International Review of Management and Marketing

ISSN: 2146-4405

available at http: www.econjournals.com

International Review of Management and Marketing, 2016, 6(S7) 306-311.

Special Issue for "International Soft Science Conference (ISSC 2016), 11-13 April 2016, Universiti Utara Malaysia, Malaysia"

The Role of Requirements in the Success or Failure of Software Projects

Azham Hussain[1]*, Emmanuel O. C. Mkpojiogu[2], Fazillah Mohmad Kamal[3]

[1]School of Computing, Universiti Utara Malaysia, Sintok 06010, Malaysia, [2]School of Computing, Universiti Utara Malaysia, Sintok 06010, Malaysia, [3]School of Quantitative Sciences, Universiti Utara Malaysia, Sintok 06010, Malaysia. *Email: azham.h@uum.edu.my

ABSTRACT

Requirements engineering (RE) is pivotal and central to every successful software development project. There are several reasons why software projects fail; however, poorly elicited, documented, validated and managed requirements contribute grossly to software projects failure. Software project failures are normally very costly and risky and these could even a times be life threatening also. Projects that overlook RE processes often suffer or are most likely to suffer from failures, challenges and other consequent risks. The cost of project failures and overruns when estimated is quite great and grave. In addition, software project failures or overruns portend a challenge in today's competitive market environment. It affects negatively the image, goodwill, profitability, and revenue drive of companies and decreases the marketability of their products, as well as, the perceived satisfaction of their customers and clients (which also leads to poor loyalty). In this paper, RE was discussed. Its role in software projects success was elaborated. The place of software requirements process in relation to software project failure was explored and examined. Furthermore, project success, challenge and failure factors were also discussed with emphasis placed on requirements factors as they play a major role in software projects' successes, challenges and failures. The paper relied on secondary statistics to explore and examine factors responsible for the successes, challenges and failures of software projects in large, medium and small scaled software companies.

Keywords: Requirements Engineering Process, Software Projects, Failure, Success
JEL Classifications: L86, M15

1. BACKGROUND

Requirement is a statement about a proposed system that all stakeholders agree must be made true in order for the customers' problems to be truly solved. It is an expression of the ideas to be embodied in a system or an application under development. Requirement is the statement of system service or constraint describing the user-level properties, general systems, specific constraints and needs of clients. However, it may also describe the attributes and behaviour of a system (Inam, 2015). Furthermore, Gupta and Wadhwa (2013) stated that requirement forms the basis for the original assessment and ideas for developing and validating any product. Krauss (2012) further stated that it is critical to defining the purpose and process of a project and it helps to analyse and manage a project. More so, requirement has to do with capturing the objectives and the purpose of a system. It is the conditions or the capability needed by users to solve problems or meet their objectives. The accuracy and quality of requirements immensely contribute to the success of a project/system development (Krauss, 2012). Furthermore, quality requirements are pivotal and key to customer/user product satisfaction (Hussain et al., 2015; Mkpojiogu and Hashim, 2015; 2016; Hussain et al., 2016a; 2016b; 2016c; Hussain and Mkpojiogu, 2016a; 2016b; 2016c).

Every project has some basic requirements that defines what the end users, customers, clients, developers, suppliers or business (i.e., stakeholders) require from it coupled with some needs of the system for efficient functioning. Requirement is a key factor during every software development as it describes what different stakeholders need and how the system will satisfy these needs. It is generally expressed in natural language so that everyone can understand it well. It helps the analyst to better understand which elements and functions are necessary in the development

of a particular software project. More so, requirements are considered as an input to design, implementation and validation phase of software product development. Thus, a software project is successful or a failure during software development because of poor requirement elicitation as well as in requirements managing process (Pfleeger and Atlee, 2006).

RE is one of the branches of software engineering. It is the systematic processes and techniques for requirements elicitation, requirement analysis, specification, verification and management of requirements. It is the initial phase of software engineering process in which user requirements are collected, understood, and specified for developing quality software products. In other words, it is a practical and systematic approach through which the software or system engineer collects functional or non-functional requirements from different customers/clients for the design and development of quality software products (Swarnalatha et al., 2014). Requirements engineering (RE) is an incremental and iterative process, performed in parallel with other software development activities such as design, implementation, testing and documentation. RE process is divided into two main set of activities; namely, requirements development and requirement management (Hussain et al., 2016). Software requirement development mainly covers the activities of discovering, analysing, documenting, verification and validation of requirements whereas software requirement management commonly includes activities related to traceability and dynamic change management of software requirements (Pandey and Suman, 2012; Swarnalatha et al., 2014).

