RISKS INFLUENCING SOFTWARE PROJECTS AND MANAGING THEM DURING THE REQUIREMENT ENGINEERING PROCESS

Kaltrina Ukalli
”Ss. Cyril and Methodius” University in Skopje, Macedonia, kaltrinaukalli@gmail.com
Ljupcho Antovski
”Ss. Cyril and Methodius” University in Skopje, Macedonia, ljupcho.antovski@finki.ukim.mk

Abstract: Software development has become one of the most important industries in the past decade, as a consequence of the inseparable connection to the modern technologies. Relying on the growth of new technologies, this industry is facing lots of innovations and a lot more challenges. The key to fit into these innovations is flexibility and adaptability to changes. These changes come from the evolution of technology and software applications and have direct impact on the productivity of the software companies. The main aim of the software companies is to make useful products to fulfill people needs and desires, and in meantime to benefit from them in many ways. Their profitability is a composition of success, enormous incomes and fame. In order to achieve their goals they have to invest in good project plans and project management. Software companies also tend to be on top, by carefully analyzing and reviewing the market needs same as the costumer and end-user needs. This is a long and crucial process, known as requirement engineering, where the services that the system ought to do are determined, as well as the constraints under the system will operate. Descriptions of these services are represented by the term requirements. The most important thing in the beginning of the product development is that the requirements should be clearly specified, in order to skip uncertainty, ambiguity and vagueness, because in case there is any doubt, the product might fail to fulfill its goals. Since the products that need to be developed are relatively new and complex, they are followed by possibilities of harm called risks, which while neglected, might lead them to problems or even project failure. Dealing with these risks is the primary job of the specific field of project management called risk management. These risks might have negative effect, but there are also some risks that have positive effect too, known as opportunities. However, if some of risks are identified in the initial stages of the project, such as during the requirement engineering, and are well managed, it will save the company of lots of troubles, including time and budget loss. This paper gives an overview about requirement engineering as the fundamental phase in project management and the importance of risk management. Furthermore, here it is discussed the significant relationship between these two, which has a great influence in the project success. Having in mind that unwanted situations can occur anytime during the product development, forecasting and managing risks from the beginning is a good start. This paper is dealing specifically with some traps that might occur during the requirement engineering process and are given options how to deal with them. The role of risk management is briefly explained, including the risk management process and its phases. Some strategies or methods to deal with risks are discussed as well. By good risk management is meant the proper usage of strategies, whether they are positive or negative risks. Some of the strategies to deal with risks in general are considered, especially in the requirement engineering process. The focus of this work is to give an idea of corresponding strategies to specific risks, in order to deal with them in a way that the product can benefit or to avoid possible damages or in the worst case product failure.

Keywords: Requirements, Requirement Engineering, Risk, Risk Management.

1. INTRODUCTION

The software industry’s aim is to continuously produce software products, whether they are new or updated versions of the existing ones. Even though developing software projects seems to be in high rates and big numbers, there are also a lot of projects that fail. According to Dedolph [1], these failures come as a result of neglecting risks or bad risk management. There are lots of risks that are threatening the software development cycle, but as to Singh [2], the major risks are related to the three relevant constraints: time, money and quality. Overrunning schedules leads to increase in cost, and trying to fit into the budget might probably decrease the quality [3]. Concerning that, the Standish Group[193] has been doing research about the software project performance for more than 30 years, and has

[193] **The Standish Group** is an independent international IT research advisory firm founded in 1985.

been publishing CHAOS Reports[194] about project success rates since 1994 [4]. The Fig.1 below represents a graph of the software project performance from 1994 till 2015 [5] [6]. As it can be seen in the figure below for over 20 years, the successful projects appear on low percentages, compared to the ones that failed or were challenged in one of the three aforementioned constraints.

