Research Journal of Applied Sciences, Engineering and Technology 8(19): 2082-2091, 2014
DOI:10.19026/rjaset.8.1201
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: May 26, 2014 Accepted: July 13, 2014 Published: November 20, 2014

Research Article

The Study of Available Techniques for Existing Requirements Engineering Challenges Based on Literature Review Evidences

Souhaib Besrour, Lukman Bin Ab Rahim and P.D.D. Dominic
Department of Computer and Information Sciences, Universiti Teknologi PETRONAS,
31750 Tronoh, Perak, Malaysia

Abstract: It is well known that software engineering suffer from various challenges. Moreover, numerous researchers found that project challenges have a negative effect on project time cost and user satisfaction. Additionally, Numerous Requirements Engineering (RE) methods have been proposed to improve the quality of requirements documents and to increase customer satisfaction about final product. Nevertheless, the choosing between various techniques may be confusing and puzzling. Therefore, this study aims to present, Literature review based study to link between RE challenges and available techniques to eliminate challenges using the utmost appropriate technique. Study conclusions are relevant for both industry and academic researchers in order to achieve effective software requirement engineering.

Keywords: Comparison criteria, literature review based study, requirements engineering, software requirement, software requirement techniques

INTRODUCTION

Various researches (Wieringa *et al*., 2005), found that there is inconsistence and incontinence between available techniques and existing RE challenges. Other researches (Kotonya and Sommerville, 1998) ensure about how critical software requirement. Therefore, this study aims to highlight existing challenges faced in RE process and the appropriate technique to eliminate those problems. A total of 18 challenges have been highlighted briefly. Moreover a total of 15 techniques presented in order to eliminate the effect of RE challenges. The research (Wieringa *et al*., 2005), discusses about the gap between techniques and existing RE challenges. All in all this study presents RE challenges and the appropriate techniques and solutions based on well supported evidences from literature. Next section presents brief about RE process structure.

RE PROCESSES

It contains several steps and procedures that should be followed in order to achieve a successful requirement process. Elicitation is a major process in software requirement engineering, it is the process of gathering and acquiring requirement for a computer based system. It purposes to gather client requirements, system constrains and goals. Requirement elicitation process is a compound process where clients' needs need to be understood correctly to obtain the correct requirement. It needs adequate expertise in dealing with social issues and software requirement processes. Various techniques are available for collecting requirements such as: interview, brainstorming, Card sorting (Spencer, 2009) and Joint Application Development (JAD) (Didar and Coulin, 2005). The second process in software requirement engineering is analysis; it targets to breakdown requirement meanings and structures. Analysis process aims to answer "what" to build rather than "how" to build. The chief techniques for analyzing software requirement are: Scenario based analysis (Use-case), Kano model Analysis, Decision table based-specification and Goal-Oriented (Chung and Supakkul, 2005). The third process in software requirement engineering is specification; it purposes to record and document system requirement in a clear format and specify client needs accurately and correctly. Accordingly, even after finishing the entire project, software requirement specification can be used as a contract document and as a strong base of additional system enhancements. There are various techniques for software requirement specification such as: IEEE Software Requirement Specification (SRS), ERD-based specification, Structured Natural Language Specification. Lastly, the fourth process in requirement engineering is

Corresponding Author: Souhaib Besrour, Department of Computer and Information Sciences Universiti Teknologi PETRONAS, 31750 Tronoh, Perak, Malaysia

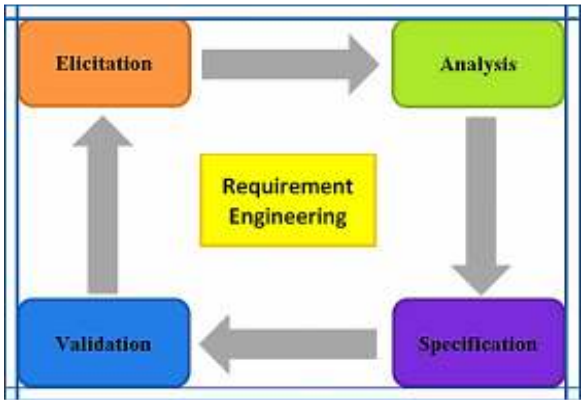

Fig. 1: Requirement engineering processes

requirement verification: it is the process of testing requirements correctness and conforming that clients' needs are correctly interpreted. Errors and faults can easily be fixed through the early periods of building system. Accordingly software requirement errors are very costly to repair and fix after the system is completely shaped. Thus, requirement verification process plays a vibrant role in reducing developed system cost dramatically. There are various techniques for requirement verification such as peer review validation (Xiong and Litman, 2010), Ad-hoc based validation (Saqi and Ahmed, 2008), Checklist-based validation (Porter *et al*., 1995) and Misuse-case (Whittle *et al*., 2008) (Fig. 1).

RE CHALLENGES AND LITERATURE REVIEW EVIDENCES

This section presents a total of 18 requirement engineering challenges, which they are discussed separately in order to present further details about each challenge. Furthermore, proposed technique has been presented in order to eliminate those challenges.

Elicitation process: This process encompasses five key challenges: First, exclusion of stakeholders' identification, second, poor communication during requirement elicitation, third undefined functional requirement, fourth, undefined non-functional requirement, fifth, exclusion considerations of organizational and social issues. Aforementioned challenges have been discussed briefly with proposed techniques based on literature review evidences.

