KNOWLEDGE-BASED AMBIGUITY DETECTION APPROACH TO ELIMINATE VAGUENESS IN USER REQUIREMENTS

JACLINE SUDAH ANAK SINPANG

UNIVERSITI TEKNOLOGI MALAYSIA

KNOWLEDGE-BASED AMBIGUITY DETECTION APPROACH TO ELIMINATE VAGUENESS IN USER REQUIREMENTS

JACLINE SUDAH ANAK SINPANG

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Master of Philosophy

School of Computing
Faculty of Engineering
Universiti Teknologi Malaysia

JANUARY 2021

ACKNOWLEDGEMENT

In preparing this thesis, I was in contact with many people, researchers, academicians, and practitioners. They have contributed towards my understanding and thoughts. In particular, I wish to express my sincere appreciation to my main thesis supervisor, Associate Professor Dr. Shahida Sulaiman, for encouragement, guidance and critics. I have learnt and gain knowledge not only for my research but also for my daily life through her guidance. I am also very thankful to my co-supervisors Dr. Norsham Idris for her guidance and advices. Without their continuous support and interest, this thesis would not have been the same as presented here.

I also want to give my gratitude to my family member especially my mom and dad who never stop believing in me. I am very grateful that throughout this journey they really show their support towards what I am pursuing. Not to forget, my friends who are in one way or another have helped me in this journey. Your support, kind words and endless encouragement will never be forgotten.

My fellow postgraduate students should also be recognised for their support specifically Malik Muhammad Ali Shahid and Norzila Ngadiman. My sincere appreciation also extends to all my colleagues and others who have provided assistance at various occasions. Their views and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space.

ABSTRACT

Requirements are the foundation of a software system. It is expected to be clear, precise, and non-ambiguous. Ambiguous requirements are the results of requirements that are gathered in natural language. Natural language is normally used while gathering requirements in verbal or non-verbal because it is easier for software engineers and stakeholders to understand each other. Vague requirements often stem from vague words in the requirements. A requirement that consists of a vague word depends on the individual interpretation and this will cause the requirements to be ambiguous. Vagueness is part of the ambiguity. This could lead to a wrong interpretation of what the system should be and should do. As requirements engineering is a crucial phase in software development, it is important to tackle the issue of vagueness to avoid the requirements to be ambiguous. This research proposes an approach known as Knowledge-based Requirements Analysis for Ambiguity Detection (KbReAD) that provides automatic detection of vague words in requirements using the rule-based reasoning technique which is a specific type of knowledge base. The knowledge base allows analysis of a large amount of data to be done in a small amount of time. It also does not depend on previous experience like how machine learning works. The knowledge base is also high in reliability. The initial expert knowledge for vagueness is captured using the rule-based technique into the KbReAD prototype tool that allows new knowledge to be added dynamically. From this knowledge, vagueness can be divided into six categories; vague subjects, vague adjectives, vague prepositions, vague verbs, vague phrases, and vague adverbs. Sets of raw requirements that are yet to be documented in Software Requirement Specification (SRS) are analyzed to evaluate the rule-based reasoning. The result from the analysis shows the proposed work is capable of predicting the actual number of vague requirements. The evaluation shows that the proposed approach is able to predict and detect vague words in the requirements accurately.