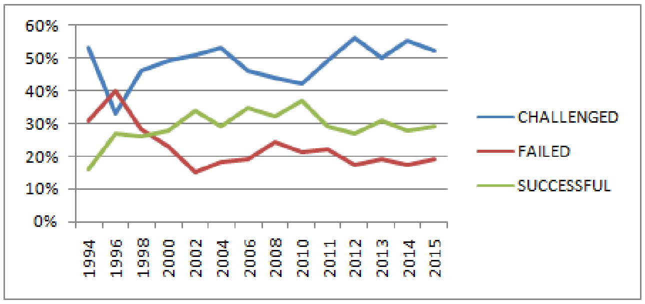


Fig. 1: Standish CHAOS Summary Reports (1994 – 2015)

Risks have a great influence in the destiny of the software project. Failing to manage them, or letting them take actions, will definitely not bring the desired outcomes. According to Singh [2], these risks can be avoided if proper requirement engineering is done. Poor requirements will lead the project toward significant risks [7]. So it seems like there is a connection between requirement engineering and risk management, which determines the progress of the project. In order to achieve software development success it is important to understand well the project requirements, otherwise the project is going to face a lot of problems, moreover it is going to fail [8].

The aim of this work is to conduct a literature review and bring back relevant information, in order to give a good overview of the relationship between risks and requirements. As requirements play a big role on project success as a foundation of the project, it is important those to be clear and complete, avoiding traps and possible events that will harm the work. What is important in this paper is to see, how risks can be managed in the initial phases like requirement engineering, by avoiding traps that might look simple but have great impact in the following phases and making disasters. This paper is organized as follows: the first section defines the aim of this paper, the second section is dedicated to the requirement engineering process and its main activities, the third section gives a brief overview of risk management and its stages, while the fourth section stands for the risk management during the requirement engineering process, the fifth section describes traps that might occur and their management in order to avoid risks, the sixth section gives an overview of numbers and reasons for project failures, followed by the seventh section which gives a sort of a model to manage risks during requirement engineering and last section which is the conclusion.

2. REQUIREMENT ENGINEERING

Requirements are descriptions of the services that a software system would provide, and represent the common term for all customer needs that have to do with that system functionality [9] [10]. As so, they need to be clear and understood well, so that the costumers end up being satisfied with the product. There is a broad set of tasks and techniques that help to understand the requirements and it is called requirement engineering (RE) [11]. This activity starts from the discussion about requirements till their modeling and validation. Depending on many factors, defining requirements may take a considerable time. As a result, projects that ensure complete and consistent requirements triple the chances for project success [7].

According to Young [12], requirements and requirement engineering are very important because they represent the basis for the following technical work: system design, development, testing, implementation and operation. However, project managers often disregard the requirement engineering. They believe that starting to work will help measure the progress, since requirements are not clearly defined, but according to industry experience, the best time investment during a project development is the requirements engineering [12].

2.1. REQUIREMENT ENGINEERING MAIN ACTIVITIES

As stated in [11], requirement engineering consists of seven important tasks:

[194] **CHAOS Report** is an annual report of the state of software development, published by The Standish Group. They monitor the % of projects that are: **Successful** (completed On Time and On Budget with required Features and Functions), **Challenged** (Late with Budget Overruns) and **Failed** (Cancelled before Completion or Delivered and Never Used)

- **Inception**. This is the activity where a basic understanding of the problem and people to be involved is settled [11].
- **Elicitation**. This is a very complex process of seeking and acquiring requirements [13]. This is the activity where resources are identified and information from all the stakeholders is gathered [14]. In this phase the stakeholders are identified, as well as, the goals to be met along with the constraints and the tasks to be performed [15]. The elicitation activity is considered critical to the project development, and is done by really experienced analysts, because if this process fails, the project will not meet the customer requirements and the project fails too [16].
- **Elaboration/Analysis**. This is the activity where all the information gathered from the previous two activities is expanded, and is characterized by the creation of user scenarios about how the system will interact with end users [11].
- **Negotiation**. This represents the activity where all the stakeholders deal with conflicts about ranking prioritizations of requirements and as well as resolving those conflicts by making a compromise and agreements [9].
- **Specification** is the activity where the requirements are documented [9].
- **Validation** is the process where the requirements are checked and revised to ensure that they are not ambiguous or inconsistent and they truly define the project that the customer wants [11] [9].
- **Management**. Identifying, controlling and tracking the requirements and changes made to them at anytime are the job of the requirement management [11]. This activity also involves evaluation of risks and their impact [15].

