A Literature Survey: Is it Necessary to Develop a New Software Development Methodology for Virtual Reality Projects?

Article *in* JOURNAL OF UNIVERSAL COMPUTER SCIENCE · November 2017

3 authors:

Ulas Gulec
TED Universitesi

Murat Yilmaz
Gazi University

Veysi Isler
Middle East Technical University

Journal of Universal Computer Science, vol. 23, no. 8 (2017), 725-754
submitted: 4/5/17, accepted: 26/8/17, appeared: 28/8/17

A Literature Survey: Is it Necessary to Develop a New Software Development Methodology for Virtual Reality Projects?

Ulas Gulec
(Department of Computer Engineering, Middle East Technical University
Ankara, Turkey
Department of Computer Engineering, Çankaya University, Ankara, Turkey
ulasgulec@cankaya.edu.tr)

Murat Yilmaz
(Department of Computer Engineering, Çankaya University, Ankara, Turkey
myilmaz@cankaya.edu.tr)

Veysi Isler
(Department of Computer Engineering, Middle East Technical University
Ankara, Turkey
isler@ceng.metu.edu.tr)

Abstract: Software development is a complex human endeavour with high failure rates. Although a variety of software development methodologies have been proposed to improve the software development process, there is no universal model for all software development organizations. Virtual reality (VR) is an emerging trend especially for the gaming industry, which should prepare itself for VR development. The goal of this study is to explore potential software development activities and determine whether designing a new software development methodology for VR projects is an important topic for software development organizations working on VR software development. For this purpose, a literature survey has been completed and 71 academic studies have been examined in detail. This study shows that no work is being conducted in the field of developing a new methodology for VR projects. However, the study does show that there are similar endeavours in the field of human computer interaction (HCI), such as game development methodology.

Key Words: virtual reality, software development methodology, literature survey
Category: D.2, I.6.3, L.5

1 Introduction

VR is a technology that generates 3D environments in which users can highly interact with various input and output equipments [Jayaram et al., 1997]. The aim of this technology is to immerse a user in a computer-generated environment and provide the feeling that they are physically in this environment [Heldal, 2007]. Hence, it provides a beneficial and immersive training environment that includes real-life conditions for individuals [Sampaio and Martins, 2014]. Virtual environments are used to increase the level of knowledge and experience of people working in many different fields, and to experience the problems and events that they may encounter in real life [Cheng et al., 2014]. Due to this property, in the last two decades, VR becomes a very popular area [Merchant et al., 2014] that provides opportunities in different domains [Schroeder, 1996].

With the increasing popularity of virtual reality, the amount of investment made in this area has been increasing proportionally. In 1996, UK companies allocated between £10,000 and £60,000 to explore this new emerging technology and develop new applications in this field [Bouchlaghem and Liyanage, 1996]. In 2013, the consortium of European countries allocated a budget of €15 million to create virtual reality projects [Nazir et al., 2013]. In support, Delaney [Delaney, 2014] points out that the US Army has allocated a budget of US$ 3 billion to develop a virtual environment for the training of military personnel. In 2014, *Facebook* invested US$ 2 billion to develop the *Oculus Rift*, which is commonly used in VR environments [Barnes, 2016].

Although VR has become more popular and the companies have invested much money to develop applications in this domain, to the best of the author's knowledge, there is no specific and applicable software development methodology for VR projects. The aim of this study is to determine whether it is necessary to generate a new software development methodology for VR applications. In order to do so, 71 studies that were carried out in the field of software engineering were studied in detail.

The remainder of this paper is organized as follows: Section 2 begins by explaining what software and software development are, and it discusses the importance of software engineering, software development processes and methods, and why many software development methods are needed. In addition, in this section, the paper explains in detail the software development methodologies developed for the HCI domain. In Section 3, the methodology of the research including search techniques, properties and the areas of expertise of the studies are described. The following section examines the results obtained from the studies in the literature. In section 5, the discussion of the results is illustrated. Finally, the last section explains the conclusion and the future work of this study.

2 Background and Related Works

Software is an expandable, functional and programmable product that can run on a computer [Humphrey, 1989]. Although the definition of software appears to be simple, the concept of software should be considered from an engineering point of view since the structure of software including analysis, design, development and testing steps is very complicated [McConnell, 2001]. Hence, software is a product obtained after several long processes. At this stage, the definition of software development becomes more meaningful since software does not have a simple structure. It is a requirement that the development of a piece of software should be significantly completed within a certain logical framework to produce a successful product.

In 1971, Weinberg [Weinberg, 1971] described software development as a human activity that consists of design, creation, implementation and maintenance phases. It may also be considered to be the combination of several significant phases that lead to transforming customer requirements to a software product [Johnson-Laird, 1992]. Asad et al. [Asad et al., 2004] list the phases of software development as a software development life cycle (SDLC) in their study. These phases are *requirement analysis, design, implementation, testing* and *maintenance*. Each of them has an importance to complete software projects successfully. The requirements analysis plays a key role in the development of software to develop more accurate and less costly products since it is the first phase of SDLC. Based on the CHAOS report declared by the Standish Group in 2015, it has become more critical nowadays since the failure rates in IT projects are very high due to determining the wrong requirements [Kannan et al., 2015]. This issue is essential for software companies, because, if an error in a system is detected at the last stages of SDLC, it is more costly for companies to repair it at the final stages than to repair it at the beginning stages [Subhan and Bhatti, 2015]. The second step in the project development process, the design phase, is also crucial in establishing the basis of the project in a robust manner since the number of logical errors between 36% and 74% is detected in this phase [Karimi, 1983]. The implementation phase is one of the most important steps in the project development process as it directly affects the quality of the end product since the coding and algorithm parts of the system are classified and constructed in this phase [Alonso et al., 1998, Dart et al., 1987]. Any wrong action made in this step will lead to the failure of the project or increase the work load in the testing phase. Hence, the testing phase is a significant step in SDLC since it improves the quality of the product by finding and fixing the missing points of the system [Lun and Chi, 2013]. Finally, in the maintenance phase, customer expectations are met by applying several different types of improvements and changes to make the system work properly [Fu et al., 2015, April and Abran, 2012]. If a product cannot work as expected, the entire project is deemed a waste of time.

These studies in the literature show that each phase of SDLC plays a critical role in developing useful and successful products. Hence, software development is an important issue such that its phases should be thought of and carried out as engineering concepts in order to produce measurable, evaluable, reusable and replicable products which have concrete outputs [McConnell, 1998]. Although it is an essential issue, most software projects fail and remain unfinished [Braude and Bernstein, 2016]. This means that most companies waste both time and money developing completed software. In order to avoid such undesirable circumstances, the software development process should be designed based on an engineering point of view since the end product requires systematic development.

The systematic development within the engineering perspective creates the "*software engineering*" concept. Software engineering is a set of engineering activities that manages the processes in SDLC in order to produce high quality software products [Ghezzi et al., 2002]. It is a standardized method to solve problems that occur in the development of software by organizing the SDLC phases [Singpurwalla and Wilson, 2012]. It improves the performance of software within a systematic scheduling to achieve software quality [Nielsen, 2015]. Software quality measures both the software development process and the software product based on a number of factors [Ming-Chang, 2014]. The studies [McCall et al., 1977, Boehm, 1978, Grady, 1992, Dromey, 1996] in the literature analyse and list these factors as: *Correctness, Reliability, Efficiency, Integrity, Usability, Maintainability, Flexibility, Testability, Portability, Reusability, Functionality, Operability, Compatibility, Modifiability.*

The studies in the literature illustrate that software should be developed within a methodology that is systematically designed to meet these quality criteria. However, the methodology can differ based on the properties of software and company resources. Therefore, companies need to manage the SDLC phases according to their resources and the requirements of their customers in order to provide those software quality criteria. Thus, there are several different software development methodologies which help organizations to produce cost-effective, qualified, successful projects [Ben-Zahia and Jaluta, 2014, Khan et al., 2014, Kaur and Sengupta, 2013]. The commonly used methodologies are: *Waterfall Development, Prototype Development, Iterative and Incremental Development (IaI), Spiral Development, Rapid Application Development (RA), V-Shaped Model (V-S)* and *Agile Development.*

It can be easily observed that there exist several different software development methodologies. This is due to the fact that none of them can be applicable to every type of project since they cannot provide every requirement for every type of project [Han and Xie, 2012]. Hence, the software development methodologies have both advantages and disadvantages since their components differ from each other [Leau et al., 2012].