ABSTRAK

Keperluan ialah asas sistem perisian. Secara ideal, keperluan seharusnya jelas, tepat, dan tidak samar. Keperluan samar ialah hasil daripada keperluan yang dikumpulkan dalam penggunaan bahasa seharian. Bahasa seharian digunakan semasa pengumpulan keperluan sama ada secara lisan atau bukan lisan kerana ia lebih mudah untuk difahami. Keperluan yang samar sering terjadi daripada perkataan yang tidak jelas. Keperluan terdiri daripada perkataan yang tidak jelas bergantung kepada penafsiran individu dan ini akan menyebabkan berlakunya kekaburan. Ketidakpastian adalah sebahagian daripada kekaburan. Kesan daripada ini menyebabkan penafsiran yang salah mengenai apa yang seharusnya dilakukan oleh sistem. Oleh kerana kejuruteraan keperluan adalah fasa penting dalam perisian, masalah kekaburan perlu ditangani bagi mengelakkan keperluan menjadi samar. Penyelidikan ini mencadangkan pendekatan Analisis Keperluan Berasaskan Pengetahuan untuk Pengesanan Kekaburan (KbReAD). Pendekatan ini memberikan pengesanan automatik bagi kumpulan perkataan yang kabur dalam keperluan menggunakan teknik penaakulan berdasarkan peraturan yang merupakan sejenis teknik khusus bagi pangkalan pengetahuan. Pangkalan pengetahuan membolehkan banyak data dapat dianalisis dalam masa yang singkat. Ia tidak bergantung pada kaedah sebelumnya seperti pembelajaran mesin berfungsi. Pengetahuan awal ditambah menggunakan teknik berasaskan peraturan ke dalam alat prototaip KbReAD yang membolehkan pengetahuan baru ditambah secara dinamik. Dari pengetahuan ini, kekaburan dapat dibahagi kepada enam kategori; subjek yang samar, kata sifat yang tidak jelas, kata depan yang tidak jelas, kata kerja yang tidak jelas, frasa yang tidak jelas, dan kata keterangan yang tidak jelas. Set keperluan asal yang belum didokumentasikan dalam Spesifikasi Keperluan Perisian (SRS) dianalisis untuk menilai penaakulan berdasarkan peraturan. Hasil daripada analisis menunjukkan karya yang dicadang mampu meramalkan jumlah keperluan samar. Penilaian menunjukkan pendekatan yang dicadang dapat meramalkan dan mengesan perkataan yang kabur dalam keperluan dengan tepat.

TABLE OF CONTENTS

	TITLE	PAGE
	DECLARATION	**iii**
	DEDICATION	**iv**
	ACKNOWLEDGEMENT	**v**
	ABSTRACT	**vi**
	ABSTRAK	**vii**
	TABLE OF CONTENTS	**viii**
	LIST OF TABLES	**xi**
	LIST OF FIGURES	**xii**
	LIST OF ABBREVIATIONS	**xiv**
	LIST OF APPENDICES	**xv**
CHAPTER 1	**INTRODUCTION**	**1**
1.1	Overview	1
1.2	Problem Background	3
1.3	Problem Statement and Research Questions	5
1.4	Goals and Objectives	7
1.5	Scope	7
1.6	Significance of Study	8
1.7	Thesis Organization	9
CHAPTER 2	**LITERATURE REVIEW**	**11**
2.1	Introduction	11
2.2	Requirements	11
2.3	Requirements Engineering	13
2.4	Quality in Software Requirements	15
2.5	Problem in Requirements	17
2.6	Vagueness in Requirements	21
2.7	Knowledge-based System	22

	2.7.1 Knowledge-based System Architecture	26
	2.7.2 Rule-based Reasoning	27
2.8	Summary	29
CHAPTER 3	**RESEARCH METHODOLOGY**	**31**
3.1	Introduction	31
3.2	Research Procedure	31
	3.2.1 Literature Review	33
	3.2.2 Development of Proposed Work	33
	3.2.3 Implementation and Experiment	34
	3.2.3.1 Experimental Design and Hypothesis	35
	3.2.4 Result and Discussion	37
	3.2.5 Conclusion	37
3.3	Theoretical Framework	38
3.4	Data Definition	39
3.5	Technology Acceptance Model	41
3.6	Precision Recall Evaluation	41
3.7	Technology and Equipment Used	43
3.8	Summary	44
CHAPTER 4	**KNOWLEDGE-BASED REQUIREMENTS ANALYSIS FOR AMBIGUITY DETECTION (KbReAD)**	**45**
4.1	Introduction	45
4.2	The Knowledge Base	45
4.3	Algorithm of Proposed Work	47
4.4	Prototype Tool	49
	4.4.1 KbReAD Architecture	50
	4.4.2 Use Case for the Prototype	51
4.5	Summary	52
CHAPTER 5	**EVALUATION**	**53**
5.1	Introduction	53
5.2	The Experiment	53

5.2.1 Result and Discussion 55
5.3 Evaluation Strategy 57
5.4 Precision-Recall Evaluation and T-test 57
5.4.1 Discussion 61
5.4.2 Hypothesis Testing Using T-test 62
5.5 Usability Study 63
5.5.1 Analysis and Finding 65
5.6 Threat to Validity 70
5.7 Comparison of KbReAD with Existing Works 71
5.8 Summary 73

CHAPTER 6 CONCLUSION 75
6.1 Research Summary 75
6.2 Contribution 76
6.3 Future Work 76
6.4 Concluding Remark 77