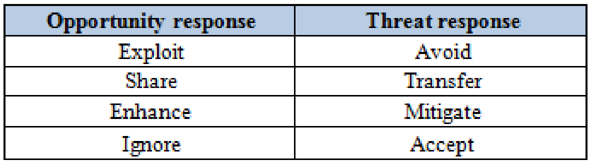
3. RISK MANAGEMENT

A risk is a potential harm or an event that may or may not occur, followed by different consequences on the project objectives [1]. What almost all definitions of risk claim is uncertainty and loss, but none is generally accepted [17]. Based on the impact that risks have on project objectives, besides threats (negative risks), they include opportunities (positive risks) as well [18]. Disregarding the traditional view of risks which is only negative, as to Hillson [19], the positive risks represent uncertainties that have positive and beneficial effect in reaching the project goals.

Managing risks takes time and a set of activities which are known as risk management. According to Wallmüller [17], risk management can be defined as a process for identifying, analyzing and controlling risks in projects. This important process is systematic that continues throughout the project, and when it's done properly it will estimate project chances to succeed, otherwise problems will arise. The risk management process consists of several activities whose goals among others are to identify, analyze and monitor risks. There are some known stages of the risk management process [9], which are listed below, as well as shown in the Fig.2.

- **Risk identification.** This is the first stage of the risk management, and the basis for the following stages. Identifying risks should be the responsibility of all the stakeholders rather than just the project manager and it should be done throughout all the project phases, especially in the requirement engineering [20]. Creating a risk checklist of possible threatening and predictable risks is one of the methods for identifying risks [11]. As to Sommerville [9], there are some types of risks that can be included in the checklist like: Technology risks; People risks; Organizational risks; Tools risks; Requirement risks; Estimation risks.
- **Risk Analysis.** Risk analysis is the second stage, where the identified risks are transformed into decision-making information [2]. This means that each identified risk is analyzed and based on experience with previous risks judgments are made about actual risks [9]. As to Passenheim [20], the so-called scenario analysis is the commonly used technique to do risk analysis, which consists of the probability of the occurring events and the impact they would have on the project. There are some models to do risk analysis during the requirement engineering process, such as:
  a) **Risk Tree Assessment**. Risk tree is composed of many events, categorized into three levels: Lowest Level (primary events), Middle Level (intermediate events) and High Level (top events). All these events are connected by gates into a tree, showing the relationship of events from the bottom to the top [21].
  b) **Goal- driven software development risk management modeling (GSRM)**. This framework's job is to assess reason, control and trace software development risk [21]. This model consists of four layers [22]: Goal layer; Obstacle layer; Assessment layer and Treatment layer or Mitigation layer. According to [21], the main purpose of this model is to control risks from an early stage by integrating risk management activities within the requirement engineering process.

- **Risk planning or risk response.** This stage considers all the identified and analyzed risks, so that it can start planning the strategies how to respond to those risks. There are some alternatives of responses defined by literature for both threats (negative risks) and opportunities (positive risks) [23].



Opportunity response	Threat response
Exploit	Avoid
Share	Transfer
Enhance	Mitigate
Ignore	Accept

Table 1: Opportunity Responses vs. Threat Responses

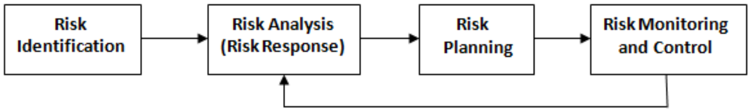


Fig. 2: The risk management process

- **Risk Monitoring and Control.** The last stage is the monitoring and control stage, which is also of a big significance since it executes the risk response strategies to check whether the effects of risks have changed [20]. The monitoring and control of risks should be done regularly along with discussions about risks, analyses about their probability to arise and its consequences (See Figure) [9].

