment [7].

To define the learning process in a clear sense, it can be repeated in any time without practicality, because in theoretical approach information is static not dynamic so there should be actions in several ways to change it to dynamic which helps to resolve the problem. No matter what the type is every game should clearly define some attributes while game play even before. This is more decisive when the serious gaming concept is under concern, because every player who is playing a serious game he/she ought to be conscious about what will he/she do, accomplish and learn. Desired commitment to the learning journey requires well defined and planned program. For instance without clearly defined characters or personalities in the game it is difficult to expect anyone to commit himself to a specific progress. Moreover the new virtual environment serves players living and acting with their commitment and this is the evident for most games success in any theme and ambition. In a software project development environment with a serious game it is necessary clarify exactly what are the characters, who are they, and what are the capabilities that they have. This clear sense is the primary factor for a player to commit himself in such an environment at first.

There is a degree of uncertainty around the terminology in being interactive with games. This shows a need to be explicit about exactly what is meant by the word interactive. Furthermore, this is more difficult for anyone at first to estimate the literal meaning of being interactive in the concept of serious game for ISO/IEC 12207. Being interactive comes from interaction so any kind of game have to provide a dialogue to player. In fact if there is no such an interaction, how do players accomplish task? The instant feedback and continuous active reactions are the wanted factors to define the term being interactive. Undoubtedly, it is believed that every subject of in any science must be readable. This is the case over years and every person accomplishes many lectures and passes their exams via reading the related resources. Texts and textbooks have to in a daily life, but in wider concepts such as engineering and medicine conceptualization needs to contain more practical program. To sustain ISO/IEC 12207 in a serious game environment the ambition of being interactive is possible with dynamic reactions between players and the game rather than text based learning.

There is an increasing desire about sandbox type games recently and this is another factor how a game should be. In games, sandboxes are safe environments where everyone can improve or destruct everything they create before without any stressful decision which is about the things can go wrong because it is isolated from outside world and therefore the effects of outside world can be minimized. This is an important factor for project managers to train software practitioners with respect to various bad experiences. Serious games have a potential to enhance this practice, because while playing a game usually players create different virtual careers based on their own interests and selections. They modify the ongoing progress according to their point of view to the concept, so any practitioner could be trained in a game setting, where they can learn from virtual situations where they make meaningful choices without experiencing negative outcomes.

5 Conclusions

This preliminary study investigates the need of a serious game from an industrial perspective. Initial results from the semi-structured interviews suggest that a serious game can improve the ability of learners of ISO/IEC 12207 standard. All five interviewees (n=5) agree that an interactive learning approach should make it easier to teach the standard. Moreover, there is a consensus among the participants that such an approach will provide a better understanding of the documented concepts. One interviewee recommends that a serious game should be in a simulated office environment (by creating a set of situations) in which several complex concepts can be explained in an iterative way. One other recommends that a usability study should have to be conducted to address user perceptions of such an approach. In addition, she suggested that the potential for learning and engagement should somehow be measured. However, a major limitation of this study is the limited number of experts, therefore, caution must be applied. To evaluate the benefits of the proposed serious game, more research should be conducted on the functionality of the production process and ultimately with the end-product.

References

1. Tsui, F.F.: Essentials of software engineering. Jones & Bartlett Publishers (2014)
2. Futrell, R.T., Shafer, L.I., Shafer, D.F.: Quality software project management. Prentice Hall PTR (2001)
3. Yilmaz, M.: A software process engineering approach to understanding software productivity and team personality characteristics: an empirical investigation. PhD thesis, Dublin City University (2013)
4. Jones, A.: Iso 12207 software life cycle processes fit for purpose? Software Quality Journal **5** (1996) 243–253
5. Singh, R.: International standard iso/iec 12207 software life cycle processes. Software Process Improvement and Practice **2** (1996) 35–50
6. ISO/IEC: Amendment to ISO/IEC 12207-2008 - Systems and software engineering Software life cycle processes. (2008)
7. Abt, C.: Serious games. University Press of Amer (1987)
8. Alvarez, J., Djaouti, D.: Introduction au serious game. Questions théoriques (2010)
9. Acuna, S.T., Juristo, N., Moreno, A.M., Mon, A.: A Software Process Model Handbook for Incorporating People's Capabilities. Springer-Verlag (2005)
10. Yilmaz, M., O'Connor, R.V., Clarke, P.: Software development roles: a multi-project empirical investigation. ACM SIGSOFT Software Engineering Notes **40** (2015) 1–5
11. Demirörs, O., Demirörs, E., Tarhan, A., Yildiz, A.: Tailoring iso/iec 12207 for instructional software development. In: Euromicro Conference. Volume 2., IEEE Computer Society (2000) 2300–2300
12. Clarke, P., OConnor, R.V.: An approach to evaluating software process adaptation. In: Software Process Improvement and Capability Determination. Springer (2011) 28–41

13. Khosrow-Pour, M.: Encyclopedia of information science and technology. Volume 1. IGI Global (2008)
14. McGonigal, J.: Reality is broken: Why games make us better and how they can change the world. Penguin Pr (2011)
15. Baker, A., Navarro, E.O., Van Der Hoek, A.: An experimental card game for teaching software engineering processes. Journal of Systems and Software **75** (2005) 3–16
16. Caulfield, C., Veal, D., Maj, S.P.: Teaching software engineering project management–a novel approach for software engineering programs. Modern Applied Science **5** (2011) p87
17. Navarro, E.O., van der Hoek, A.: Simse: An interactive simulation game for software engineering education. In: CATE. (2004) 12–17
18. Drappa, A., Ludewig, J.: Simulation in software engineering training. In: Proceedings of the 22nd international conference on Software engineering, ACM (2000) 199–208
19. von Wangenheim, C.G., Savi, R., Borgatto, A.F.: Deliver!–an educational game for teaching earned value management in computing courses. Information and Software Technology **54** (2012) 286–298
20. Calderón, A., Ruiz, M.: Prodec: a serious game for software project management training. In: ICSEA 2013, The Eighth International Conference on Software Engineering Advances. (2013) 565–570
21. Baker, A., Navarro, E.O., Van Der Hoek, A.: An experimental card game for teaching software engineering. In: Software Engineering Education and Training, 2003.(CSEE&T 2003). Proceedings. 16th Conference on, IEEE (2003) 216–223
22. Gee, J.P.: What video games have to teach us about learning and literacy. Computers in Entertainment (CIE) **1** (2003) 20–20
23. Gee, J.P.: What video games have to teach us about learning and literacy. Macmillan (2014)
24. Navarro, E.O., Baker, A., Van Der Hoek, A.: Teaching software engineering using simulation games. In: ICSIE04: Proceedings of the 2004 International Conference on Simulation in Education. (2004)