Research reveals that software project failures are mainly due to inadequate requirements, changing requirements, poor requirements, and impracticable expectations, etc. Nonetheless, the application of a systematic approach will reduce the challenges of RE process and the chances of any project failing. It is also very crucial to gather accurate information about the proposed system/product and analyse the organizational needs and practices, document the requirement acquisition and ensure completeness and consistency with stakeholder requirements whilst effectively managing conflicting requirements (Hussain et al., 2015; Mkpojiogu and Hashim, 2015; 2016; Hussain et al., 2016a; Hussain and Mkpojiogu, 2016a). Requirements of software are captured through RE which is the process of determining requirements (Cheng and Atlee, 2009). Cheng and Atlee (2009) mentioned that successful RE involves the discovering of the stakeholders needs, understanding of the requirements contexts, modelling, analysing, negotiating, validating, as well as assessing documented requirements; and managing of the requirements (Shah and Patel, 2014). There are many researches that identify the need for the development of quality software that meet the needs and objectives of the customers and give value to stakeholders (Wiegers, 2013; Inam, 2015). Asghar and Umar (2010) pointed out that RE is acknowledged as the first phase of software engineering process and it is considered as one of the main phases in software development. Furthermore, Khan et al. (2014), and Shah and Patel (2014), asserted that, unclear requirement is the main reason of software project failures. Khan et al. (2014) said that "RE phase is difficult and crucial." Also, Young (2004) stated that the neglect of RE contributes to project failures. RE impacts productivity as well as product quality. Thus, it can be stated that RE is an essential phase for software development (Sankhwar et al., 2014), and therefore RE practices should be taken into consideration in every software development project. In this paper, RE process is defined based on Wiegers (2003). He maintained that RE is composed of two main activities which are: Requirements development and requirements management.

According to Kavitha and Thomas (2011), proper comprehension and management of requirements are the main determinants of success in the process of development of software. In this paper, secondary statistics from previous studies were closely examined and used to assess and succinctly understand why software projects succeed or fail.

In summary, there are many reasons for software project failures; however, poorly engineered requirements process contributes immensely to the reason why software projects fail (Inam, 2015). Software projects failure are usually costly and risky and could also be life threatening. Projects that undermine RE suffer or are likely to suffer from failures, challenges and other attending risks. The cost of project failures and overruns when estimated is very huge. Furthermore, software project failures or overruns pose a challenge in today's competitive market environment. It affects the company's image, goodwill, profitability, and revenue drive and decreases the marketability, and the perceived satisfaction of customers and clients (which leads to their poor loyalty to the company and their products) (Hussain and Mkpojiogu, 2016a; 2016b; 2016c; Hussain et al., 2016b).

The remaining part of this paper is presented as follows: Section 2: Why software projects succeed or fail; Section 3: The role of requirements in software projects success; and lastly, Section 4: Conclusion.

2. WHY SOFTWARE PROJECTS SUCCEED OR FAIL

A software project, as categorized by the Standish Group, can be successful, challenged, or failed. A successful software project is one that is completed on time and within allocated budget, and which has all the originally specified features and functions. A challenged project is one that is completed but with time and budget overrun and also with fewer features and functions when compared to those originally specified. A failed project is one that is aborted or cancelled before its completion. It is also one that is completed, but never implemented (Kamuni, 2015).

2.1. Software Projects' Success, Challenge, and Failure Factors

The Standish Group study of 2009 reported that only 34% of software projects succeeded, 44% were challenged and 22% failed (Kamuni, 2015). The 1995 Chaos report established that RE practices contributed more than 42% of overall project success. Likewise, inappropriate RE practices represent more than 43% of the reasons for software project failure. In addition, many previous