First, Base on Bourne (2009, 2010) exclusion of stakeholder's identification is one challenge that may affect project quality. Stakeholders, refers to entity that have direct interaction with system. Moreover, stakeholders could be a human, a system or any other entity that communicate with the system. Based (Alexander, 2005) the exclusion of stakeholders identification may results incomplete requirement and low integrity. Kamata and Tamai (2007) have shown that numerous large projects fail because of requirement errors (Sommerville, 2010). In the other hand, Brainstorming is a lateral thinking process and is designed to improve thinking patterns into new ways of looking at things. Participants in the brainstorming process can be from wide range of disciplines. This brings a broad range of experiences to the session and helps to make it more creative (Mohd Kasirun and Salim, 2008; Herrmann and Nolte, 2010; Herrmann and Nolte, 2010). Based on Nuseibeh Easterbrook (2000) Brainstorming is core technique in requirement elicitation process. Furthermore, the research don by Scheinholtz and Wilmont (2011) reveal that Brainstorming used to eliminate Exclusion of Stakeholders identification (Litchfield, 2008). This hypothesis was also proved and supported by Mohd Kasirun and Salim (2008).

Second, Base on Zave (1997) poor communication during requirement elicitation is one challenge during requirement gathering. Moreover, before completing requirement gathering process it is hard to know what is inside customer thoughts. The Research Kamata and Tamai (2007) have shown that numerous large projects fail because of inadequate requirement process. This inadequacy is often related to requirement elicitation and social issues (Goguen and Linde, 1992). In the other hand, interview is a conversation between two or more people where questions are asked by the interviewer to elicit facts or statements from the interviewee (Burke and Miller, 2001). Based on Lloyd *et al*. (2002) interview is major technique in requirement elicitation. Moreover, it is used to elicit information, requirement and system constrains. Furthermore, the research don by Scheinholtz and Wilmont (2011) and Opdenakker (2006) reveal that interview used to eliminate poor communication during requirement elicitation. This hypothesis was also proved and supported by Goguen and Linde (1992).

Third and fourth, based on Glinz (2007) undefined functional and non-functional requirement is critical problem that may affect project success. Non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. Additionally, functional requirement defines a function of a system or its component. A function is described as a set of inputs, the behavior and outputs. In the other hand, functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Although aforementioned terms have been used for more than two decades, there is still various fails because of undefined functional and non-functional requirements (Ullah *et al*., 2011; Firesmith, 2007). In the other hand, JAD was originally developed for internal use at IBM (Davison, 2000). It is a

technique used to gather information and system constrains by conducting a structured meeting. During JAD session, users will be involved in intensive discussion and conversation to clarify ambiguous and complex perspectives (Carmel *et al*., 1993). Based on Nuseibeh and Easterbrook (2000) JAD is key technique in requirement elicitation process. Additionally, the research don by Duggan and Thachenkary (2003) reveal that JAD used to eliminate undefined functional/ non-functional Requirements and system constraints. This hypothesis was also proved and supported by Davidson (1999).

Fifth, Based on Exclusion considerations of social and organizational issue may produce incomplete requirement data. Moreover, it is hard to find two projects that have completely similar requirement and social issues. Therefore, social and organizational issue has to be sensitively conducted due to uniqueness of each project. Built projects effects by various variables such as country low, project budget, user preferences, company policy, gender different preferences, organization culture. Base on Zave (1997) the Elimination of social and organizational issue may affect the project quality and project success. In the other hand, Card sorting was originally developed by psychologists as a method to the study of how people organize and categorize their knowledge. In the world of information technology, information architects and developers of desktop and Web-based software applications are faced with the problem of organizing information items, features and functions to make it easier for users to find them. Card sorting can be an effective means of discovering the optimal organization of information for potential users' viewpoint (Wood and Wood, 2008). Based on Nuseibeh and Easterbrook (2000) Card sorting is chief technique in requirement elicitation process. Moreover, the research don by Spencer (2009) reveal thatCard sorting used to eliminate exclusion considerations of social and organizational issue. This hypothesis was also proved and supported by Nurmuliani *et al*. (2004).

ANALYSIS PROCESS

This process encompasses four key challenges: First, execution of analyzing complex requirement, second, exclusion of requirements prioritization, third, exclusion of understanding and modeling functional requirements, Fourth exclusion of understanding non-functional requirements. Aforementioned challenges have been discussed briefly with proposed techniques based on literature review evidences.

First, the phenomenon of large-scale, highly-complex systems is not limited to NASA and the Defense Department, but has extended to the commercial infrastructure as well (Carr, 2000). In last decade software industry became more and more challenging for different response. Thus designed system became more complicated to face those challenges. Based on Heninger (1980) and Roman (1985) Execution of analyzing complex requirement is requirement challenge and may produce severe problem to project. In the other hand, decision Table is a precise yet compact way to model complicated logic. It is effective techniques used to analyses complex requirements and system constrains (Becker, 1998). Based on Subramanian *et al*. (1992) Decision Table is principal technique in requirement analysis process. Moreover, the research done by Kohavi and Daniel (1998) and Dai *et al*. (2013) reveal that Decision Table used to eliminate Execution of analyzing complex requirement. This hypothesis was also proved and supported by Becker (1998) and Huysmans *et al*. (2011).