REFERENCES 79

Appendix A 85
Appendix B 101
Appendix C 103

LIST OF TABLES

TABLE NO.	TITLE	PAGE
Table 2.1	Problem in requirements	18
Table 2.2	Existing solution in requirements engineering	20
Table 2.3	Knowledge-based implementation in requirements engineering	24
Table 3.1	Component of experiment	35
Table 3.2	RALIC project scope	40
Table 3.3	Excerpt of RALIC elicited requirements	40
Table 4.1	KbReAD initial knowledge	46
Table 4.2	KbReAD current knowledge	47
Table 5.1	Requirement that contains one type of vagueness	54
Table 5.2	Requirement that contains more than one type of vagueness	56
Table 5.3	Confusion matrix table	58
Table 5.4	Sample requirements	60
Table 5.5	Sections in questionnaire	64
Table 5.6	Perceived usefulness questions	64
Table 5.7	Perceived ease-of-use questions	65
Table 5.8	Perceived usefulness result	67
Table 5.9	Perceive ease-of-use result	68
Table 5.10	Comparison result	72

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
Figure 2.1	Example of functional requirements	13
Figure 2.2	Requirements engineering	14
Figure 2.3	Qualitative properties	16
Figure 2.4	Different techniques in AI	22
Figure 2.5	Classification of knowledge in RE	25
Figure 2.6	General KBS architecture	26
Figure 2.7	Elaborated KBS architecture	27
Figure 2.8	Example of production rule	28
Figure 3.1	Research procedure flowchart	32
Figure 3.2	Process of the experiment	36
Figure 3.3	Sample mean calculation	36
Figure 3.4	Sample standard deviation calculation	37
Figure 3.5	T-test calculation	37
Figure 3.6	Forward chaining	39
Figure 3.7	Precision recall metric	42
Figure 3.8	Precision formula	43
Figure 3.9	Recall formula	43
Figure 4.1	Algorithm of KbReAD	48
Figure 4.2	Knowledge base searching	49
Figure 4.3	Architecture of KbReAD	50
Figure 4.4	Use case diagram	51
Figure 5.1	Result of precision-recall KbReAD	59
Figure 5.2	T-test analysis	63
Figure 5.3	Respondents demographic	66

Figure 5.4	Perceived usefulness bar chart	67
Figure 5.5	Perceived ease-of use bar chart	69

LIST OF ABBREVIATIONS

AI	-	Artificial Intelligence
CI	-	Computational Intelligence
KbReAD	-	Knowledge-based Requirements Analysis for Ambiguity Detection
KBS	-	Knowledge-based System
NL	-	Natural Language
RALIC	-	Replacement Access, Library and ID Card
RE	-	Requirements Engineering
SDLC	-	Software Development Life Cycle
SRS	-	Software Requirements specification

LIST OF APPENDICES

APPENDIX	TITLE	PAGE
Appendix A	RALIC raw requirements	85
Appendix B	The knowledge-base	101
Appendix C	Technology Acceptance Survey	100

CHAPTER 1

INTRODUCTION

1.1 Overview

The usage of computers has grown exponentially over the past two decades, from simple to complex software. Thus, software engineers have been chosen to resolve progressively bigger and larger issues as well as savvy and productive ways. The developments and past encounters towards creating a good quality of software are partly responsible for the rise of the software engineering discipline (Mall, 2014). According to Sommerville (2011), software engineering is an engineering discipline that is interested in all the aspect of software production.

It is imperative to go through a series of formulaic steps to attain a high-quality result in delivering software for clients (Pressman, 2005); In software engineering, it is called the software process. Software process involves four basic activities that include (Sommerville, 2011) Specification, Development, Validation, and Evolution.

"Requirements engineering is concerned with discovering, eliciting, developing, analyzing, determining verification methods, validating, communicating, documenting, and managing requirements" (ISO/IEC/IEEE 29148:2011, 2011). Requirements engineering can be divided into six phases which include the feasibility study, acquisition of requirements, analyzing the requirements, categorizing the requirements, managing the requirements, and finally documenting the requirements (Sankhwar et al., 2014). According to the international standard of IEEE ISO/IEC/IEEE (2011), a requirement is a form of a report that transcribes a necessity and its related limitations and conditions.