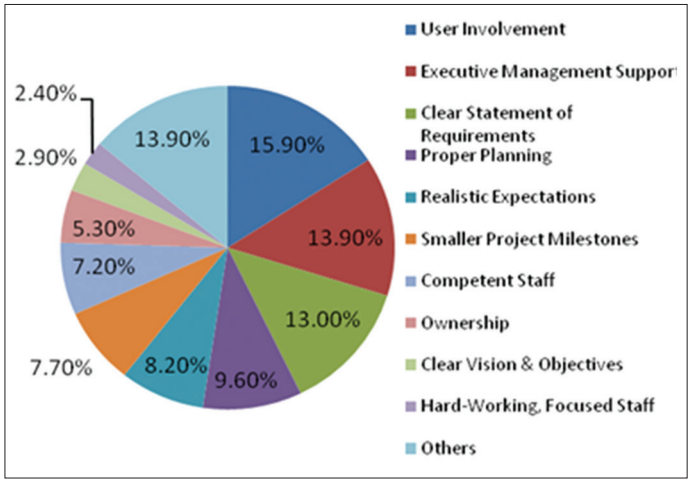
researchers have identified that 70% of the requirements were difficult to identify and 54% were not clear and well organized (Gause and Weinberg, 1989; Asghar and Umar, 2010; Khan and Mahrin, 2014; Young, 2004; Sankhwar et al., 2014; Wiegers, 2003; Kavitha and Thomas, 2011; Kamuni, 2015). The 1995 chaos report lists "incomplete requirements" as the leading cause of software project failure. The Standish Group reports a low point in 1994 in which only 16% of projects were successful (Wiklund and Pucciarelli, 2009). Gause and Weinberg (1989) also pointed out that: (i) Requirements are difficult and challenging to describe in natural language; (ii) requirements have many different types and levels of details; (iii) requirements are difficult to manage if they are not in control; (iv) most of the requirements change during software development. Taimour (2005) identified the following: Poor planning including missing dependencies, requirements changed and not finalized, key requirements missed and high turnover of top IT manager; as reasons why software projects fail. The Standish Group chaos report (1994) show that 29% of all projects succeeded (i.e., delivered on time, on budget, with required features and function); 53% were challenged (i.e., delivered late, over budget and/or with less the required features and functions); and 18% failed (cancelled prior to completion or delivered, but never used). Figures 1-3 display the project success, challenged, and failure factors of the Chaos report as republished by Project Smart. In the Figure 1, user involvement (15.90%), executive management support (13.90%) and clear statement of requirements (13%) are the top three factors responsible for project success.

In Figure 2, lack of user input (12.80%), incomplete requirements and specification (12.30%), and changing requirements (11.80%) are the top three factors responsible for challenged projects. In Figure 3, incomplete requirements (13.10%), lack of user involvement (12.40%), and lack of resources (10.60%) are the top three factors causing impaired or failed projects. From these presentations, it is very clear that requirements related issues are the top factors for affect software project success, challenge, and failure (impairment).

Furthermore, the 1995 Standish Group chaos report (Project Smart, 2014) identified user involvement, executive managerial support, clear statement of requirement, proper planning, realistic expectation, and smaller projects milestones, etc. as success factors and reports incomplete requirements, lack of user involvement, lack of resources, unrealistic expectation, lack of executive support, changing requirements and specifications, lack of planning, etc. as problem causes. Wiklund and Pucciarelli (2009) in their study revealed that 25% of projects fail out-right, 20-25% do not meet return on investment and up to 50% require material rework.

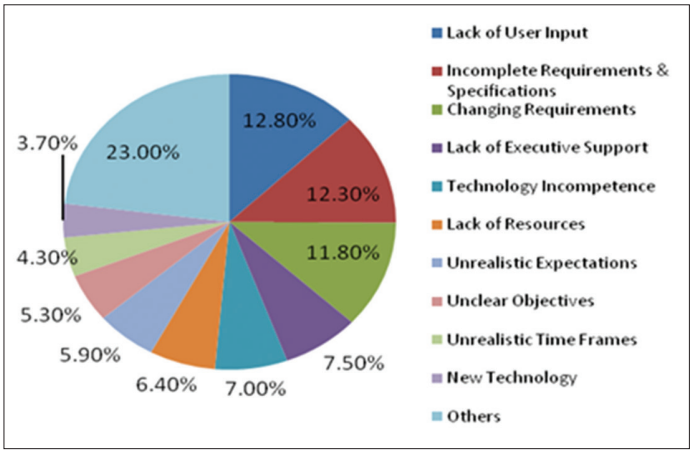
In addition, from the 1995 Chaos report, the figures for failure were equally disheartening in companies of all sizes. Only 9% of projects in large companies were successful. At 16.2% and 28% respectively, medium and small companies were somewhat more successful. A whopping 61.5% of all large company projects were challenged compared to 46.7% for medium companies and 50.4% for small companies. 37.1% of projects were impaired and subsequently cancelled (failed) in medium companies, compared