Second, prioritization is a crucial step towards making good decisions regarding product planning for single and multiple releases. Various aspects of functionality are considered, such as importance, risk, cost, etc. Prioritization decisions are made by stakeholders, including users, managers, developers, or their representatives (Berander and Andrews, 2005). Requirement prioritization is used in Software industry for determining which candidate requirements of a software product should be included in a certain release. Requirements are also prioritized to minimize risk during development so that the most important or high risk requirements are implemented first. Noteworthy that, execution of Requirements prioritization has negative effect to project and overall time consuming (Firesmith, 2004; Lehtola *et al*., 2004). In the other hand, Kano Analysis is analysis techniques used to provide an effective categorization of customer requirements and to understand their nature. Kano's classifies customer preferences into various categories. Additionally, customer is not always having same level of satisfaction about system requirements and constrains (Chaudha *et al*., 2010). Based on Sauerwein *et al*. (1996) Kano Analysis is an important technique in analysis process. Furthermore, the research done by Baek *et al*. (2009) and Von Dran *et al*. (1999) reveal that Kano Analysis used to eliminate Execution of Requirements Prioritization. This hypothesis was also proved and supported by Lai *et al*. (2004).

Third, Based on Carr (2000), Roman (1985) and Lutz (1993) Exclusion of Modeling and understanding Functional requirements has a negative effect to the project. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish (Sommerville, 2010). Noteworthy that requirements analysis is critical to the success of a systems or software project (Abran *et al*., 2005). In the other hand, use case is a list of steps, typically defining interactions between a role and a

system, to achieve a goal. The actor can be a human or an external system. In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stakeholder goals. As an important requirement technique, use cases have been widely used in modern software engineering over the last two decades (Siau and Lee, 2004; El-Attar and Miller, 2007). Based on Sendall (2003) Use-case is principal technique in requirement analysis process. Moreover, the research don by Génova1 *et al*. (2005) reveal that use-case used to eliminate Exclusion of Modeling and understanding Functional requirements. This hypothesis was also proved and supported by García, *et al*. (2004).

Fourth, Based on Glinz (2007) and Ullah *et al*. (2011) exclusion of understanding non-functional requirements and constraints of the system is critical challenge during system development. Non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. Non-functional requirements are often called qualities of a system. Other terms for non-functional requirements are "constraints", "quality attributes", "quality goals", "quality of service requirements" and "non-behavioral requirements" (Stellman and Greene, 2005). Consequently the exclusion of understanding non-functional has a serious effect on project success (Glinz, 2007). In the other hand, Goal-Oriented is model that allows representing non-functional requirements using actors and dependencies instead of components and connectors. It offers a better analysis in the requirements stage since requirements are explicitly specified in goal-oriented models in order to support reasoning about organizational objectives, alternatives and implications, thus having a deep understanding about the domain (Grau and Franch, 2007). Based on Van Lamsweerde (2001) Goal-Oriented is principal technique in requirement analysis process. Furthermore, the research don by Cysneiros and Sampaio do Prado Leite (2004) and Chung and Sampaio do Prado Leite (2009) reveal that Goal-Oriented used to eliminate Exclusion of understanding non-functional requirements and constraints of the system. This hypothesis was also proved and supported by Aguilar *et al*. (2011) and Chung and Supakkul (2005).

Specification process: This process encompasses four key challenges: First, poor-defined specification structure and system terminology, Second, exclusion of documenting functional requirements, Third, exclusion of documenting non-functional requirements, Fourth, exclusion of documenting the relationship among requirements. Aforementioned challenges have been discussed briefly with proposed techniques based on literature review evidences.

First, poor-defined specification structure and system terminology it refers to Natural Language (NL) syntactically ambiguous and semantically inconsistent. Natural language is syntactically ambiguous and semantically inconsistent. Hence, the NL specifications of software requirements can not only result in erroneous and absurd software designs and implementations but the informal nature of NL is also a main obstacle in machine processing of NL specification of the software requirements (Umber and Bajwa, 2011). Natural language is flexible and wide-spread, but unfortunately also inherently ambiguous. Even worse, often neither customers nor software developers recognize an ambiguity and each derives an interpretation that differs from that of others without noticing this difference. Consequently, software developers design and implement a system that does not behave as intended by the customers. Additionally, NL lack of clear structure to produce good requirements (Sommerville, 2010). In the other hand, numerous studies such as Jiang (2005) and Sommerville and Sawyer (1997) present guidelines and good practices. In order to have a better structured natural language specification. Based on Kandt (2003) Structured NL is an essential technique in specification process. Furthermore, the research done by Cleuziou *et al*. (2007) and Ferrari *et al*. (2013) reveal that Structured NL used to eliminate poor-defined specification structure and system terminology. This hypothesis was also proved and supported by Tablan *et al*. (2008) and Sneed and Verhoef (2013).