A study by Johnson (2000) through CHAOS report demonstrates that half of the influences associated with project and product success are related to requirements. This means that when conducting requirements engineering activities, it is expected to produce error-free requirements. However, Pandey et al. (2010) reveal that it is difficult to maintain an accurate set of requirements in a large or complex system. In addition, Malik et al. (2013) state that requirements keep on changing throughout the development. Parra et al. (2015) indicate that the quality of the requirements could cause an error while developing the project.

Meziane and Vadera (2010) state that a number of researchers adopt Artificial Intelligence (AI) techniques in order to improve the software development activities and that there is huge potential in utilizing AI for supporting and upgrading software engineering. A few existing works have applied AI techniques at certain phases in requirements such as software reliability prediction using artificial techniques by Al Gargoor and Saleem (2013), Usability study for creativity assistance on the interface of an AI system by Botega et al. (2018) and application of machine learning techniques for statistical analysis of software reliability data sets by Shanti et al. (2018). Hence, they have successfully proved its significance in enhancing the software development activities.

AI as defined in the IEEE std 1232-1995 are (i) The investigation of planning computer system showing the attributes related with insight in human conduct including understanding language, learning, thinking from fragmented or dubious data, and taking care of issues, (ii) The order for creating computer system able to do breezing through the Turing Test in which the conduct of the computer system is no different from human conduct and (iii) The investigation of critical thinking utilizing computational models. By referring to the first definition of AI, this study aims to provide an approach that can perform an intelligent activity to solve the problem in requirements analysis as AI has shown a great potential to improve software engineering process.

1.2 Problem Background

Requirements engineering is an essential action, which can influence the whole action of software development project (Pandey et al., 2010). Requirements lay out the foundation to refine and elaborate the technical work plan (Schmidt, 2016). The development of any software task must be founded on high-quality requirements designing procedure (Parra et al., 2015). Thus, requirements need to be free from error and ambiguity.

Problems in requirements analysis are complicated as it needs to cope with the stakeholders needs and, at the same time, it needs to cope with the designers need (Macaulay, 2012). If new insights are added to the project, there is a high probability that the requirements need to be revised or changed accordingly (Vara et al., 2013).

There are several issues associated with requirements engineering phase throughout the process of software development. The issues can be summarized as shown below (Meziane and Vadera, 2010):

(a) Requirements are ambiguous: This happens due to the variety of the stakeholders' background and Natural Language (NL) which the first means to record the requirements. Characteristically questionable and adds to the deficiency of requirements the same number of suspicions made on a few issues

(b) Requirements are incomplete, unclear and unspecific: Some requirements are unclear and hard to validate

(c) Requirements are contradicting: Struggle in requirements engineering happen when two unique requirements vie for the same resources or when the fulfillment of one requirement blocks that of another

(d) Requirements are unstable: User needs develop after some time. It is not unordinary that amid the time it takes to build up a system, user requirements have effectively changed

(e) There are correspondence issues between stakeholders: Correspondence with and seeing every one of the stakeholders is a very troublesome and challenging task

(f) Requirements are hard to oversee: Traceability is the procedure of taking after requirements from its elicitation to usage and verification and validation. Connecting the diverse periods of requirements acceptance is regularly overlooked. Other administration issues identified with programming administration are task administration, programming expense, advancement time, assets administration and dealing with the evolving environment

There are six types of ambiguity as stated by Massey et al. (2014) and this study will only focus on vagueness as this type of ambiguity is often overlooked. On numerous occasions, researcher focuses on the most common ambiguity types. Umber and Bajwa (2011) focused on semantic and syntactic ambiguity, Dalpiaz et al. (2019) aimed at semantic ambiguity while Bussel (2009), study on lexical, semantic and syntactic ambiguity. One vague word in a requirement could cause the requirement to be ambiguous. Vagueness is a statement that admits borderline cases or relative interpretation (Massey et al., 2014). Firesmith (2003) indicates five types of vagueness that occur in requirements. First, vague subjects where the subject in the requirements can refer to multiple things. Second, vague adjectives. The adjectives may mean different things to the different readers. Third, vague preposition where user may have use vague nouns. Fourth, vague verbs which are more qualitative than quantitative. Lastly, subjective phrases where the meaning of the phrases depend on the understanding an individual. This study will look into depth on the issue of ambiguous requirements and implement AI technique to tackle this particular issue to reduce the vagueness in requirements.