Figure 1: Project success factors



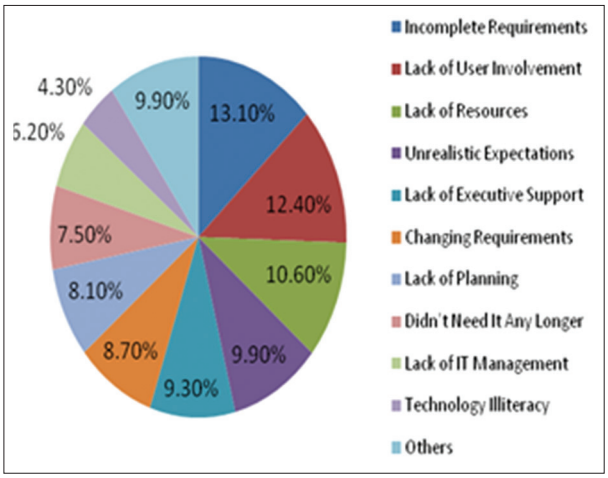
Source: Project Smart (2014)

Figure 2: Project challenge factors



Source: Project Smart (2014)

Figure 3: Project failure factors



Source: Project Smart (2014)

to 29.5% in large companies and 21.6% in small companies (Project Smart, 2014).

The Standish Group categorized software companies into large, medium and small based on their annual income. A large company

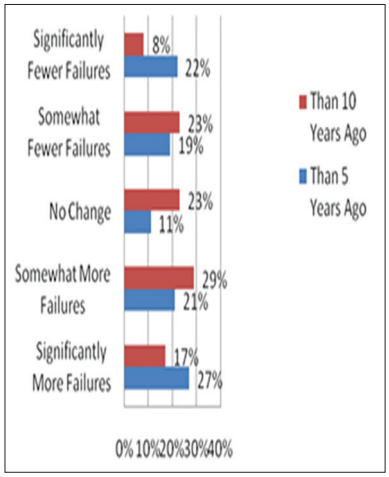
is one with >$500 million in annual revenue. A medium company is one that has between $200 million and $500 million in yearly revenue while a small company has between $100 million to $200 million revenue per year. The Standish Group observed that only 9% of the projects in large companies, 16.2% of projects in medium companies and 28% of projects in small companies were successful. Furthermore, 61% of all large company projects were challenged. Most of the failed projects were within the medium scale company category (37.1%) in comparison to large companies (29.5%) and small companies (21.6%) (Kamuni, 2015). In a related survey by the Standish Group, the success rate was 24% in large software companies, 37.2% in medium scale companies and 48% in small scale software companies. In addition, 69.5% of large software company projects were very challenging, in comparison to 52.7% and 60.4% in medium and small software companies respectively. Also, 39.5% of projects in large software companies were cancelled in comparison to 45.1% and 31.6% in medium and small scale software companies, respectively (Swarnalatha et al., 2015). As could be consistently observed, poor requirements processes are responsible for software projects challenges and failures. A good requirements collection and process contributes to software projects successes.

One of the major causes of both cost and time overruns is restarts. For every 100 projects that start, there are 94 restarts. This does not mean that 94 of 100 will have one restart; some projects can have several restarts (Project Smart, 2014). The most important aspect of the research is discovering why projects fail. To do this, The Standish Group surveyed IT executive managers for their opinions about why projects succeed. The three major reasons why a project will succeed are user involvement, executive management support, and a clear statement of requirements. There are other success criteria, but with these three elements in place, the chances of success are much greater. Without them, chance of failure increases dramatically. The survey participants were also asked about the factors that cause projects to be challenged. Opinions about why projects are impaired and ultimately cancelled ranked incomplete requirements and lack of user involvement at the top of the list (Project Smart, 2014) (Figures 1-3). Another key finding of the survey is that a high percentage of executive managers believe that there are more project failures now than 5 and 10 years ago. This is in spite of the fact that technology has had time to mature (Project Smart, 2014) (Figure 4).