4. RISK MANAGEMENT DURING THE REQUIREMENT ENGINEERING PROCESS

Developing the right software solution that meets user's expectations is really hard [10]. Conducting the appropriate requirement engineering and sticking onto the defined requirements is the key to successful software solutions. However risks are inevitable part, especially during the requirement engineering process. Among the most important classifications of risks in general, requirement engineering considers the category of Software Requirement Risks which come as a result of [24]: Lack of analysis for change of requirements; Lack of report for requirements; Ambiguity of requirements; Inadequate of requirements; Invalid requirements; Change extension of requirements; Poor definition of requirements; Change of requirements; Impossible requirements.

These risks can be managed by appropriate risk management techniques such as [25]:

- **Requirements scrubbing**. This technique represents the process of removing unnecessary requirements, or just simplifying the complex ones, after examining the product specification [26].
- **Prototyping**. Prototypes are models that look or behave like the system that is going to be built. By taking many forms, from paper illustrations to online working models, they can be used to verify the validity of the requirements [27].
- **Cost-benefit analysis**. As its name claims, this technique is evaluating and comparing the costs and benefits of a specific project, in order to decide whether it is worth starting it [28].
- **Design to cost**. This technique estimates in advance all the costs in order to systematically constrain the design goals according to the available budget [29].
- **User surveys**. This technique helps in better understanding the user needs, in order to minimize the risk of their discontent with the system.

5. MANAGING TRAPS DURING REQUIREMENT ENGINEERING IN ORDER TO AVOID RISKS

Among many project failure reasons, poor requirement engineering is one of the most important ones [30]. There is a special connection between requirement engineering and the project success, because requirements bridge the needs or ideas, to the project outcomes [31], and play a critical role in its success or failure. According to IBM [32], among others, there are some mistakes or traps that can be avoided, if the requirements are understood right. These traps are listed and briefly explained below.

a) **Scope creep.** This is the case when the requirements are not well defined and the scope of the project uncontrollably grows, leading to resources overrun. This is as a consequence of stakeholders trying to make changes in the requirements, or new requirements are continually added during development [33]. As to Suresh [34], there are two classifications of scope creep:

- **Business Scope Creep.** Because of the changes in the market, requirements already defined might change. This happens due to insufficient requirement analysis, failing to meet user expectations or not involving users in the initial phases [34].

- **Technology Scope Creep.** Uncontrolled growth can also happen when the company is trying to please the customer all the time in changing requirements or when the when the developers add things that have not been defined in the requirement definition [34].

In order to avoid scope creep, business requirements must be very detailed and specifically described [32].

b) **Asking customers what they want and avoiding ambiguous requirements.** Fact is that, usually customers don't know how to explain what they want. As a result, developers don't understand the problem and end up with defining vogue and ambiguous requirements, which are result of unclear information [32]. The project team then spends lots to time trying to make them clearer because these requirements will influence the next phase, and will damage the project especially the fixed-cost projects [32]. This can be avoided with in depth conversations with the customers about what they want the system to do. Stakeholders should also have a lot of meetings and write down everything with text and visuals, that everyone can understand.

c) **Inability to adapt to change.** One thing is for sure, if the project hasn't defined a process to deal with requirement changes, it is following a roadmap that leads to waste of time and money [32]. The salvation here is planning in advance what to do if changes happen and keeping stakeholders informed about everything.

d) **Failure to communicate effectively and frequently.** The reason why most project fail is lack of communication between stakeholders, especially with the customers. Requirements should be expressed in a manner that all the stakeholders understand, and not guessing what the customer wants but communicate with them, in order to avoid ambiguity [32]. There should be a lot of meetings and communication with all the stakeholders, so that things become clearer and the developed product satisfies customer's needs.