There are various studies in the literature which compare software development methodologies. Saxena and Upadhyay [Saxena and Upadhyay, 2016] conducted a study that compared waterfall and prototype development methodologies. In this study, the total development time, expenditure to develop the project and user satisfaction are designated as factors that determine whether a project is successful. In the light of these factors, both methods have a number of advantages and disadvantages similarly to the other software development models. The advantages of the waterfall development model can be listed as having well defined stages, being easy to understand and easy to use, and having a simple implementation process. In addition to these advantages, this model requires minimum resources to complete a project in comparison to the other models. On the other hand, this model also has a number of disadvantages, the first of which is that the system cannot be used for requirements not occurring at the beginning of the development stages. Another disadvantage is that this model does not allow users to be involved in the development process of a system from the beginning. In the prototype development model, users are included in the project development process from the beginning of the project. This means that the users can continuously give feedbacks about the project. Therefore, errors may be easily detected. In summary, the waterfall development model is suitable for simple and well-defined projects which require minimal updates when the prototype development model should be used to develop more complex systems due to its dynamic and flexible structure.

Drury-Grogan and Kennedy [Drury-Grogan and Kennedy, 2013] compare the waterfall and agile development models in their study. According to this study, the waterfall model enables project members to focus on the project tasks, which occur in a linear sequence. A new task follows after an existing task is completed. Thus, the aim of project members is to complete the task assigned to them within the time schedule. For this reason, the project can be easily managed. Although this can be seen as an advantage, it also brings some disadvantages. The first disadvantage is that the different teams are generally assigned to each phase of SDLC in the waterfall development model. This situation causes a continuous lack of sharing of knowledge and information among team members due to every team member focusing on his own respective task. Thus, there is no strong interaction between team members. Another disadvantage is that the project is tested in the last stages of SDLC. This is a significant deficiency of this method since nearly every project requires reworking. In addition, if a company wishes to rework a project to rectify problems, the operation will be costly for the company. The agile development model fills these lacunae in the waterfall development model. It allows for an iterative development of the project since the process of this method is dynamic. Therefore, any new demands from customers may be more easily performed. Short-term tasks are assigned to small teams which have high-level interactions between members.

A significant analysis and discussion on comparing the software development methods was presented by Moniruzzaman and Hossain [Moniruzzaman and Hossain, 2013]. This study illustrates the deficiencies of existing traditional software development methods by comparing them with the agile development model. According to this study, traditional software development models feature the importance of planning instead of adapting customer requirements, which seem to occur after a project starts. This situation makes the projects more manageable; however, the studies in the literature [Nerur et al., 2005, Dyba and Dingsoyr, 2009] indicate that easy management of a project is not meaningful if it does not satisfy customer expectations. Traditional methods attempt to guarantee to finish a product based on the requirements determined at the beginning of the project, whereas the aim of the agile development model is to integrate the customers' new requirements into the system rapidly. Consequently, more qualified products can be produced at the end of the development process. Additionally, the agile development model brings other advantages. The first additional advantage is that it divides the objectives into the small tasks. Thus, the time needed to complete a task is short. This provides rapid development in addition to a rapid testing environment. Another advantage is that customers are included in many aspects and steps of the project development process. This produces opportunities for customers and developers since customers can see any missing or undesirable parts of the projects, thereby enabling developers to deal immediately with any deficiencies in a project. The last advantage is that this model reduces the total reworking cost since the errors in the system are detected in the early stages of the software development process.

Another comparative study completed by Mishra and Dubey [Mishra and Dubey, 2013] illustrates the similarities and differences between the waterfall, V-Shaped, spiral and rapid application development methods by explaining both the advantages and disadvantages of each software development model. According to this study, the waterfall development model cannot be applicable to a project in which requirements are dynamically changed or updated. The other models were discovered to fill this gap. For example, in the spiral development model, the developers start with a small set of requirements at the beginning of the project and, if necessary, add new requirements in the following stages. However, this dynamism causes delays in the delivery of projects if the manpower is not adequate to complete any complex systems. The V-Shaped model is an improved version of the waterfall development model in terms of the testing phase. Although testing occurs after each phase, the requirements cannot be easily included in the project development phase. The rapid application development model can have projects finish in a short-time period if the requirements are well-defined; however, the error rate may increase in the end product.

The results obtained from the comparisons are summarized in Table 1. In this table, "*-1*" means that the related model has a disadvantage on the indicated property, "*0*" means that there is no advantage or disadvantage for the related model regarding that property, and "*1*" means that the model has a positive manner for that property.

Table 1 illustrates that the existing software development methodologies can-

	Waterfall	Prototype	IaI	Spiral	RA	V-S	Agile
Usable	1	0	1	-1	1	1	1
Understandable	1	1	1	-1	-1	1	1
Manageable	1	-1	1	-1	-1	1	-1
Cost-Efficient	-1	-1	1	-1	-1	-1	-1
Qualified	0	1	-1	1	1	1	1
Modifiable	-1	1	1	1	1	-1	1
Flexible	-1	1	1	1	1	-1	1
Customer Oriented	-1	1	1	1	1	-1	1

Table 1: *Advantages and disadvantages of existing software development models*

not satisfy all non-functional requirements in which the end product should contain. This means that each of them has some disadvantages on the indicated property. Hence, companies need to choose correctly while determining which methodology should be used in their projects in order to avoid loss since the benefits provided by each methodology are different.

When the application areas of VR are examined, it is observed that there are several different VR projects in different domains including education [Lorenzo et al., 2016], sport [Donath et al., 2016], psychology [Castro et al., 2014] and medical [Yiannakopoulou et al., 2015]. In addition to this situation, with the progress of technology in VR over time, both the number of VR projects, the number of companies that deal with VR projects and the number of domains where VR projects are completed are increasing day by day [Gregory, 2017]. In support, Pantano [Pantano, 2014] indicates that many businesses are investing in this area to take advantage of the benefits of VR technology. However, even though this area receives a lot of investment, there are still several disadvantages of this technology [Jimeno and Puerta, 2007]. In some cases, the capabilities of VR technology may not be adequate to provide the properties of the complex systems [Maddox et al., 2017]. Especially in 3D games, since the players have motion sickness problem when they move too much with 3D glasses, this situation makes the games more stable by lowering the quality of the games [Chenglei et al., 2016]. These studies show that despite the widespread and effective use of VR applications in many areas, there are still some shortcomings of this kind of applications. Hence, their effect can be further increased by a systematic project development process in order to prevent such undesirable cases and to increase the quality of the VR applications.

Although the existing software development methodologies are available to be applied to the projects, there are a number of reasons why they cannot be successfully applied to developing VR applications. The first reason is that VR is a multidisciplinary area due to the inclusion of artificial intelligence, computer graphics, art and design, sounds, human factors, and physical control systems [Klinger et al., 2010]. Therefore, traditional software development methodologies cannot provide the requirements of multidisciplinary applications such as games and VR applications since they are compounds of several different areas [Aleem et al., 2016]. The second reason is that workers are generally engineers in most software projects. Jones [Jones, 2003] conducted a study to determine the occupation groups in software project development. In this study, approximately 12,000 projects were analysed. According to the result of this analysis, most engineers who are programmers work in software projects. However, workers in VR projects are from different kinds of working areas. Engineers, designers, scenarists and experts in specific fields can work together to produce successful and high quality VR applications [dos Santos Nunes et al., 2011]. For this reason, traditional software development methods, which are used to manage mostly engineers, may have some trouble when they are used to organize people in various professions. Hence, a new software development method needs to be developed in order to enhance the quality of VR projects. In accordance with this purpose, the literature has been scanned to find any studies that have been carried out to create a new development model for VR applications. However, to the best of the author's knowledge, no such exact methodology for the development of VR projects exists.

Since VR is a type of HCI system that influences users in a more immersive manner [Mujber et al., 2004, Brunnett et al., 2015], software development methodologies which have been developed for serious games, one of the interdisciplinary areas, should be analysed and explained in detail. Serious games are those games which are developed to impart something to users in an enjoyable manner [Ritterfeld et al., 2009]. This type of game is also an interdisciplinary area due to the inclusion of a number of various fields including artificial intelligence and computer graphics [Tanenbaum et al., 2013].