Second and third, Based on Giakoumakis and Xylomenos (1996) Exclusion of documenting Functional and non-functional requirements leads to poor requirement documentation. Moreover, the way in which requirements are documented plays an important role in ensuring that they can be read, analyzed, (re-) written and validated (Nuseibeh and Easterbrook, 2000; Juristo *et al*., 2002). In the other hand, Software Requirements Specification (SRS) is a requirements specification for a software system. It is a complete description of the behavior of a system to be developed. SRS contains non-functional requirements section: it is constraints on the design or implementation (such as performance engineering requirements, quality standards, Maintainability, Portability and Availability). Furthermore, SRS contains functional requirement constrains such as “System interfaces, User interfaces constrains, Hardware constrains, Software constrains and Communications constrains” (IEEE Computer Society, 1998). The SRS document enlists all necessary requirements that are required for the project development. To derive the requirements we need to have clear and thorough understanding of the products to be developed. This is prepared after detail communications with the project team and customer (IEEE Computer Society, 1998). Based on Sommerville

(2010) SRS is an essential technique in specification process. Furthermore, the research don by Jiang (2005) reveal that SRS used to eliminate Exclusion of documenting Functional and non-functional requirements. This hypothesis was also proved and supported by Giakoumakis and Xylomenos (1996) and Kandt (2003).

Fourth, Based on Kesh (1995) and Ochoa *et al*. (2009) Exclusion of documenting the relationship among requirements, the link between requirements and stakeholders may leads to incomplete requirement specification. Therefore due to incomplete requirement specification, project quality may affect negatively (Roman, 1985). In the other hand, in software engineering, an Entity Relationship Diagram (ERD) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way. The entity-relationship model can be used as a basis for unification of different views of data. This model incorporates some of the important semantic information about the real world. The main ERD model components are first, entities: is a piece of data-an object or concept about which data is stored. Second the relationships that can exist among them (how the data is shared between entities). Based on Jiang (2005) ERD based specification is an essential technique in specification process. Furthermore, the research don by Cagiltay *et al*. (2013) reveal that ERD used to eliminate exclusion of documenting the relationship among requirements and the link between requirements and stakeholders. This hypothesis was also proved and supported by Song *et al*. (1995) and Yeh *et al*. (2008).

Validation process: This process encompasses four key challenges: First, exclusion of ensuring correctness of requirements. Second, exclusion of ensuring completeness of requirements. Third, exclusion of ensuring unambiguity of the requirements. Fourth, exclusion of defining requirements redundancy. Fifth, exclusion of ensuring stakeholders' satisfaction of the requirements. Aforementioned challenges have been discussed briefly with proposed techniques based on literature review evidences.

First, correctness of a requirements specification describes the correspondence of that specification with the real needs of the intended users much the same way that correctness of a piece of software refers to the agreement of the software part with its specification. Based on Zowghi and Gervasi (2003) exclusion of ensuring requirements correctness is common problem during requirement validation. Moreover it is a Symptom of serious requirements problems (Carr, 2000). Midsized systems often have hundreds of requirements and many large systems can end up with several thousand separate requirements. Therefore, with large amount of requirement, exclusion of ensuring requirements correctness became possible to happen during project construction. In the other hand, Ad-hoc based validation is a popular technique used in requirement validation process. With Ad-hoc technique, no guidance is provided during inspection, however it depends on reviewers' knowledge and experience to identify the defects in the document (Saqi and Ahmed, 2008). Based on Fusaro *et al*. (1997) Ad-hoc technique is an essential technique in validation process. Furthermore, the research don by Porter *et al*. (1995) reveal that Ad-hoc based validation used to eliminate exclusion of ensuring requirements correctness. This hypothesis was also proved and supported by Singer (2013).

Second and third, because missing requirements are much harder to spot during requirements evaluations than incorrect or poorly-specified requirements, their absence is often missed until the system is integrated, undergoing operational testing, being manufactured, or being deployed. Worst case scenario, the missing requirements may not be discovered until the system is in use by hundreds, thousands, or an even larger number of users. Such requirements are typically much more difficult and expensive to fix, especially if they are architecturally-significant requirements (Firesmith, 2007). Noteworthy, user satisfaction is generally regarded as one of the most important measures of Information Systems success. User satisfaction has received considerable attention of researchers since the 1980s as an important surrogate measure of information systems success (Ives *et al*., 1983; Bailey and Pearson, 1983; Baroudi *et al*., 1986; Benson, 1983). In the other hand, Checklist-based validation used to reduce failure by compensating for potential limits of human memory and attention. It helps to ensure completeness and user satisfaction in carrying out a task. In software engineering checklist based is one of the commonly used techniques. According to Laitenberger and DeBaud (2000), check-list based reading technique is used to be a standard reading technique in most of the organizations. It contains set of items which guides the reviewer/ inspector during review/inspection. Check list based reading technique includes set of elements which are related to quality of the requirements (Laitenberger, 2002). Based on Sommerville (2010) Checklist-based validation is an essential technique in validation process. Furthermore, the research don by Porter *et al*. (1995) reveal that Checklist technique is used to eliminate exclusion of ensuring completeness of requirements and to ensuring stakeholders' satisfaction of the requirements. This hypothesis was also proved and supported by Fusaro *et al*. (1997) and Chen *et al*. (2006).