e) **Analysis Paralysis.** This is the case when requirement development doesn't seem to stop, and there is a tendency to create "the perfect" requirements before beginning the construction [33]. This trap keeps stakeholders paralyzed instead of building complete and clearly expressed requirements, and making clearer those who are not.

f) **Failure to measure and assess requirements processes.** Of a very big significance is the assessment of the requirement process, considering the complexity of requirement management, in order to reduce chances of failure and improving the process as well [32].

g) **Isolating your requirements.** Another factor for risk increasing is not being able to capture relationships between requirements, and changing one of them might affect others and create a disaster [32]. To avoid this, a clear picture of all the requirements should always be in mind, in order to know how they are related and how they affect each other.

6. REASONS FOR FAILURE IN REQUIREMENT ENGINEERING

There are many reasons why for software project failure, but high percentage of these failures comes as a result of very poor requirement engineering [35], as discussed in section 5. Disregarding the process of requirement engineering results into project suffering in terms of time and budget overrun, furthermore to project failure, which affects the image of the company in the competitive market [36].

As to [21], the probability of risk occurrence is important to consider, whereas Low Level Risks have higher percentage (38%) than the Medium (31%) or High Level ones (24%). Most of the High Level Risks (major affecting risks) have backgrounds in the requirement engineering phase, so by proper requirement engineering the errors can be detected and corrected with lower cost, otherwise in the later phases the project can have problems and be canceled, due to time and money overrun [21]. This is confirmed by the Standish Group CHAOS reports for more than 20 years. As it can be seen in the graph in Fig.1 for the time interval of 20 years we have high percentages of Challenged projects (Late with budget overruns), more specifically in 2015, 52% of the projects result Challenged, 29% result Successful and 19% are Failed [37]. What also important concerning the CHAOS Reports (see Fig.1) is that the percentage of Challenged projects in 1994 is almost the same as in 2015. According to the CHAOS Report 1994, among the factors that cause challenges or failure, the biggest percentages belong to incomplete requirements or changing requirements [38]. Also this report claims that in 100 starting projects, 94 projects will have restarts (more than one), which leads to time and budget overrun. Considering Wiegers [39], another important factor for upcoming problems is not prioritizing requirements, and treating them as they have the same importance. Another relevant reason for failure as to [40], is the poor management of changes in requirements. There are a lot more statistics, but these are the most relevant ones.

7. EFFECTIVE RISK MANAGEMENT IN REQUIREMENT ENGINEERING

Being aware of the risks that can harm the project and managing them in the early stages during the requirement engineering can save the project from a lot of troubles. In order to have more effective risk management in requirement engineering, some things must be considered:

- Throughout the requirement activities of inception and elicitation, the technique of user surveys should be conducted, in order to gather more information and clarify more the needs of the users;
- During the process of analysis of the requirements, lots of user scenarios must be done, using the UML[195] diagrams, in order to better understand the interaction between the system and end users. Using the prototyping technique as well, is a good way to be more precise about the system's functionalities and see whether they meet user's expectations. However if changes are made to requirements or added new ones, the technique of requirement scrubbing is needed to minimize scope creep.
- Having frequent meetings with all the stakeholders is essential in the negotiation activity, bur also necessary throughout the whole requirement engineering process.
- What is really important before the requirement specification is conducted is using the cost-benefit analysis technique, so that the right requirements are properly documented.
- User surveys and meetings are a good solution for requirement validation, while for further risk analyses and risk management from an early stage, using the GSRM framework is the key.