It can be observed in the literature that many studies have been conducted showing similarities between VR and serious games. Marsh [Marsh, 2011] states that VR is an extensive version of serious games by using more interactive techniques in the educational domain. In support, Chung and Gardner [Chung and Gardner, 2012] determine the commonalities between VR applications and computer games in their study. According to their study, using interaction methods and high-resolution 2D or 3D graphics, collaborating with participants, immersive properties and having the participant feel as though he is in the virtual or game environment inside the "*presence*" concept are listed as similarities of

VR systems and games. Stone in his study [Stone, 2009] takes a different perspective on this issue. He specifies that the understanding of how modelling and rendering are used to create 3D virtual environments leads to the development of more interactive 3D serious games since both research areas consist of the common properties such as user interaction, motion capturing with sensors, immersive 3D environments and physical tools. Ninaus et al. [Ninaus et al., 2014] summarize the studies which contain applications developed for teaching neuroscientific techniques to participants by benefiting from game-based learning properties. The numerical results obtained in this study indicate that 25 of these studies have been completed by using computer games and 10 of them have been concluded by using VR applications. Hence, both computer games and VR applications can be used for the same purpose. Another study completed by Virvou et al. [Virvou et al., 2005] shows the relationship between games and VR. In this study, a VR game called "*VR-ENGAGE*" is developed to educate participants in an enjoyable way. During the development of this game, game elements, game dynamics and game mechanics are integrated into the VR system.

These studies present the fact that games and VR environments are intertwined and include common properties. Hence, the software development methods developed for games should be analysed in detail since games are also multidisciplinary areas. First of all, Chandler [Chandler, 2009] develops a software development methodology for game design which consists of four steps: pre-production, production, testing and wrap-up. These steps do not include the specific tasks. They only cover the general tasks which have to be executed in order to complete the game project successfully. In the pre-production phase, a detailed project plan, including the aim of the game, time-schedule, the budget of the project and manpower requirements, is defined. In the production phase, the main features and assets of the game begin to be developed by planning the implementation steps. In addition, risk management and observing the progress of the project are also carried out during this phase. In the testing step, bugs and errors are fixed by the development team of the project. During this step, quality metrics are considered while the missing points of the project are being rectified. This step plays a significant role in creating a satisfactory project. In the last step of this method, the project members document their experiences and acquirements about the project development process in order to use the data in future projects. Although this study introduces a new methodology for game development, it is not suitable for more complex games or interdisciplinary projects since it requires iterative steps in these types of projects.

Ramadan and Widyani [Ramadan and Widyani, 2013] designed a new software development model for game development as a second model. This model has six phases: initiation, pre-production, production, testing, beta and release. The initiation phase is very similar to Chandler's pre-production phase in terms of defining the game. However, Ramadan and Widyani's pre-production step differs from Chandler's pre-production phase. In Ramadan and Widyani's pre-production phase, game design and game prototype notions are additionally used. Game design means that game mechanics, story, challenges, fun elements and game-play are documented. A game prototype is an initial product built upon the use of information gathered in the first design. When the design document is updated, the game prototype is updated respectively. Finally, the end product is obtained by improving the prototype iteratively. Production and testing are very close to Chandler's production and testing phases. However, Ramadan and Widyani's testing phase has an extension with a beta phase. In the beta phase, the product is tested by external users instead of the members in the development team. The users' feedback and error reports are gathered during this step. The release step, which is the final step of this model, contains documentation operations for future work.

The third model was developed by Blitz Studio [Studio, 2014]. This model also contains six phases: pitch, pre-production, main production, alpha, beta and master. In the pitch phase, a feasibility study is completed to determine whether the project is useful, necessary and successful at the end of the project. After deciding this issue, the project is planned and scheduled in the pre-production phase, similarly to the other development methods. When a definite development plan is established, the project is implemented in the main production phase. In this phase, a version of the game, which can also be referred as the prototype, is presented to the customer every month or every six week in order for them to have the opportunity to follow the production process closely. The project is updated by receiving feedback from the customers after each demonstration and these updates are documented for the purpose of solving any problems of adaptation of the people newly included in the project. This process continues until the alpha stage, when the game becomes playable, notwithstanding any deficiencies in the game design. The project continues to be updated and upgraded in this phase until the system reaches beta level, that is, until the game design is completely finished. During the beta phase, user and game-play tests are performed to detect bugs and errors in the system and these faults are reported and corrected. In the master stage, which is the final stage of the project, the final product is produced with respect to customer specifications by completing user tests and correcting, as much as possible, any faults in the projects.

McGrath [McGrath, 2014] provides another methodology for game development. This study is an extended version of the agile methodology. According to this study, the planning phase in the agile methodology is not organized very well such that employees in the development team are forced to work more diligently in order to complete the project on time. This study suggests that teams in game projects are usually composed of small teams, and therefore, putting such teams under such pressure would regularly ruin the process. In order to overcome this difficulty, a new methodology, consisting of a total of six steps, including design, develop, evaluate, re-develop, test, release and repeat, has been developed. As is the case with other methods, after the general design of the game has been made during the design phase, the process undergoes the development phase. The goal at this stage is to create a prototype that can be evaluated. Once the resulting product is tested according to universally used standards by project staff, the correction of errors obtained from the tests occurs during the redevelopment stage. At this stage, while any errors are being corrected, information is documented as to what changes were made and which methods were used to correct them for future projects. The test phase starts after the error correction finishes. In this phase, user tests are performed and user comments are retrieved and, if any, new errors are found and corrected and the materials used in the project are updated according to user feedback. Finally, a well informed and documented, vigorous and useful end product is produced based on the idea in this study.

The last game development methodology completed by Hendrick [Hendrick, 2014] consists of a five-phase process life cycle including prototype, pre-production, production, beta and live phases. At the prototype stage, the subjects that are decided upon include which tools are to be used to develop the project, with which programming language it will be coded, how to design the animations to be used, and how to have an interface. Once these issues have been agreed upon, the design and coding of two or three zones of the game are completed in the pre-production phase so that an idea of the final product of the project can be determined. In the production phase, all remaining zones are completed, including all sub-fields such as data entry, screen designs, and artificial intelligence scripts. A beta version of the game is offered to users so that the finished product obtained in the production phase can be tested by users. During the beta phase, faults in the product are regularly reported and corrected. Finally, an updated version of the game is published to users.

3 Methodology

This study aims to provide us with an understanding of the state of the art of VR in software engineering. In addition to this purpose, the importance of software engineering and software development methodologies, the existing studies and the reasons why the actual methods cannot be applicable to VR projects are determined. The main research question of this study is:

RQ: *Is it necessary to develop a new software development methodology for virtual reality projects?*

The systematic mapping of this study was carried out according to the instructions in [Kitchenham et al., 2011, Petersen et al., 2015]. This section explains the steps in the systematic mapping that are the additional research questions, the search strategy, the selection technique, the data sources and the evaluation strategy. The 71 studies were found to meet these criteria.

The literature searching process of the study is shown in Figure 1. According to this scheme, the searching process started with determining the main research question. After defining the main research question, the sub-research questions were established in order to create the background of the study. When the aim of the research was concluded, the search operation was launched to find the relevant studies. Before the first filtering, a total of 783 studies found. In the first filtering, the duplicated entries were removed and the irrelevant studies found by reading the title, keywords and abstracts of studies were also eliminated. At the last step, the full text of the remaining 218 works was read and 71 related works emerged.

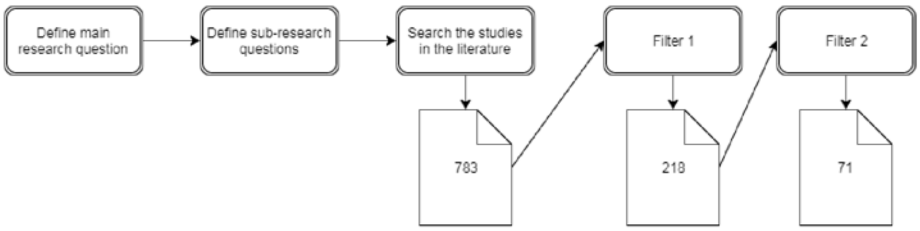

Figure 1: *The literature searching process.*

3.1 Research Questions

In order to be able to respond to the main research question and establish the substructure of the study, a literature search was carried out to provide answers to the following sub-questions:

RQ.S1: Why do we need software engineering?
RQ.S2: Why do we need software development methodologies?
RQ.S3: Why do we need several different software development methodologies?
RQ.S4: Why are there existing software engineering methodologies that are not applicable to VR projects?
RQ.S5: What are the specific software development methodologies developed for HCI projects?