1.3 Problem Statement and Research Questions

Gervasi and Zowghi (2010) mentioned that ambiguity has always been a challenge in requirements specification because misunderstanding or misinterpretation of the requirements will be carried to next stages of development. Ambiguity in requirements analysis can lead to several problems that influence the final product of the system (Nigam et al., 2012). Imprecise requirements or ambiguous requirements occur when software engineers collect the requirements from stakeholders and they unconsciously disambiguate the requirements according to their own understanding without really knowing that the requirements are actually ambiguous (Wang et al., 2013). Based on the study conducted by Rani and Aggarwal (2020), ambiguities are one of the major problems in SRS.

Vagueness in requirements could create a confusing meaning to the receiving end. How the requirements engineer or software engineers interpret the requirements could be different from what the requirements intended to deliver if the requirements contain a vague word. When there is a vague word in a requirement, the interpretation of the requirement would solely depend on how the requirements engineer or the software engineer understand the requirements. Thus, it is better if requirements can be free from any vague words so that the requirements will be interpreted as how it is intended to.

To steer clear from vagueness and ambiguity, Gupta and Deraman (2019) propose a framework to assist and support requirement engineer. The framework helps requirements engineer to choose the right elicitation technique in order to write requirements that are not vague and unambiguous. Sadiq (2017) proposed a fuzzy based approach to deal with vagueness and impreciseness in goal-oriented requirements elicitation process. The fuzzy based approach aids in prioritizing software requirements from the stakeholders.

A deeper analysis is conducted to better understand vagueness in requirements analysis. Problems that are caused by ambiguity in requirements are also investigated and a thorough study on knowledge-based system is done to decide on the technique to be used in this study.

Thus, the main research question for this study is "Can AI technique detect ambiguity in requirements?" and the sub-questions for this study are:

(a) What are the issues related to vagueness in user requirements and the existing work that has been done to solve the issues?

(b) What are the attributes of high-quality requirements?

(c) What is the suitable KBS to improve the requirements analysis phase, and how can AI techniques KBS enhance the requirements analysis phase?

1.4 Goals and Objectives

The main goal of this study is to develop an approach that can be used to tackle the issue of ambiguity in requirements especially vagueness in requirements which causing the requirement to be ambiguous.

The objectives of this study are:

(a) To develop algorithm to detect ambiguities and vagueness using the proposed approach

(b) To demonstrate the applicability of the proposed approach through the development of its supporting prototype tool

(c) To measure the effectiveness of the proposed approach in requirements analysis

1.5 Scope

This study focuses on the conventional way of requirements engineering model that uses incremental Software Development Life Cycle (SDLC). Requirement engineering that implemented the iterative SDLC such as agile requirement engineering is only studied to look into the issue of requirements engineering as a whole.

The scopes of this study are:

(a) Requirements engineering phase specifically the analysis phase

(b) KBS techniques or approaches that show potential in improving requirements analysis

(c) Raw requirements that have not been documented into Software Requirements Specification (SRS)

(d) Functional requirements are the focus to verify the technique

(e) Ambiguity which is limited to vagueness

1.6 Significance of Study

Berry and Kamties (2004) mention in their study that unidentified ambiguity is believed to be one of the most major factors for failure in requirements analysis. Firesmiths (2003) state that numerous requirements are not testable. Ambiguous requirements are one of the requirements that cannot be tested. An ambiguous requirement could cause misunderstanding during implementation phase (Genova et al., 2013). Apart from that, ambiguous requirements can create a totally opposite meaning for the same request to the very same or dissimilar stakeholders (Rani and Aggarwal, 2020). Therefore, it is important to eliminate the issue of ambiguity in requirements to ensure that the system built is according to the requirements given by users.

As there are several types of ambiguity, and this study will be focusing on vagueness. A vague word in a requirement could cause the requirement to be a vague requirement. Thus, studying in depth on the issue of vagueness and providing a technique to avoid such requirements to happen is important to ensure the success of the system or the project.

Meziane and Vadera (2010) state that there is enormous potential in using AI for supporting and overhauling software engineering. Hence, it is essential to do this study to produce a systematic literature review and to identify the AI technique that can help to improve requirement engineering phase and to measure the effectiveness of proposed AI technique in requirements engineering.

1.7 Thesis Organization

Chapter 1 provides an overview of the study. The background of the problem that is studied is provided in this chapter. From the background of the problem, a problem statement is issued to find the research gap. Once the research gap is identified, research questions are formulated to help to narrow down the study. The goals and objectives of the study are established from the research questions. The scope of the study is identified so that it will not deviate from the goals and objectives of the study.