3. THE ROLE OF REQUIREMENTS PROCESS IN SOFTWARE PROJECT SUCCESS

RE is the important phase of software development process. It basically aims at collecting meaningful and well defined requirements from clients in the proper way. It is important to develop quality software that can satisfy user's needs without errors. It is mandatory to apply RE practices at every stage of software development process (Swarnalatha et al., 2014). RE is commonly accepted to be the most important, critical and complex process in the software development process. A well-defined requirement is software functionality that satisfies client's needs (Inam, 2015). The RE process has the highest

Figure 4: Executive managers' perceptions on project failures



Source: Project Smart (2014)

impact on the capabilities of the emerging software product (Swarnalatha et al., 2014). RE is important because it helps to define the purpose of any project by defining the constraints, specifying the process involved and documenting it. It also ensures incremental improvement by matching the most effective measures with the crucial problems. Furthermore, it critically identifies what the stakeholders' need and helps to make decisions more efficiently, hence providing effective results. The success or failure of a project is dependent on the accuracy and effective management of requirements. It is crucial to determine the mix of effective techniques to use for requirement acquisition and properly document the process and the requirements to reduce the challenges and chances of failure. RE should therefore be the starting point and backbone of any project or decision because it helps to determine and focus on the objective, match needs of stakeholders to the product development process thereby increasing the chances of achieving the best result. However, it must be managed throughout the entire system or product development life cycle for project success and the mitigation of failures.

4. CONCLUSIONS

RE is at the foundation of every successful software project. There are many reasons for software project failures; however, poorly engineered requirements process contributes immensely to the reason why software projects fail. Software project failure is usually costly and risky and could also be life threatening. Projects that undermine RE suffer or are likely to suffer from failures, challenges and other attending risks. The cost of project failures and overruns when estimated is very huge. Furthermore, software project failures or overruns pose a challenge in today's competitive market environment. It affects the company's image, goodwill, and revenue drive and decreases the perceived satisfaction of customers and clients. In this paper, RE was discussed. Its role in software projects success was elaborated. The place of software requirements process in relation to software project failure was explored and examined. Also, project success and failure factors were also discussed with emphasis placed on requirements factors as they play a major role in software projects' challenges, successes

and failures. The paper relied on secondary statistics to explore and examine factors responsible for the successes, challenges and failures of software projects in large, medium and small scaled software companies.

In conclusion, the success or failure of any given software development project hinges on how the software requirements process was carried out. The cost or the risks involved in a poorly engineered requirements process are great and sometimes irreparable. RE stands as a bedrock upon which the success of software projects stands. Colossal wastes can be avoided if adequate attention is given to proper RE in all software development projects. In this paper, the connection of requirements to project success or failure is established and emphasized using secondary data analysis from previous studies. As could be consistently observed, poor requirements processes are responsible for software projects challenges and failures while on the other hand, a good requirements collection and process contributes to software projects successes. Thus, it behoves of software project planners, analysts, engineers and managers to incorporate adequate RE process in every software development project to achieve project success and eliminate project failures and challenges.

5. ACKNOWLEDGMENT

This study was funded by Ministry of Higher Education under RAGS grant.

REFERENCES

Asghar, S., Umar, M. (2010), Requirement engineering challenges in development of software applications and selection of customer-off-the-shelf (COTS) components. International Journal of Software Engineering, 1(1), 32-50.

Cheng, B.H.C., Atlee, J.M. (2009), Current and future research directions in requirements engineering. Design Requirements Engineering: A Ten-Year Perspective. Heidelberg: Springer.

Gause, D.C., Weinberg, G.M. (1989), Exploring Requirements: Quality Before Design. New York: Dorset House.

Gupta, S., Wadhwa, M. (2013), Requirement engineering: An overview. International Journal of Research in Engineering and Technology, 1(2), 155-160.

Hussain, A., Mkpojiogu, E.O.C. (2016a), An application of Kano method in the elicitation of stakeholder satisfying requirements for an e-Ebola awareness system. International Journal of Systems Applications, Engineering and Development, 10, 169-178.

Hussain, A., Mkpojiogu, E.O.C. (2016b), Requirements model for an e-health awareness portal, 1st International Soft Science Conference (ISSC'16), Langkawi Island, Malaysia, 11-13 April, 2016.

Hussain, A., Mkpojiogu, E.O.C. (2016c), Predicting the perceived worth of software products requirements with customer satisfaction. Advanced Research in Engineering and Information Technology International Conference (AREITIC'16), 31 May - 2 June, 2016, Bandung, Indonesia.