It is important to clarify the answers to these research questions since it is not meaningful to propose the importance of a new software development methodology for VR projects without mentioning the underlying structure of whole software projects. In order to create the substructure of the study, it is necessary to figure out the importance of software engineering and software development methodology. In addition, it is also critical to illustrate why there exist different software methodologies and why they may not be successfully applied to VR projects. Consequently, it is essential to analyse specific software development methodologies developed for HCI projects since VR is a kind of HCI system that includes more interactive techniques.

3.2 Search Strategy, Selection Technique and Data Sources

In order to find the relevant studies, the search operation was completed with four main categories, these being *SDLC*, *software development methodologies*, *VR* and *HCI*. The first category, SDLC, was selected to identify the steps that should be taken in a new software development methodology and to determine which tasks in the project should be carried out during which phase of the methodology. The reason for researching software development methodologies is to understand what the commonly used methodologies in the literature are, their advantages and disadvantages, and why problems can arise when they are applied to VR projects. VR was selected as a third category since it is important to determine the definition, its properties, and application areas and whether it is a software development methodology designed for VR projects. Therefore, this information about VR can give us an idea of whether a new methodology is needed, and if so, how to design this new methodology. The last category, the HCI area, was chosen to find and illustrate software development methodologies designed for HCI projects due to the fact that VR is a sub-category of HCI and there is no applicable software development methodology designed for VR projects. Table 2 lists the strings that were used to find the studies.

In Table 2, words with similar meanings to the main terms were used as alternative terms to increase the number of studies to be crawled under the literature search. These alternative terms were linked to each other with the Boolean OR operator to indicate that they are being used instead of others in order to give the same meaning at different times. In some situations, the major terms or alternative terms of the related major terms were connected to each other with the Boolean AND operator. This means that both terms were used simultaneously in the search process.

The search strategy of this study is defined as shown in Table 3. The aim is to detect mostly journals in the selected academic databases. In addition, academic conferences and workshops were also included in the search operation.

Main Terms	Alternative Terms
SDLC	(life cycle OR software development phases OR software development phases OR software development steps OR software development process OR software process)
Software Development Methodologies	(software methodologies OR development methodologies OR software methods OR software development techniques)
	AND
VR	(virtual reality OR virtual environment)
	AND
HCI	(serious game OR game)

Table 2: *Search strings*

The search operation completed recursively. This means that the relevant studies on the reference list of studies were also examined.

In addition to the academic databases, Google was used to include non-academic works into research. Google was chosen as a search engine for the non-academic studies, because, it was the most popular search engine used in almost all systematic mapping studies. However, Google is getting a lot of results when searching for a topic. This situation causes some difficulties in finding relevant studies. Hence, it is significant to define certain selection criteria for non-academic works. Table 4 shows the selection strategy and criteria for non-academic studies. According to this table, conference and workshop papers, magazines, technical reports and surveys published until June 2017 were included to our study. On the other hand, PowerPoint presentations, personal blogs, product brochures, papers which are not related to software engineering or virtual reality were not considered to be analysed in the scope of this study.

3.3 Evaluation

A five-point Likert-scale questionnaire was administered with 5 experts in software engineering and virtual reality to measure the quality of the selected studies and contribution to the research within the research methodology. The questions asked within the scope of this questionnaire are stated as:

EQ1: Does this study illustrate the importance of software engineering?
EQ2: Does this study show the importance of software development methodologies?
EQ3: Does this study explain the reasons why there exist different software development methodologies?
EQ4: Does this study clarify the reasons why the existing software development methodologies may not be applicable to VR projects?
EQ5: Does this study use suitable search strings for finding relevant studies in the literature?
EQ6: Does this study use adequate resources for finding relevant studies in the literature?
EQ7: Does this study provide evidence showing the importance of developing new software development methodologies for VR projects?

The aim of asking these questions to the experts is to figure out whether the study meets the intended objective by taking the comments of the experts. The experts give a score between "*I do not agree (0)*" and "*I strongly agree (5)*" for each question to reflect their views. The evaluation process is question-based and the evaluation of each problem is done by taking the arithmetic average of the scores given by the experts.

4 Results

4.1 Results of Search

In the searching process, no year limitation was made since the software and software engineering bases are old. However, while investigating similar studies in the HCI space, the year 2012 was set as the lower limit, because, there is an increasing trend in publishing studies in this topic in the recent years. The distribution of the studies by year is shown in Figure 2. As shown in the figure, most of the selected studies from the literature published after 2012. This situation illustrates that the studies in the literature used in this study are up to date.

The distribution of the studies by type of forum is shown in Figure 3. When the results obtained from the search strategy are analysed, almost 60% percent of the studies are papers published in the journals. The remaining part of the studies is mostly formed conference papers and books.

The aggregated distribution of the studies in research area is shown in Figure 4. The relevant studies were categorized based on their main subjects. These categories are:

– **Software:** Since VR is a kind of software, it is necessary to figure out what the software is in order to establish the structure of the study. In addition,

Search Strategy	
Selected Academic Databases	• IEEExplore • Springer Database • ACM Digital Library
Other Data Sources	• ResearchGate • Google (only non-academic sources) • Google Scholar
Target Items	• Journal Papers • Conference Papers • Workshop Papers • Books • Others (technical reports, web-sites)
Searched Applied to	• Title • Abstract • Keywords
Language	• English
Publication Period	• Until June 2017

Table 3: *Search strategy*

it is also important to give the definition of software in order to explain why software development is required. Hence, the studies which define software and the studies which illustrate that the software is a complex structure were selected.

– **Software Development:** Since the aim of this study is to understand the importance of a new software development methodology for VR projects, it is significant to define what software development is and its importance. In addition, it is important to point out what the purpose of the software development is, because, it supports the aim of the study since we try to understand whether a new strategy for VR software development is necessary. Hence, the studies which shows the steps and aim of software development were chosen.

– **SDLC:** This category was created to illustrate the phases of software development. In addition, it is crucial to review and explain how important

Inclusion/exclusion criteria	
Inclusion criteria	A studies that consists of all words in the search string Conference and Workshop Papers Magazines, Technical Reports and Surveys Papers in written English Publication date: Until June 2017
Exclusion criteria	PowerPoint presentation Personal blogs and Product brochures Papers which are not related to software engineering or virtual reality

Table 4: *Summary of the selection strategy and criteria for non-academic studies*

they are because there will be steps in a new software development process. Hence, the studies which mention the importance of the phases of software development were selected.

- **Software Engineering:** This category was instituted to show why the phases of software development should be managed from an engineering point of view. Therefore, the studies describing the reasons for the emergence of the concept of software engineering were included.
- **Software Quality:** This category explains what features should be in developed software and how managing software development processes with an engineering perspective is important for fulfilling these features.
- **Software Development Methodologies:** This category was established to list the existing software development methodologies. The studies which compare the existing software development methodologies were chosen. In this way, we demonstrated both advantages and disadvantages of the existing software development methodologies. This situation presents the fact that although there are many other reasons for existing software development processes to be applied to VR projects, even if they are intended to be applied to VR projects, none of them can fully provide the non-functional requirements.
- **Virtual Reality:** This category was constructed to give details about VR technology. At first, the studies which define VR technology were selected. Then, the studies which mention the popularity of VR technology by showing

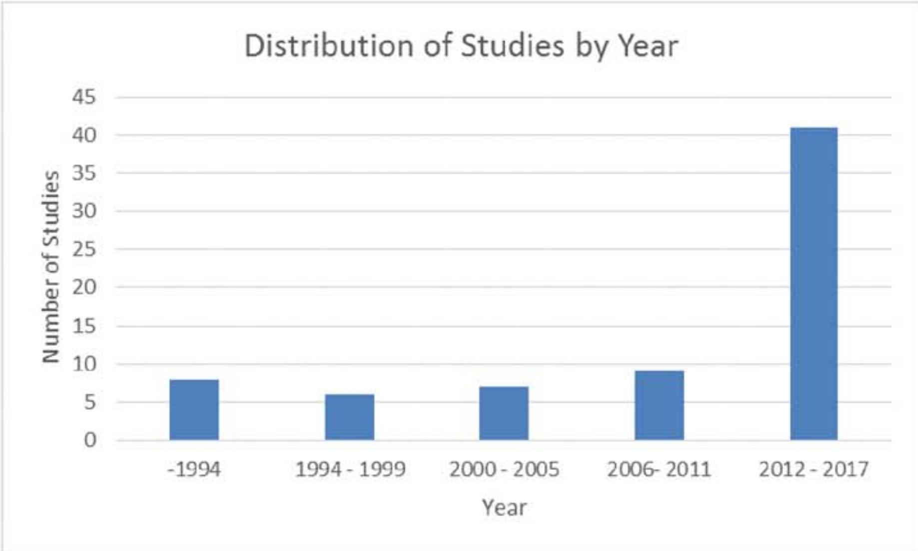

Figure 2: *Distribution of the studies by year.*

the increasing trend in the budget of VR projects were chosen. After that, the studies which list the disadvantages and missing points of this technology were added to this study. Therefore, we showed that although it is an important and popular technology, it still needs to be improved. In addition, the studies which declare why existing software development methodologies may not be applied to VR projects were included to our study.