Fourth, Based on Opdahl and Sindre (2009) exclusion of ensuring security issues in requirements is critical error and may threaten developed system. The

penetration of computerized information systems into almost every aspect of society, especially when combined with their increasingly ubiquitous nature, has made society more vulnerable to security breaches in these systems. At the same time, the tendency towards larger systems that are distributed over the Internet has introduced many new security threats. Hence, there is an increased need to focus on security requirements when developing new information systems. Based on Firesmith (2003) security requirements are often poorly understood by software practitioners and, as a result, security issues are often not considered until late design or coding, or even patched in later after security defects are discovered in a fielded application. This late handling of security concerns can be very costly (Jurjens, 2002) if the chosen design turns out not to enable the wanted level of security. In the other hand, Misuse-case is a process modeling technique used in the software development industry (Sindre and Opdahl, 2004). The term Misuse Case or mis-use case is derived from and is the inverse of use case. The term was first used in the 1990s by GuttormSindre of the Norwegian University of Science and Technology and Andreas L. Opdahl of the University of Bergen, Norway. Noteworthy, Misuse Case is valuable in threat and hazard analysis, system design, eliciting requirements and generating test cases. In the other hand, misuse Case highlights something that should not happen (i.e., a Negative Scenario). It describes the process of executing a malicious act against a system (Whittle *et al*., 2008). Based on Opdahl and Sindre (2009) Misuse-case is an essential technique in validation process. Furthermore, the research don by (Alexander, 2003) reveal that Misuse-case technique issued to eliminate exclusion of ensuring security issues in requirements. This hypothesis was also proved and supported by John and Gunnar (2006) and Tøndel *et al*. (2010).

Fifth, Based on Didar and Vincenzo (2003) consistency requires that no, two or more requirements in a specification contradict each other. Moreover inconsistency may obstruct project expected goal and may lead to various errors in following software processes. Consistency it is also often regarded as the case where words and terms have the same meaning throughout the requirements specifications. These two views of consistency imply that mutually exclusive statements and clashes in terminology should be avoided. In the other hand, peer review is the evaluation process done by one or more people of similar competence to validate a specific task. It constitutes a form of self-regulation by qualified members of a profession within the relevant field. Peer review methods are employed to maintain standards of quality improve performance and provide credibility. In academia paper peer review is often used to determine an academic paper's quality and suitability for publication. In software engineering, one of the methods to validate system requirement is by peer review technique. It aims to evaluate requirement and validate its contents and structure. Noteworthy, numerous errors caused by human nature mistakes such as forgetfulness and omission (Ragone *et al*., 2013). Based on Saqi and Ahmed (2008) peer review is an essential technique in validation process. Furthermore, the research don by Ragone *et al*. (2013) reveal that Misuse-case technique is used to eliminate Exclusion of ensuring consistency of the requirements. This hypothesis was also proved and supported by He *et al*. (2008) and Xiong and Litman (2010).

CONCLUSION

This study presents, literature review based study to bond between RE challenges and existing techniques. Moreover this study aims to eliminate RE challenges by proposing appropriate technique for each challenge. A total of 18 challenges have been presented with supportive evidences from literature review. In the other hand a total of 15 techniques have been introduced, in order to eliminate afore mentioned challenges and improve industry ability to face existing RE obstacles. All in all this entire study finding is relevant for both industry and academic researchers in order to eliminate effect of challenges and to have a decent RE quality.

ACKNOWLEDGMENT

I would like to thank University Teknologi PETRONAS, which had given me the opportunity to study and conduct my reach in faculty of computer and information science. Furthermore, I would like thank CIS department staff for their guidance and dedication.

REFERENCES

Abran, A., J.W. Moore, P. Bourque and R. Dupuis, 2005. Software Requirements. Guide to the Software Engineering Body of Knowledge. Version 2004, IEEE Computer Society Press, Los Alamitos, California.

Aguilar, J.A., I. Garrigós, and J.N. Mazón, 2011. A goal-oriented approach for optimizing non-functional requirements in web applications. In: De Troyer, O. *et al*., (Eds.), ER 2011 Workshop. LNCS 6999, Springer-Verlag, Berlin, Heidelberg, pp: 14-23.

Alexander, I., 2003. Misuse cases: Use cases with hostile intent. IEEE Software, 20(1): 58-66.

Alexander, I.F., 2005. A taxonomy of stakeholders: Human roles in system development. Int. J. Technol. Hum. Interaction, 1(1): 23-59.

Baek, S.I., S.K. Paik and W.S. Yoo, 2009. Understanding key attributes in mobile service: Kano model approach. In: Smith, M.J. and G. Salvendy (Eds.), Human Interface, Part II, HCII 2009. LNCS 5618, Springer-Verlag, Berlin, Heidelberg, pp: 355-364.

Bailey, J.E. and S.W. Pearson, 1983. Development of a tool for measuring and analyzing computer user satisfaction. Manage. Sci., 29(5): 530-545.

Baroudi, J.J., M.H. Olson and B. Ives, 1986. An empirical study of the impact of user involvement on system usage and information satisfaction. Commun. ACM, 29(3): 232-238.

Becker, B.G., 1998. Visualizing decision table classifiers. Proceeding of IEEE Symposium on Information Visualization, pp: 102-105.

Benson, D.H., 1983. Field study of end-user computing: Findings and issues. MIS Quart., 7(4): 35-45.

Berander, P. and A. Andrews, 2005. Requirements prioritization. In: Aurum, A. and C. Wohlin, (Eds.), Engineering and Managing Software Requirements, Springer, Heidelberg, pp: 69-94.

Bourne, L., 2009. Stakeholder Relationship Management. Gower Publishing, USA.

Bourne, L., 2010. Using the Stakeholder Circle methodology for more effective stakeholder engagement of senior management. Proceeding of 7th Project Management National Benchmarking Forum PMI Chapter, Rio de Janeiro, Brazil.