8. CONCLUSION

Being the core elements of a project to be developed, the requirements need a special treatment and lots of analysis in order to satisfy the project goals and user needs. This makes the requirement engineering process initial as well as the most important part of the whole process. However statistics show that this part is often responsible for the upcoming risks. Managing risks from the beginning is crucial and deterministic for the future of the project. No matter the size of the project, requirement engineering and risk management should be its inevitable parts. This paper presented an overview of them, their relationship and role in tracking mistakes that are made during the requirement definition. Requirement engineering activities are briefly described as a means to ensure completeness and consistency of requirements, as these requirements determine the functionality of the system being developed. In case that they are unclear and ambiguous, they are followed by developer's conjecture and result in undesirable outcomes, or product failure. Risk management is described as a set of activities that undertake the appropriate actions to avoid or minimize risks, thus continues to improve project evolution using some risk management techniques during the requirement engineering process. Some traps are also discussed, that can be faced during requirement engineering, which in case of responding them with indifference might affect the following phases of the development, as well as influencing the overall project destiny. Considering the fact that the majority of the high level risks lie in the requirement engineering process some statistics of project failures are discussed, including the reasons for their failure. Last but not least is given a roadmap or a model how to avoid or deal with risks during the initial phase of requirement engineering.

REFERENCES

[1] Michael F. Dedolph, "The neglected management activity: Software risk management," *Bell Labs Technical Journal*, vol. 8, no. 3, pp. 91-95, September 2003.

[2] Rajinder Singh, "Association between Requirement Engineering Processes and Risk Management," *International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)*, vol. 8, no. 3, pp. 223-229, March-May 2014.

[3] Shipra Gupta and Manoj Wadhwa, "Requirement engineering: An overview," *International Journal of Research in Engineering and Technology*, vol. 1, no. 2, pp. 155-160, July 2013.

[4] The Standish Group International, Inc. (2018) The Standish Group. [Online]. "https://www.standishgroup.com/about" https://www.standishgroup.com/about

[5] Márcio d'Ávila. (2011, June) Blog do Márcio d'Ávila - Tecnologia da informação, notas e cotidiano. [Online]. HYPERLINK "http://blog.mhavila.com.br/2011/06/18/sucesso-de-projetos-atualizado/" http://blog.mhavila.com.br/2011/06/18/sucesso-de-projetos-atualizado/

[195] UML – Unified Modeling Language

[6] Steven R. Meier, "Technology Portfolio Management for Project Managers," SRM Consulting LLC, Presentation 28 February 2017.

[7] Louis Wheatcraft, "Triple Your Chances of Project Success Risk and Requirements," *INCOSE International Symposium*, vol. 21, no. 1, pp. 543-558, June 2011.

[8] Paul Ralph, "The illusion of requirements in software development," *Requirements Engineering*, vol. 18, no. 3, pp. 293-296, September 2013.

[9] Ian Sommerville, *Software Engineering*, 9th ed., Marcia Horton, Ed. Boston, United States of America: Pearson Education, Inc, 2011.

[10] Marija Boban, Zeljka Pozgaj, and Hrvoje Sertic, "Strategies for successful software development risk management," *Management*, vol. 8, no. 2, pp. 71 - 99, January 2003.

[11] Roger S. Pressman and Bruce R. Maxim, *Software Engineering; A practitioner's Approach*, 8th ed. New York, United States of America: McGraw-Hill Education, 2015.

[12] Ralph Rowland Young, *The requirements engineering handbook*.: Artech House Inc., 2004.

[13] Didar Zowghi and Chad Coulin, "Requirements elicitation: A survey of techniques, approaches, and tools," in *Engineering and managing software requirements*, Aybüke Aurum and Claes Wohlin, Eds. Berlin, Heidelberg: Springer, 2005, pp. 19 - 46.

[14] Sacha Martin, Aybüke Aurum, Ross Jeffery, and Barbara Paech, "Requirements engineering process models in practice," in *7th Australian workshop on requirements engineering*, Melbourne, Australia, 2002, pp. 41-47.

[15] Bashar Nuseibeh and Steve Easterbrook, "Requirements engineering: a roadmap," in *Proceedings of the Conference on the Future of Software Engineering*, ACM, 2000, pp. 35-46.

[16] Marcos A. Chiarello and Adolfo Gustavo SS, "An Approach of Software Requirements Elicitation Based on the Model and Notation Business Process (BPMN)," *Lecture Notes on Software Engineering*, vol. 2, no. 1, pp. 65-70, February 2014.