- **Human Computer Interaction Area:** This category was formed to represent that VR is a kind of HCI topic with more immersive manner. Since we could not find any software development methodology for VR projects, we found serious games in the field of HCI, which is the nearest field of VR. Hence, the studies which give the similarities of VR and serious games were selected. In addition, the software development methodologies developed for games were described in detailed.

According to the results in the figure, the field of "*Virtual Reality*" with a rate of 34 percent is the area in which studies are mostly selected. Following this field, human computer interaction area, software development methodologies, and SDLC domains are the areas with the mostly used studies with rates of 17, 15, and 13 percent, respectively.

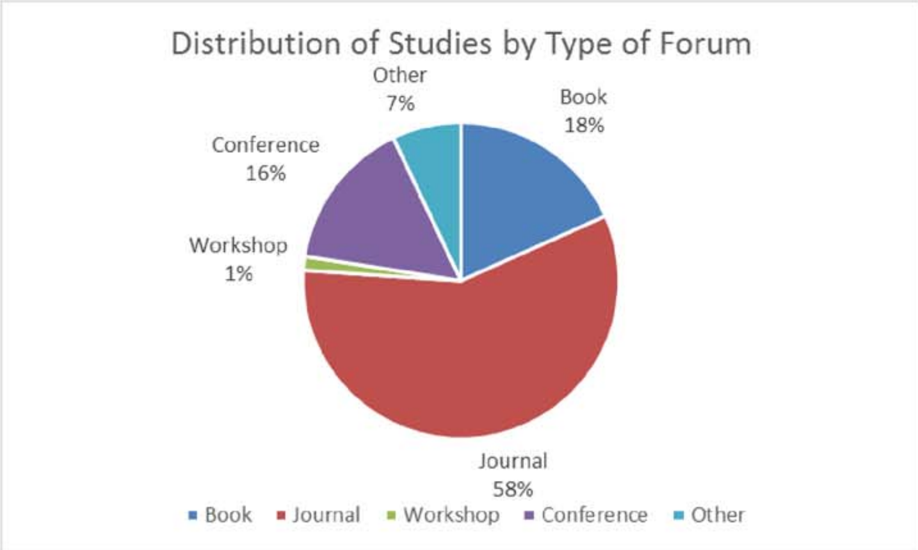

Figure 3: *Distribution of the studies by type of forum.*

4.2 Research Questions

The most important finding that arises as a result of this work is that there is no applicable software development methodology designed for VR projects. This situation has led us to find software development methodologies developed for managing projects in the HCI domain since VR is a sub-division of HCI. In order to find similar works, we attempted to find a workspace closest to VR. As a result of this search, serious games were found as the closest space to VR. It has been determined that there are 5 different methods [Chandler, 2009, Ramadan and Widyani, 2013, Studio, 2014, McGrath, 2014, Hendrick, 2014] designed for game projects as an answer to the fifth question (RQ.S5) from the research questions.

The second significant result of this literature search explains why existing software development processes cannot be applied to VR projects. It is our opinion that the reason for developing five different software development methodologies for game projects is when existing methods are applied to game projects, the results obtained from the project development fail. There are many studies [Klinger et al., 2010, Aleem et al., 2016, Jones, 2003, dos Santos Nunes et al., 2011] supporting this idea in the literature. According to the reasons stated in these studies, it is emphasized that existing software development processes are often used to manage projects involving mostly coding and organizing the engineers who describe themselves as coders. Therefore, as an answer to the fourth research question (RQ.S4), when these processes are applied to mul-

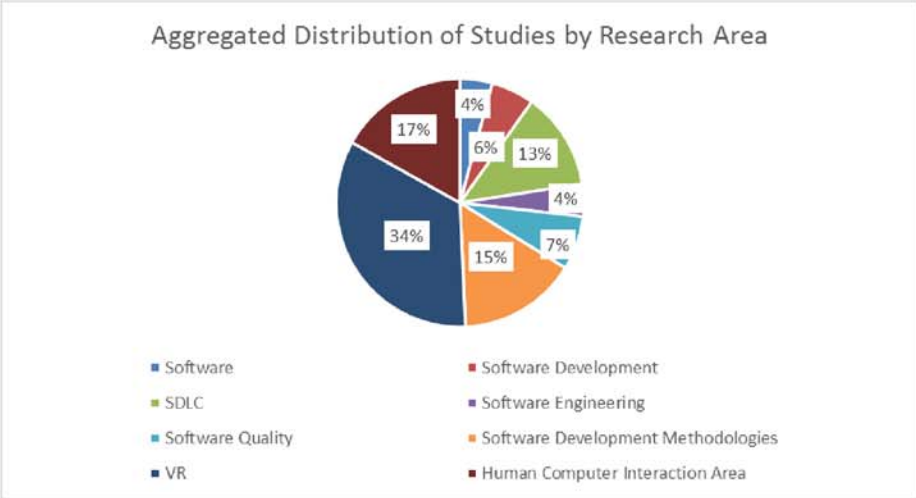

Figure 4: *Aggregated distribution of the studies by research area.*

tidisciplinary projects, including artificial intelligence, computer graphics, art and design, sounds, human factors, properties of software products and physical control systems, the requirements of these applications cannot be provided since they are compounds of several different areas and they differ from the requirements of standard software projects. In addition, the project teams of multidisciplinary projects are formed by engineers, designers, scenarists and experts in specific fields. Hence, when the existing software development processes, which are mostly used to organize engineers engaged in coding, are applied to organizing teams, which consist of experts in different fields, the results may be ineffective.

It is critical to determine why software development methodologies are required since this study endeavours to illustrate the importance of developing a new software development process for VR projects. There are large numbers of studies [Ben-Zahia and Jaluta, 2014, Khan et al., 2014, Kaur and Sengupta, 2013, Royce, 1970, Sommerville, 2010, Victor, 2003, Boehm, 1986, Pisacane, 2014, Schmittner et al., 2015, Munassar and Govardhan, 2010, Flora and Chande, 2014, Takeuchi and Nonaka, 1986, Anderson, 2010, Beck, 2000, Butt et al., 2014] in the literature that explain why software development processes are needed and how these processes work. In order to respond to the second research question (RQ.S2), these studies indicate that a software package should be developed within a methodology that is systematically designed to meet these software quality criteria.

The existing software development methodologies have been compared to each other in order to answer the third research question (RQ.S3). This comparison was made considering the studies [Leau et al., 2012, Saxena and Upadhyay, 2016, Drury-Grogan and Kennedy, 2013, Moniruzzaman and Hossain, 2013, Nerur et al., 2005, Dyba and Dingsoyr, 2009, Mishra and Dubey, 2013] conducted in the literature. This comparison was made in order to demonstrate the advantages and disadvantages of existing methods, and to explain that the results are successful when the most appropriate process is selected and applied to a project when based on the requirements. There are many studies [Ben-Zahia and Jaluta, 2014, Khan et al., 2014, Kaur and Sengupta, 2013, Han and Xie, 2012] in the literature that clarify that the methodology can differ based on the properties of software and the resources of companies. Therefore, companies should manage the SDLC phases according to their resources and the requirements of their customers in order to produce cost-effective, qualified and successful projects. Another reason for existing different software development processes is that none of them can be applicable to the every type of project since they cannot provide every requirement for all project types. Since the benefits provided by each methodology differ, companies need to make the correct choice while determining which methodology should be used in their projects in order to avoid loss.