Burke, L.A. and M.K. Miller, 2001. Phone interviewing as a means of data collection: Lessons learned and practical recommendations. Qual. Soc. Res., 2(2): Art. 7.

Cagiltay, N.E., G. Tokdemir, O. Kilic and D. Topalli, 2013. Performing and analyzing non-formal inspections of Entity Relationship Diagram (ERD). J. Syst. Software, 86: 2184-2195.

Carmel, E., R.D. Whitaker and J.F. George, 1993. PD and joint application design: A transatlantic comparison. Commun. ACM, 36(6): 40-48.

Carr, J.J., 2000. Requirements engineering and management: The key to designing quality complex systems. TQM Mag., 12(6): 400-407.

Chaudha, A., R. Jain, A.R. Singh and P.K. Mishra, 2010. Integration of Kano's Model into Quality Function Deployment (QFD). Int. J. Adv. Manuf. Tech., 53: 689-698.

Chen, T.Y., P.L. Poon, S.F. Tang, T.H. Tse and Y.T. Yu, 2006. Applying testing to requirements inspection for software quality assurance. Inform. Syst. Control J., Vol. 6.

Chung, L. and S. Supakkul, 2005. Representing NFRs and FRs: A goal-oriented and use case driven approach. In: Dosch, W., R.Y. Lee and C. Wu (Eds.), SERA 2004. LNCS 3647, Springer-Verlag, Berlin, Heidelberg, pp: 29-41.

Chung, L. and J.C. Sampaio do Prado Leite, 2009. On non-functional requirements in software engineering. In: Borgida, A.T. *et al*. (Eds.), Mylopoulos Festschrift. LNCS 5600, Springer-Verlag, Berlin, Heidelberg, pp: 363-379.

Cleuziou, G., L. Martin and C. Vrain, 2007. Structuring natural language data by learning rewriting rules. In: Muggleton, S., R. Otero and A. Tamaddoni-Nezhad (Eds.), ILP 2006. LNAI 4455, Springer-Verlag, Berlin, Heidelberg, pp: 125-138.

Cysneiros, L.M. and J.C. Sampaio do Prado Leite, 2004. Nonfunctional requirements: From elicitation to conceptual models. IEEE T. Software Eng., 30(5): 328-350.

Dai, J., W. Wang and Q. Xu, 2013. An uncertainty measure for incomplete decision tables and its applications. IEEE T. Cyb., 43(4): 1277-1289.

Davidson, E.J., 1999. Joint Application Design (JAD) in practice. J. Syst. Software, 45(3): 216-223.

Davison, R., 2000. The role of groupware in requirements specification. Group Decis. Negot., 9: 149-160.

Didar, Z. and G. Vincenzo, 2003. On the interplay between consistency, completeness, and correctness in requirements evolution. Inform. Software Tech., 45: 993-1009.

Didar Z. and C. Coulin, 2005. Requirements Elicitation: A Survey of Techniques, Approaches, and Tools. Maté, JL & Silva, A (Eds.), Idea Group, USA.

Duggan, E.W. and C.S. Thachenkary, 2003. Higher quality requirements: Supporting joint application development with the nominal group technique. Inform. Technol. Manag., 4: 391-408.

El-Attar, M. and J. Miller, 2007. Producing robust use case diagrams via reverse engineering of use case descriptions. Softw. Syst. Model., 7: 67-83.

Ferrari, A., S. Gnesi and G. Tolomei, 2013. Using clustering to improve the structure of natural language requirements documents. In: Doer, J. and A.L. Opdahl (Eds.), REFSQ 2013. LNCS 7830, Springer-Verlag, Berlin, Heidelberg, pp: 34-49.

Firesmith, D., 2003. Engineering security requirements. J. Object Technol., 2: 53-68.

Firesmith, D., 2004. Prioritizing requirements. J. Object Technol., 3(8): 35-47.

Firesmith, D., 2007. Common requirements problems, their negative consequences and the industry best practices to help solve them. J. Object Technol., 6(1): 17-33.

Fusaro, P., F. Lanubile and G. Visaggio, 1997. A replicated experiment to assess requirements inspection techniques. Empir. Softw. Eng., 2(1): 39-57.

García, J.D., J. Carretero, J. Maria Perez and F. García, 2004. A model for use case priorization using criticality analysis. In: Lagana, A. *et al*. (Eds.), ICCSA 2004. LNCS 3046, Springer-Verlag, Berlin, Heidelberg, pp: 496-505.

Génova1, G., J. Llorens, P. Metz, R. Prieto-Díaz and H. Astudillo, 2005. Open issues in industrial use case modeling. In: Jardim Nunes, N. *et al*. (Eds.), UML 2004 Satellite Activities. LNCS 3297, Springer-Verlag, Berlin, Heidelberg, pp: 52-61.

Giakoumakis, E.A. and G. Xylomenos, 1996. Evaluation and selection criteria for software requirements specification standards. Software Eng. J., 11(5): 307-319.

Glinz, M., 2007. On non-functional requirements. Proceeding of 15th IEEE International Requirements Engineering Conference, pp: 21-26.

Goguen, J.A. and C. Linde, 1992. Techniques for requirements elicitation. Proceeding of IEEE International Symposium on Requirements Engineering, pp: 152-164.