[17] E. Wallmüller, "Risk management for IT and software projects," in *Business continuity*. Berlin, Heidelberg: Springer, 2002, pp. 165 - 175.

[18] Paul Newton, "Managing Project Risk," *Project Skills [available on http://www. free-management-ebooks. com/dldebk-pdf/fme-project-risk. pdf]*, 2015.

[19] David Hillson, "Extending the risk process to manage opportunities," *International Journal of project management*, vol. 20, no. 3, pp. 235 - 240, April 2002.

[20] Olaf Passenheim, *Project Management. bookboon*.: Ventus Publishing ApS, 2009.

[21] Saima Amber, Narmeen Shawoo, and Saira Begum, "Determination of Risk During Requirement Engineering Process," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 3, no. 3, pp. 358 - 364, March 2012.

[22] Shareeful Islam, "Software Development Risk Management Model- a goal-driven approach," Lehrstuhl fur Software & Systems Engineering - Institut fur Informatik - Technische Universitat Munchen, Munchen, PhD Thesis 2011.

[23] David Hillson, "Effective strategies for exploiting opportunities," in *Proceedings of the Project Management Institute Annual Seminars & Symposium*, Nashville, 2001.

[24] Hooman Hoodat and Hassan Rashidi, "Classification and Analysis of Risks in Software Engineering," *World Academy of Science, Engineering and Technology*, vol. 56, no. 32, pp. 446 - 452, August 2009.

[25] Barry Bohem and Daniel Port, "Educating software engineering students to manage risk," in *Proceedings of the 23rd International Conference on Software Engineering*, IEEE Computer Society, 2001, pp. 591 - 600.

[26] Steve McConnell, "Rapid Development," *Interactions-New York*, vol. 4, no. 1, p. 63, 1997.

[27] Andres Norgen, Requirements engineering and prototyping in a legacy software setting, 2004.

[28] Euston Quah and J.B.S Haldane, *Cost-benefit analysis*.: Routledge, 2007.

[29] Jack V. Michaels and William P. Wood, *Design to cost*.: John Wiley & Sons, 1989, vol. 3.

[30] Azham Hussain and Emmanuel O. C. Mkpojiogu, "Requirements: Towards an understanding on why software

projects fail," *AIP Conference Proceedings*, vol. 1761, no. 1, p. 020046, August 2016.

[31] June Verner, Karl Cox, Steven Bleistein, and Narciso Cerpa, "Requirements engineering and software project success: an industrial survey in Australia and the US," *Australasian Journal of information systems*, vol. 13, no. 1, November 2005.

[32] IBM, "Getting requirements right: avoiding," IBM Corporation, New York, USA, White Paper RAW14154-USEN-00, 2009.

[33] Karl Wiegers, "Karl Wiegers describes 10 requirements traps to avoid," *Software Testing \& Quality Engineering*, vol. 2, no. 1, January 2000.

[34] Babu Suresh, "Scope Creep Management," The PROJECT PERFECT White Paper Collection, Sydney, Australia, White Paper 2005.

[35] Jesse Fewell, "The blame game," *PM Network*, vol. 24, no. 7, p. 23, 2010.

[36] Azham Hussain, Emmanuel O.C. Mkpojiogu, and Fazillah Mohmad Kamal, "The role of requirements in the success or failure of software projects," *International Review of Management and Marketing*, vol. 6, no. 7S, January 2016.

[37] John Thorp. (2017, August) The Thorp Network. [Online]. HYPERLINK "http://www.thorpnet.com/" http://www.thorpnet.com/

[38] The Standish Group, CHAOS Report, 1995.

[39] Karl Wiegers and Joy Beatty, *Software Requirments*.: Pearson Education, 2013.

[40] Byron J Williams, Jeffrey Carver, and Rayford B Vaughn, "Change Risk Assessment: Understanding Risks," in *Software Engineering Research and Practice*, 2006, pp. 966 - 971.