Finally, the studies [McConnell, 1998, Braude and Bernstein, 2016, Ghezzi et al., 2002, Singpurwalla and Wilson, 2012, Nielsen, 2015, Ming-Chang, 2014, McCall et al., 1977, Boehm, 1978, Grady, 1992, Dromey, 1996] in the literature emphasize the importance of software engineering as an answer of the first research question (RQ.S1). They say that software development is a significant topic such that its phases should be thought about and implemented in terms of engineering concepts so as to produce measurable, evaluable, reusable and replicable products which have concrete outputs as well as to satisfy software quality criteria.

Table 5 summarizes the entire result part of the study by assigning the studies in the literature to the answers of the research questions.

4.3 Evaluation

In order to evaluate the quality of the study, 7 evaluation questions, which was listed in Section 3, were asked to 5 experts in software engineering and virtual reality. A five-point Likert-scale method has been used to evaluate experts' thoughts about the questions. The experts gave a value between the values as 0 for "*I do not agree*" and 5 for "*I strongly agree*" in order to indicate whether or not they agreed with the questions asked. Both the manuscript and the research questions were given to the experts before the evaluation stage and the experts were asked to read the articles by considering the research questions within 1 week. After reading the article, they submitted their answers according to the

Research Questions	The Studies in the Literature
RQ.S1	[R9], [R12], [R24], [R29], [R30], [R51], [R52], [R56], [R63], [R75]
RQ.S2	[R3], [R7], [R8], [R9], [R14], [R27], [R42], [R43], [R60], [R67], [R70], [R73], [R76], [R80], [R82]
RQ.S3	[R8], [R25], [R26], [R32], [R42], [R43], [R46], [R57], [R58], [R62], [R72]
RQ.S4	[R1], [R23], [R39], [R45]
RQ.S5	[R16], [R34], [R54], [R68], [R78]

Table 5: *Assignment of the studies as an answer of the research questions*

questions asked at the requested scale. Table 6 shows the scores given by each expert to the questions.

Expert No.	EQ1	EQ2	EQ3	EQ4	EQ5	EQ6	EQ7
Expert 1	5	4	5	5	4	4	4
Expert 2	5	5	5	5	5	5	5
Expert 3	4	4	5	4	5	5	4
Expert 4	3	3	4	4	4	4	3
Expert 5	4	4	5	4	5	4	5

Table 6: *Expert evaluation scores*

The answers given by the experts to the questions (Figure 5) were evaluated on a question-based analysis by taking the arithmetic mean of the answers given to the questions.

4.4 Threats to Validity

There are some main threats that may affect the validity of the study. These threats can be listed as:

1. **Missing of important studies:** In the search operation, the academic databases which are IEEExplore, Springer and ACM Digital Library were

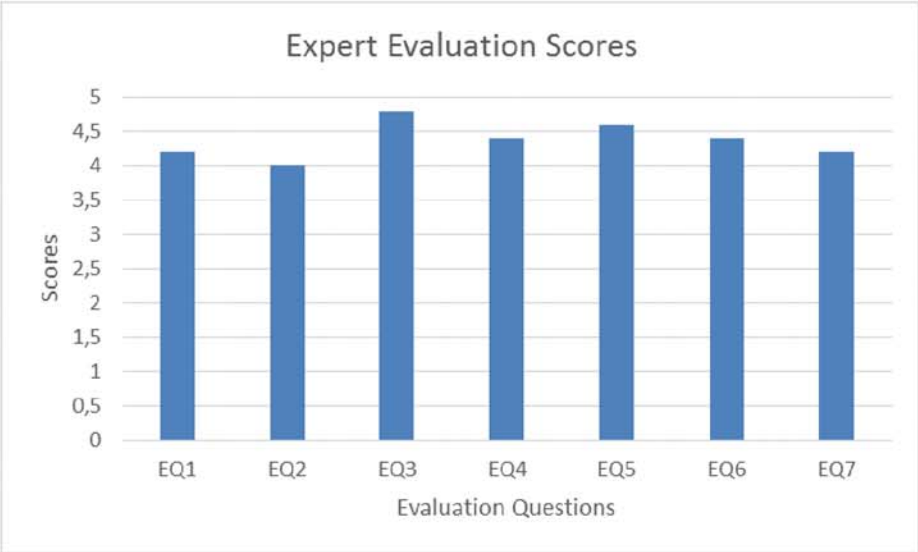

Figure 5: *The arithmetic average values of the answers given by the experts to the questions.*

selected to analyse the relevant studies in the literature. In support, some platforms which are Google Scholar and ResearchGate were used to increase the number of relevant studies in the literature. Although we have taken much time to find relevant studies in the literature and to study these studies, some important studies may have been omitted.

2. **Selection operation:** Research questions have been defined in order to determine the boundaries of the work and to better select the related studies. In addition, the search strings have been determined to select the studies. We believe that both the research questions and the search strings were selected to cover the relevant area. However, the number of search strings or the content of the research questions may be changed.

3. **Evaluation strategy:** The evaluation of the study was completed by administrating a questionnaire with 5 experts in software engineering and virtual reality. 7 evaluation questions were asked to the experts and the experts were asked to give a value between 0 and 5 to determine whether they agreed in each question. There may be 2 possible threats in this evaluation strategy. The first possible problem is the content of the evaluation question. We have created the contents of the evaluation questions in such a way as to be an answer to the research questions as well as to determine whether the research

words and databases used are appropriate. However, the content of the evaluation questions may be altered. The second one is that the evaluation of the study may be seen as subjective since the experts' opinions were evaluated. Although we believe that the results of the study are consistent with the experts' opinions, perhaps the results may be different if the analysis is done with other people, as people's thoughts and opinions about a subject may always differ.

5 Discussion

In this section, a discussion of the results obtained from the literature review including the opinions of the experts about the study is provided.

According to the studies in the literature, VR, one of the most popular workspaces of recent years, has many useful applications. In addition, these applications are becoming more important and the number of applications developed in this area is increasing rapidly. There are many companies that develop VR applications in many different areas including health, education, sports and psychology. However, although VR applications are so widely used, such applications still have some drawbacks that need to be improved. The most important drawback of VR applications is motion sickness problem. Especially in games, the applications have a lot of movement. However, when the companies try to simulate the movements in the virtual world, this causes motion sickness problem. In order to solve this problem, the player stands in a fixed place and the environment moves around the player, but, this situation reduces the sense of presence since the games become more static.

It has been observed from the literature that VR has many problems similar to this type of problem. Actually, these types of problems can be seen as the non-functional requirements of the VR applications. Since VR is a kind of software and development of a software is not an easy task, there is an immediate need for a new software development methodology for VR projects in order to fulfil these gaps. In support, the studies in the literature indicate that the software development methodologies manage the phases of SDLC in order to increase the software quality which is measured by considering the non-functional requirements of the systems. However, existing software development methodologies may not be sufficient to develop VR projects due to the reasons we have mentioned in the previous sections. The most important reason of this circumstance is that the requirements of VR projects are different from the requirements of standard software development projects since VR is an interdisciplinary area. In order to support this idea, we found many different software development methodologies developed for game projects, which is an interdisciplinary space such as VR. However, there is no software development methodology developed for VR projects in the literature. Hence, we believe that there is a need for a new software development process for VR projects because of the following main reasons:

– There are several problems that can arise when existing methodologies are applied to VR projects.

– There exist several different software development methodologies for other interdisciplinary fields.

– The existing quality of VR projects needs to be increased.

In order to support this idea, an evaluation was carried out with 5 experts in software engineering and virtual reality. During this evaluation process, the experts read the article and gave a score between 0 and 5 for each evaluation question (N = 7). According to the results obtained from this assessment, the experts, except the fourth expert, gave scores of 4 and 5 which means "*I agree*" and "*I strongly agree*" to all evaluation questions. While the fourth expert gave score of 4 to the third, fourth, fifth and sixth evaluation questions, he gave score of 3, which means "*neither agree nor disagree*", to the first, second and seventh evaluation questions. In addition, when an analysis based on the question is completed, it is seen (Figure 5) that the average score of all questions is over 4. This means that the overall opinion is that the experts support the idea of the study.

6 Conclusion and Future Work

The main objective of this study is to determine whether it is necessary to develop a new software development methodology for VR projects. In order to achieve this objective, first of all, brief information about VR was discussed followed by mentioning that the popularity of VR has increased considerably and the business fields in which it has been used. After that, software, software development and software development steps were explained so as to be able to create the infrastructure of the study. In addition, information on how software engineering emerged, the importance of software engineering, what the development methods were, and why many software development processes are needed. After revealing why existing processes cannot be applied to VR projects, it is stated that VR is a multidisciplinary area and a subcategory of HCI. Furthermore, it has been illustrated that serious games, another multidisciplinary field, have been found to be the closest space to VR. Therefore, software development methodologies developed for game projects were analysed in detail. In order to systematically perform these steps, the academic literature was screened within a specific logical framework so that research questions could be identified and answered. In conclusion, it is stated that it is useful to develop a new software development process for VR projects in order to reduces the risk of errors and yield more successful findings since several software development methodologies have been developed for game projects that are close to the VR space and many studies in the literature have indicated that existing processes can result in failure when applied to multidisciplinary fields.