Grau, G. and X. Franch, 2007. A goal-oriented approach for the generation and evaluation of alternative architectures. In: Oquendo, F. (Ed.), ECSA 2007. LNCS 4758, Springer-Verlag, Berlin, Heidelberg, pp: 139-155.

He, L., J.C. Carver and R.B. Vaughn, 2008. Using inspections to teach requirements validation. CrossTalk: J. Defense Software Eng., 21(1).

Heninger, K.L., 1980. Specifying software requirements for complex systems: New techniques and their application. IEEE T. Software Eng., SE-6(1): 2-13.

Herrmann, T. and A. Nolte, 2010. The integration of collaborative process modeling and electronic brainstorming in co-located meetings. In: Kolfchoten, G., T. Herrmann and S. Lukosch (Eds.), CRIWG 2010. LNCS 6257, Springer-Verlag, Berlin, Heidelberg, pp: 145-160.

Huysmans, J., K. Dejaeger, C. Mues, J. Vanthienen and B. Baesens, 2011. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. Decis. Support Syst., 51: 141-154.

IEEE Computer Society, 1998. Recommended Practice for Software Requirements Specifications. IEEE Standard 830-1998, pp: 1-40.

Ives, B., M.H. Olson and J.J. Baroudi, 1983. The measurement of user information satisfaction. Commun. ACM, 26(10): 785-793.

Jiang, L., 2005. A framework for the requirements engineering process development. Ph.D. Thesis, University of Calgary Calgary, Alberta.

John, S. and P. Gunnar, 2006. Defining misuse within the development process. IEEE Secur. Priv., 4(6).

Juristo, N., A. Moreno and A. Silva, 2002. Is the European industry moving toward solving requirements engineering problems? IEEE Software, 19(6): 70-77.

Jurjens, J., 2002. UMLsec: Extending UML for secure systems development. In: Jezequel, J.M., H. Haussmann and S. Cook (Eds.), Proceeding of the Unified Modeling Language, 5th International Conference. LNCS 2460, Springer, Dresden, Germany, pp: 412-425.

Kamata, M.I. and T. Tamai, 2007. How does requirements quality relate to project success or failure? Proceeding of 15th IEEE International Requirements Engineering Conference (RE'07), pp: 69-78.

Kandt, R.K., 2003. Software quality improvement software requirements engineering: Practices and techniques. SQI Report R-3.

Kesh, S., 1995. Evaluating the quality of entity relationship models. Inform. Software Tech., 37(12): 681-689.

Kohavi, R. and S. Daniel, 1998. Targeting business users with decision table classiers. Proceeding of KDD-98.

Kotonya, G. and I. Sommerville, 1998. Requirements engineering: Processes and techniques. John Wiley and Sons, United Kingdom, pp: 5-282.

Lai, X., M. Xie and K.C. Tan, 2004. Optimizing product design using the kano model and QFD. Proceeding of IEEE International Engineering Management Conference, pp: 1085-1089.

Laitenberger, O., 2002. A Survey of Software Inspection Technologies. In: Chang, S.K. (Ed.), Handbook on Software Engineering and Knowledge Engineering. World Scientific, Singapore, pp: 517-556.

Laitenberger, O. and J.M. DeBaud, 2000. An encompassing life cycle centric survey of software inspection. J. Syst. Software, 50(1): 5-31.

Lehtola, L., M. Kauppinen and S. Kujala, 2004. Requirements prioritization challenges in practice. In: Bomarius, F. and H. Iida (Eds.), PROFES 2004. LNCS 3009, Springer-Verlag, Berlin, Heidelberg, pp: 497-508.

Litchfield, R.C., 2008. Brainstorming rules as assigned goals: Does brainstorming really improve idea quantity? Motiv. Emotion, 33: 25-31.

Lloyd, W.J., M.B. Rosson and J.D. Arthur, 2002. Effectiveness of elicitation techniques in distributed requirements engineering. Proceeding of the IEEE Joint International Conference on Requirements Engineering, pp: 311-318.

Lutz, R.R., 1993. Analyzing software requirements errors in safety-critical embedded systems. Proceeding of IEEE International Symposium on Requirements Engineering, pp: 126-133.

Mohd Kasirun, Z. and S.S. Salim, 2008. Focus group discussion model for requirements elicitation activity. Proceeding of International Conference on Computer and Electrical Engineering, pp: 102-105.

Nurmuliani, N., D. Zowghi and S.P. Williams, 2004. Using card sorting technique to classify requirements change. Proceeding of the 12th IEEE International Requirements Engineering Conference, pp: 240-248.

Nuseibeh, B. and S. Easterbrook, 2000. Requirements engineering: A roadmap. Proceeding of the Conference on the Future of Software Engineering (ICSE'00), pp: 35-46.

Ochoa, S., R. Alarcon and L. Guerrero, 2009. Understanding the relationship between requirements and context elements in mobile collaboration. In: Jacko, J.A. (Ed.), Human-Computer Interaction, Part III, HCII 2009. LNCS 5612, Springer-Verlag, Berlin, Heidelberg, pp: 67-76.

Opdahl, A.L. and G. Sindre, 2009. Experimental comparison of attack trees and misuse cases for security threat identification. Inform. Software Tech., 51(5): 946-932.

Opdenakker, R., 2006. Advantages and disadvantages of four interview techniques in qualitative research. Qual. Soc. Res., 7(4), Art. 11.