Hussain, A., Mkpojiogu, E.O.C., Abdullah, I. (2016a), Investigation of Current Requirements Engineering Practices Among Software Developers at the Universiti Utara Malaysia Information Technology (UUMIT) Centre. 1st International Soft Science Conference (ISSC'16), Langkawi Island, Malaysia, 11-13 April, 2016.

Hussain, A., Mkpojiogu, E.O.C., Hassan, F. (2016b), Assessing the influence of self-reported requirements importance on the perceived quality of proposed software products. 2nd International Conference on Information and Communication Technology for Transformation (IC-ICT4T'16), 5-7 April 2016, Kota Kinabalu, Sabah, Malaysia.

Hussain, A., Mkpojiogu, E.O.C., Husin, Z. (2016c), Requirements: Towards an understanding on why software projects fail. 1st International Soft Science Conference (ISSC'16), 11-13 April 2016, Langkawi, Island, Malaysia.

Hussain, A., Mkpojiogu, E.O.C., Kamal, F.M. (2015), Eliciting user satisfying requirements for an e-health awareness system using kano model. Proceedings of the 14th WSEAS International Conference on Computer and Computational Science (ACACOS'15), Kuala Lumpur, Malaysia.

Inam, A. (2015), A Study of Requirements Engineering Practices Among Software Developers at UUM Information Technology. MSc. Dissertation Report. Malaysia: Universiti Utara Malaysia.

Kamuni, S.K. (2015), Study of Factors that Induce Software Project Overrun Time, Mechanical and Manufacturing Engineering. Paper, 10.

Kavitha, C.R., Thomas, S.M. (2011), Requirement gathering for small projects using agile methods. IJCA Special Issue on Computational Science-New Dimensions and Perspectives, 3, 122-128.

Khan, H.H., Mahrin, M. (2014), Factors generating risks during requirement engineering process in global software development environment. International Journal of Digital Information and Wireless Communications (IJDIWC), 4(1), 63-78.

Krauss, E. (2012), Requirement Engineering and Project Management. Heidelberg: Springer.

Mkpojiogu, E.O.C., Hashim, N.L. (2015), Quality-based prioritization: An approach for prioritizing software requirements. 2015, 2nd Advancement on Information Technology International Conference (ADVCIT'15), Krabi, Thailand, 3-5 December, 2015.

Mkpojiogu, E.O.C., Hashim, N.L. (2016), Understanding the relationship between Kano model's customer satisfaction scores and self-stated requirements importance. Springerplus, 5(1), 1-22.

Pandey, D., Suman, U. (2012), An effective requirements engineering process model for software development & requirements management. International Conference on Advances in Recent Technologies in Communications and Computing. p287-291.

Pfleeger, S.L., Atlee, J.M. (2006), Software Engineering: Theory and Practice. London, UK: Pearson, India.

Project Smart. (2014), The Standish Group, 1995 Chaos Report. Available from: https://www.projectsmart.co.uk/whitepapers/chaos-report.pdf.

Sankhwar, S., Singh, V., Pandey, D. (2014), Requirement engineering paradigm. Global Journal of Multidisciplinary Studies, 3(3), 1-8.

Shah, T., Patel, V.S. (2014), A review of requirement engineering issues and challenges in various software development methods. International Journal of Computer Applications, 99(15), 36-45.

Swarnalatha, K.S., Srinivasan, G.N., Dravid, N., Kasera, R., Sharma, K. (2014), A survey on software requirements engineering for real time projects based on customer requirements. International Journal of Advanced Research in Computer and Communication Engineering, 3(1), 5045-5050.

Swarnalatha, K.S., Srinivasan, G.N., Rakesh, R., Dwivedi, V. (2015), Software Requirements Collection Enhancement Using Sampling Technique and Applying T-Distribution. Available from: http://www.ijsetr.com.

Taimour, A. (2005), Why IT Projects Fail. Available from: http://www.projectperfect.com.au.

The Standish Group. (1994), 1994 Chaos Report. Available from: http://www.standishgroup.com/services.php.
Wiegers, K. (2013), Creating a Software Engineering Culture. Reading, MA: Addison-Wesley.
Wiegers, K.E. (2003), Software Requirements: Practical techniques for gathering & managing requirement through the product development cycle. USA: Microsoft Corp.
Wiklund, D., Pucciarelli, J. (2009), Improving IT Projects Outcomes by Systematically Managing and Hedging Risk: An IDC Insight Research Document.
Young, R.R. (2004), The Requirements Engineering Handbook. Norwood, MA: Artech House.