For future work, a new software development methodology for VR projects will be designed. This process will be applied to VR projects in order to understand its efficiency. In addition, a virtual environment will be created to teach the steps of the newly developed methodology to team members of VR projects.

References

[Aleem et al., 2016] Aleem, S., Capretz, L. F., and Ahmed, F. (2016). Game development software engineering process life cycle: a systematic review. *Journal of Software Engineering Research and Development*, 4(1):6.

[Alonso et al., 1998] Alonso, F., Fuertes, J., Montes, C., and Navajo, R. (1998). A quality model: How to improve the object-oriented software process. In *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, volume 5, pages 4884–4889. IEEE.

[Anderson, 2010] Anderson, D. (2010). The principles of the kanban method. *David J. Anderson & Associates.*

[April and Abran, 2012] April, A. and Abran, A. (2012). *Software maintenance management: evaluation and continuous improvement*, volume 67. John Wiley & Sons.

[Asad et al., 2004] Asad, C. A., Ullah, M. I., and Rehman, M.-U. (2004). An approach for software reliability model selection. In *Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International*, pages 534–539. IEEE.

[Barnes, 2016] Barnes, S. (2016). Understanding virtual reality in marketing: Nature, implications and potential.

[Beck, 2000] Beck, K. (2000). *Extreme programming explained: embrace change.* addison-wesley professional.

[Ben-Zahia and Jaluta, 2014] Ben-Zahia, M. A. and Jaluta, I. (2014). Criteria for selecting software development models. In *Computer & Information Technology (GSCIT), 2014 Global Summit on*, pages 1–6. IEEE.

[Boehm, 1986] Boehm, B. (1986). A spiral model of software development and enhancement. *ACM SIGSOFT Software Engineering Notes*, 11(4):14–24.

[Boehm, 1978] Boehm, B. W. (1978). *Characteristics of software quality*, volume 1. North-Holland.

[Bouchlaghem and Liyanage, 1996] Bouchlaghem, N. and Liyanage, I. (1996). Virtual reality applications in the uk's construction industry. *CIB REPORT*, pages 89–94.

[Braude and Bernstein, 2016] Braude, E. J. and Bernstein, M. E. (2016). *Software engineering: modern approaches.* Waveland Press.

[Brunnett et al., 2015] Brunnett, G., Coquillart, S., van Liere, R., Welch, G., and Váša, L. (2015). *Virtual Realities: International Dagstuhl Seminar, Dagstuhl Castle, Germany, June 9-14, 2013, Revised Selected Papers*, volume 8844. Springer.

[Butt et al., 2014] Butt, S. M., Onn, A., Butt, M. M., and Tabassam, N. (2014). Towards a model-based framework for integrating usability evaluation techniques in agile software model. In *Recent Advances on Soft Computing and Data Mining*, pages 561–570. Springer.

[Castro et al., 2014] Castro, W. P., Sánchez, M. J. R., González, C. T. P., Bethencourt, J. M., de la Fuente Portero, J. A., and Marco, R. G. (2014). Cognitive-behavioral treatment and antidepressants combined with virtual reality exposure for patients with chronic agoraphobia. *International Journal of Clinical and Health Psychology*, 14(1):9–17.

[Chandler, 2009] Chandler, H. M. (2009). *The game production handbook.* Jones & Bartlett Publishers.

[Cheng et al., 2014] Cheng, L.-K., Chieng, M.-H., and Chieng, W.-H. (2014). Measuring virtual experience in a three-dimensional virtual reality interactive simulator environment: a structural equation modeling approach. *Virtual Reality*, 18(3):173–188.

[Chenglei et al., 2016] Chenglei, Y., Gai, W., Lu, W., Xiangxu, M., Mingda, D., Qin, P., Xiaowen, S., Liu, J., and Nianmei, Z. (2016). New pattern and method of virtual reality system based on mobile devices. US Patent App. 15/251,252.

[Chung and Gardner, 2012] Chung, J. and Gardner, H. J. (2012). Temporal presence variation in immersive computer games. *International Journal of Human-Computer Interaction*, 28(8):511–529.

[Dart et al., 1987] Dart, S. A., Ellison, R. J., Feiler, P. H., and Habermann, A. N. (1987). Software development environments. *Computer*, 20(11):18–28.

[Delaney, 2014] Delaney, B. (2014). *Sex, drugs and tessellation: The truth about virtual reality, as revealed in the pages of CyberEdge Journal.* CyberEdge Information Services.

[Donath et al., 2016] Donath, L., Rössler, R., and Faude, O. (2016). Effects of virtual reality training (exergaming) compared to alternative exercise training and passive control on standing balance and functional mobility in healthy community-dwelling seniors: a meta-analytical review. *Sports medicine*, 46(9):1293–1309.

[dos Santos Nunes et al., 2011] dos Santos Nunes, F. d. L., da Costa, R. M. E. M., dos Santos Machado, L., and de Moraes, R. M. (2011). Realidade virtual para saúde no brasil: conceitos, desafios e oportunidades. *Rev. Bras. Eng. Biom*, 27(4):243–258.

[Dromey, 1996] Dromey, R. G. (1996). Cornering the chimera. *IEEE Software*, 13(1):33.

[Drury-Grogan and Kennedy, 2013] Drury-Grogan, M. L. and Kennedy, D. M. (2013). Highlighting communication activities and inefficiencies between agile vs. waterfall methods: An agent based model of knowledge sharing. In *8th Pre-ICIS International Research Workshop on Information Technology Project Management (IRWITPM 2013)*, page 46.

[Dyba and Dingsoyr, 2009] Dyba, T. and Dingsoyr, T. (2009). What do we know about agile software development? *IEEE software*, 26(5):6–9.

[Flora and Chande, 2014] Flora, H. K. and Chande, S. V. (2014). A systematic study on agile software development methodologies and practices. *International Journal of Computer Science and Information Technologies*, 5(3):3626–3637.

[Fu et al., 2015] Fu, Y., Yan, M., Zhang, X., Xu, L., Yang, D., and Kymer, J. D. (2015). Automated classification of software change messages by semi-supervised latent dirichlet allocation. *Information and Software Technology*, 57:369–377.

[Ghezzi et al., 2002] Ghezzi, C., Jazayeri, M., and Mandrioli, D. (2002). *Fundamentals of software engineering.* Prentice Hall PTR.

[Grady, 1992] Grady, R. B. (1992). *Practical software metrics for project management and process improvement.* Prentice-Hall, Inc.

[Gregory, 2017] Gregory, J. (2017). *Virtual reality.* Cherry Lake.

[Han and Xie, 2012] Han, B. and Xie, J. (2012). Practical experience: Adopt agile methodology combined with kanban for virtual reality development.

[Heldal, 2007] Heldal, I. (2007). Supporting participation in planning new roads by using virtual reality systems. *Virtual Reality*, 11(2-3):145–159.

[Hendrick, 2014] Hendrick, A. (2014). Project management for game development. june 15.

[Humphrey, 1989] Humphrey, W. S. (1989). The software engineering process: definition and scope. *ACM SIGSOFT Software Engineering Notes*, 14(4):82–83.

[Jayaram et al., 1997] Jayaram, S., Connacher, H. I., and Lyons, K. W. (1997). Virtual assembly using virtual reality techniques. *Computer-aided design*, 29(8):575–584.

[Jimeno and Puerta, 2007] Jimeno, A. and Puerta, A. (2007). State of the art of the virtual reality applied to design and manufacturing processes. *The International Journal of Advanced Manufacturing Technology*, 33(9-10):866–874.

[Johnson-Laird, 1992] Johnson-Laird, A. (1992). Reverse engineering of software: Separating legal mythology from actual technology. *Software Lj*, 5:331.

[Jones, 2003] Jones, C. (2003). Variations in software development practices. *IEEE software*, 20(6):22–27.