Porter, A.A., L.G. Jr. Votta and V.R. Basili, 1995. Comparing detection methods for software requirements inspections: A replicated experiment. IEEE T. Software Eng., 21(6): 563-575.

Ragone, A., K. Mirylenka, F. Casati and M. Marchese, 2013. On peer review in computer science: Analysis of its effectiveness and suggestions for improvement. Scientometrics, 97: 317-356.

Roman, G., 1985. A taxonomy of current issues in requirements engineering. Computer, 18(4): 14-23.

Saqi, S.B. and S. Ahmed, 2008. Requirements Validation Techniques practiced in industry: Studies of six companies. M.A. Thesis, Software Engineering, School of Engineering, Blekinge Institute of Technology, Sweden.

Sauerwein, E., F. Bailom, K. Matzler and H.H. Hinterhuber, 1996. The Kano model: How to delight your customers. Proceeding of International Working Seminar on Production Economics, pp: 313-327.

Scheinholtz, L.A. and I. Wilmont, 2011. Interview patterns for requirements elicitation. In: Berry, D. and X. Franch (Eds.), REFSQ 2011. LNCS 6606, Springer-Verlag, Berlin, Heidelberg, pp: 72-77.

Sendall, S., 2003. Requirements elicitation with use cases. Lecture Notes Comput. Sc., 2604: 203.

Siau, K. and L. Lee, 2004. Are use case and class diagrams complementary in requirements analysis? An experimental study on use case and class diagrams in UML. Requir. Eng., 9: 229-237.

Sindre, G. and A.L. Opdahl, 2004. Eliciting security requirements with misuse cases. Requir. Eng., 10: 34-44.

Singer, M., 2013. Validation in reading comprehension. Curr. Dir. Psychol. Sci., 22: 361-366.

Sneed, H.M. and C. Verhoef, 2013. Natural language requirement specification for web service testing. Proceeding of 15th IEEE International Symposium on Web Systems Evolution (WSE, 2013), pp: 5-14.

Sommerville, I., 2010. Software Engineering. 9th Edn., Addison-Wesley, ISBN 10: 0-13-703515-2.

Sommerville, I. and P. Sawyer, 1997. Requirements Engineering: A Good Practice Guide. John Wiley & Sons, New York.

Song, Y., M. Evans and E.K. Park, 1995. A comparative analysis of entity-relationship diagrams1. J. Comput. Software Eng., 3(4): 427-459.

Spencer, D., 2009. Card Sorting: Designing Usable Categories. Brooklyn, New York, USA.

Stellman, A. and J. Greene, 2005. Applied Software Project Management. O'Reilly Media Inc., USA, pp: 113.

Subramanian, G.H., J. Nosek, S.P. Raghunathan and S.S. Kanitkar, 1992. Comparison of the decision table and tree. Commun. ACM, 35(1): 89-94.

Tablan, V., D. Damljanovic and K. Bontcheva, 2008. A natural language query interface to structured information. Proceeding of the 5th European Semantic Web Conference on the Semantic Web: Research and Applications (ESWC'08), pp: 361-375.

Tøndel, I.A., J. Jensen and L. Røstad, 2010. Combining misuse cases with attack trees and security activity models. Proceeding of 10th International Conference on Availability, Reliability and Security (ARES, 2010), pp: 438-4445.

Ullah, S., M. Iqbal and A.M. Khan, 2011. A survey on issues in non-functional requirements elicitation. Proceeding of International Conference on Computer Networks and Information Technology (ICCNIT, 2011), pp: 333-340.

Umber, A. and I.S. Bajwa, 2011. Minimizing ambiguity in natural language software requirements specification. Proceeding of 6th International Conference on Digital Information Management (ICDIM, 2011), pp: 102-107.

Van Lamsweerde, A., 2001. Goal-oriented requirements engineering: A guided tour. Proceeding of 5th IEEE International Symposium on Requirements Engineering, pp: 249-262.

Von Dran, G.M., P. Zhang and R. Small, 1999. Quality websites: An application of the kano model to website design. Proceeding of the 5th Americas Conference on Information Systems (AMCIS'99), August 13-15.

Whittle, J., D. Wijesekera and M. Hartong, 2008. Executable misuse cases for modeling security concerns. Proceeding of 30th ACM/IEEE International Conference on Software Engineering (ICSE '08).

Wieringa, R., N. Maiden, N. Mead and C. Rolland, 2005. Requirements engineering paper classification and evaluation criteria: A proposal and a discussion. Requir. Eng., 11: 102-107.

Wood, J.R. and L.E. Wood, 2008. Card sorting: Current practices and beyond. J. Usability Stud., 4(1): 1-6.

Xiong, W. and D. Litman, 2010. Identifying problem localization in peer-review feedback. In: Aleven, V., J. Kay and J. Mostow (Eds.), ITS 2010. Part II, LNCS 6095, Springer-Verlag, Berlin, Heidelberg, pp: 429-431.

Yeh, D., Y. Li and W. Chu, 2008. Extracting entity-relationship diagram from a table-based legacy database. J. Syst. Software, 81: 764-771.

Zave, P., 1997. Classification of research efforts in requirements engineering. ACM Comput. Surv., 29(4): 315-321.

Zowghi, D. and V. Gervasi, 2003. On the interplay between consistency, completeness, and correctness in requirements evolution, Inform. Software Tech., 45: 993-1009.