[Kannan et al., 2015] Kannan, K. et al. (2015). An approach for decomposing requirements into analysis pattern using problem frames (drap-pf). In *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, pages 2392–2396. IEEE.

[Karimi, 1983] Karimi, J. (1983). Computer aided process organization in software design.

[Kaur and Sengupta, 2013] Kaur, R. and Sengupta, J. (2013). Software process models and analysis on failure of software development projects. *arXiv preprint arXiv:1306.1068*.

[Khan et al., 2014] Khan, M. A., Parveen, A., and Sadiq, M. (2014). A method for the selection of software development life cycle models using analytic hierarchy process. In *Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014 International Conference on*, pages 534–540. IEEE.

[Kitchenham et al., 2011] Kitchenham, B. A., Budgen, D., and Brereton, O. P. (2011). Using mapping studies as the basis for further research–a participant-observer case study. *Information and Software Technology*, 53(6):638–651.

[Klinger et al., 2010] Klinger, E., Weiss, P. L., and Joseph, P.-A. (2010). Virtual reality for learning and rehabilitation. In *Rethinking physical and rehabilitation medicine*, pages 203–221. Springer.

[Leau et al., 2012] Leau, Y. B., Loo, W. K., Tham, W. Y., and Tan, S. F. (2012). Software development life cycle agile vs traditional approaches. In *International Conference on Information and Network Technology*, volume 37, pages 162–167.

[Lorenzo et al., 2016] Lorenzo, G., Lledó, A., Pomares, J., and Roig, R. (2016). Design and application of an immersive virtual reality system to enhance emotional skills for children with autism spectrum disorders. *Computers & Education*, 98:192–205.

[Lun and Chi, 2013] Lun, L. and Chi, X. (2013). On the relation of software architecture testing criteria in the c2 style. In *Conference Anthology, IEEE*, pages 1–5. IEEE.

[Maddox et al., 2017] Maddox, M. M., Feibus, A., Liu, J., Wang, J., Thomas, R., and Silberstein, J. L. (2017). 3d-printed soft-tissue physical models of renal malignancies for individualized surgical simulation: a feasibility study. *Journal of robotic surgery*, pages 1–7.

[Marsh, 2011] Marsh, T. (2011). Serious games continuum: Between games for purpose and experiential environments for purpose. *Entertainment Computing*, 2(2):61–68.

[McCall et al., 1977] McCall, J. A., Richards, P. K., and Walters, G. F. (1977). Factors in software quality. volume i. concepts and definitions of software quality. Technical report, DTIC Document.

[McConnell, 1998] McConnell, S. (1998). The art, science, and engineering of software development. *IEEE Software*, 15(1):120–118.

[McConnell, 2001] McConnell, S. (2001). Who needs software engineering? *IEEE Software*, 18(1):5–8.

[McGrath, 2014] McGrath, J. (2014). The game development lifecycle: A theory for the extension of the agile project methodology.

[Merchant et al., 2014] Merchant, Z., Goetz, E. T., Cifuentes, L., Keeney-Kennicutt, W., and Davis, T. J. (2014). Effectiveness of virtual reality-based instruction on students' learning outcomes in k-12 and higher education: A meta-analysis. *Computers & Education*, 70:29–40.

[Ming-Chang, 2014] Ming-Chang, L. (2014). Software quality factors and software quality metrics to enhance software quality assurance. *British Journal of Applied Science & Technology*, 4(21):3069.

[Mishra and Dubey, 2013] Mishra, A. and Dubey, D. (2013). A comparative study of different software development life cycle models in different scenarios. *International Journal*, 1(5):64–69.

[Moniruzzaman and Hossain, 2013] Moniruzzaman, A. and Hossain, D. S. A. (2013). Comparative study on agile software development methodologies. *arXiv preprint arXiv:1307.3356*.

[Mujber et al., 2004] Mujber, T. S., Szecsi, T., and Hashmi, M. S. (2004). Virtual reality applications in manufacturing process simulation. *Journal of materials processing technology*, 155:1834–1838.

[Munassar and Govardhan, 2010] Munassar, N. M. A. and Govardhan, A. (2010). A comparison between five models of software engineering. *IJCSI*, 5:95–101.

[Nazir et al., 2013] Nazir, S., Colombo, S., Manca, D., et al. (2013). Virtual reality as effective tool for training and decision-making: preliminary results of experiments performed with a plant simulator. In *European HSE Conference and Exhibition*. Society of Petroleum Engineers.

[Nerur et al., 2005] Nerur, S., Mahapatra, R., and Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM*, 48(5):72–78.

[Nielsen, 2015] Nielsen, P. D. (2015). Software engineering and the persistent pursuit of software quality. *CrossTalk*, page 5.

[Ninaus et al., 2014] Ninaus, M., Kober, S. E., Friedrich, E. V., Dunwell, I., De Freitas, S., Arnab, S., Ott, M., Kravcik, M., Lim, T., Louchart, S., et al. (2014). Neurophysiological methods for monitoring brain activity in serious games and virtual environments: a review. *International Journal of Technology Enhanced Learning*, 6(1):78–103.

[Pantano, 2014] Pantano, E. (2014). Innovation drivers in retail industry. *International Journal of Information Management*, 34(3):344–350.

[Petersen et al., 2015] Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18.

[Pisacane, 2014] Pisacane, V. L. (2014). Space systems engineering. In *The International Handbook of Space Technology*, pages 143–163. Springer.

[Ramadan and Widyani, 2013] Ramadan, R. and Widyani, Y. (2013). Game development life cycle guidelines. In *Advanced Computer Science and Information Systems (ICACSIS), 2013 International Conference on*, pages 95–100. IEEE.

[Ritterfeld et al., 2009] Ritterfeld, U., Cody, M., and Vorderer, P. (2009). *Serious games: Mechanisms and effects*. Routledge.

[Royce, 1970] Royce, W. W. (1970). Managing the development of large software systems. In *proceedings of IEEE WESCON*, volume 26, pages 328–338. Los Angeles.

[Sampaio and Martins, 2014] Sampaio, A. Z. and Martins, O. P. (2014). The application of virtual reality technology in the construction of bridge: The cantilever and incremental launching methods. *Automation in construction*, 37:58–67.

[Saxena and Upadhyay, 2016] Saxena, A. and Upadhyay, P. (2016). Waterfall vs. prototype: Comparative study of sdlc. *Imperial Journal of Interdisciplinary Research*, 2(6).

[Schmittner et al., 2015] Schmittner, C., Ma, Z., and Schoitsch, E. (2015). Combined safety and security development lifecylce. In *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, pages 1408–1415. IEEE.

[Schroeder, 1996] Schroeder, R. (1996). *Possible worlds: the social dynamic of virtual reality technology*. Westview Press, Inc.

[Singpurwalla and Wilson, 2012] Singpurwalla, N. D. and Wilson, S. P. (2012). *Statistical methods in software engineering: reliability and risk*. Springer Science & Business Media.

[Sommerville, 2010] Sommerville, I. (2010). *Software Engineering*. Addison-Wesley.

[Stone, 2009] Stone, R. (2009). Serious games: virtual realitys second coming? *Virtual reality*, 13(1):1–2.

[Studio, 2014] Studio, B. G. (2014). Project lifecycle.

[Subhan and Bhatti, 2015] Subhan, Z. and Bhatti, A. T. (2015). Requirements analysis and design in the context of various software development approaches. *International Journal for Research in Applied Science & Engineering*, 3:812–820.

[Takeuchi and Nonaka, 1986] Takeuchi, H. and Nonaka, I. (1986). The new new product development game. *Harvard business review*, 64(1):137–146.

[Tanenbaum et al., 2013] Tanenbaum, J. G., Antle, A. N., and Robinson, J. (2013). Three perspectives on behavior change for serious games. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3389–3392. ACM.

[Victor, 2003] Victor, R. (2003). Iterative and incremental development: A brief history. *IEEE Computer Society*, pages 47–56.

[Virvou et al., 2005] Virvou, M., Katsionis, G., and Manos, K. (2005). Combining software games with education: evaluation of its educational effectiveness. *Educational Technology & Society*, 8(2):54–65.

[Weinberg, 1971] Weinberg, G. M. (1971). *The psychology of computer programming*, volume 932633420. Van Nostrand Reinhold New York.

[Yiannakopoulou et al., 2015] Yiannakopoulou, E., Nikiteas, N., Perrea, D., and Tsigris, C. (2015). Virtual reality simulators and training in laparoscopic surgery. *International Journal of Surgery*, 13:60–64.