Chapter 2 of this study explains more on requirements engineering and problems in requirements engineering. Techniques and approaches of AI in requirements engineering are analyzed in this chapter. The knowledge-based approach is introduced in this chapter. This chapter also presents the systematic literature review on requirements engineering.

Chapter 3 elaborates on the research methodology applied for the study. A theoretical framework of the study is presented in this chapter. Data that was used to verify and evaluate the proposed approach is defined in this chapter.

Chapter 4 aims to explain the proposed approach. The design of the approach is elaborated in this chapter. A detailed explanation of the proposed approach is presented here. The evaluation of this study will be discussed in chapter 5. The result will be presented according to the objectives of this study. Chapter 6 will conclude the whole study. A summary of the whole study will be explained in this chapter.

REFERENCES

Agarwal, Udit. (2012) *Software Engineering*. S.K.Kataria & Sons

Arora, C., Sabetzadeh, M., Briand, L., and Zimmer, F. (2015) Automated checking of conformance to requirements templates using natural language processing. *IEEE transactions on Software Engineering*, pp. 944-968.

Abke, J., Gold, C., Kälberer, N., and Kuhn, M. (2014) Collaborative knowledge transfer via Wiki: A project based learning approach in software engineering. *In Interactive Collaborative Learning (ICL),* 2014 International Conference on pp. 283-288.

Bassiliades, N., and Vlahavas, I. (2000) Active knowledge-based systems. *In Knowledge-Based Systems* pp. 1-36.

Batanov, D. (2010) Knowledge-Based Support for Software Engineering. *In IFIP International Conference on Artificial Intelligence Applications and Innovations* pp. 219-229.

Berry, D. M., and Kamsties, E. (2004) Ambiguity in requirements specification. I*n Perspectives on software requirements* pp. 7-44.

Gosnell, M., Woodley, R., Hicks, J. and Cudney, E. (2014) 'Exploring the Mahalanobis-Taguchi Approach to Extract Vehicle Prognostics and Diagnostics', in *Computational Intelligence in Vehicles and Transportation Systems (CIVTS), 2014 IEEE Symposium on*, pp. 84–91.

Bhardwaj, S., and Goyal, A. K. (2014) A Comparative Analysis of Agent Oriented Requirement Engineering Frameworks. *International Journal of Computer Applications*, pp. 87-94 .

Botega, L. F., da Silva, J. C., & Murphy, G. R. (2018) Usability study on the interface of an artificial inteligence system for creativity support. *Human Factors in Design*, 7(13), 041-060.

Bussel, D. (2009) D*etecting ambiguity in requirements specifications*. Phd Thesis. Tilburg University, Netherlands.

Cauvin, S. R., Sleeman, D., and Vasconcelos, W. W. (2015) Towards knowledge-intensive software engineering. *In Software Technologies (ICSOFT), 2015 10th International Joint Conference* on Vol. 1, pp. 1-8. IEEE.

Cheng, P. H., Chang, H. C., and Chang, F. H. (2012) Another extensible requirements elicitation and analysis method. *In Computational Intelligence, Communication Systems and Networks (CICSyN)* pp. 161-166.

Cruz, B. D., Jayaraman, B., Dwarakanath, A., and McMillan, C. (2017) Detecting Vague Words & Phrases in Requirements Documents in a Multilingual Environment. *In Requirements Engineering Conference (RE)* pp. 233-242.

De Souza, R., and Ying, Z. Z. (2000) Intensive Knowledge-Based Enterprise Modelling. *In Knowledge-Based Systems* pp. 103-123.

Ferrari, A., Spoletini, P., & Gnesi, S. (2016) Ambiguity cues in requirements elicitation interviews. *In Requirements Engineering Conference (RE)*, 2016 IEEE 24th International pp. 56-6).

Firesmith, D. (2003) Specifying good requirements. *Journal of Object Technology*, 2(4), pp. 77-87.

Garg, N., Agarwal, P., & Khan, S. (2015) Recent advancements in requirement elicitation and prioritization techniques. In *Computer Engineering and Applications (ICACEA), 2015 International Conference on Advances* in pp. 237-240.

Génova, G., Fuentes, J. M., Llorens, J., Hurtado, O., and Moreno, V. (2013) A framework to measure and improve the quality of textual requirements. *Requirements engineering*, 18(1), pp. 25-41.

Gervasi, V., and Zowghi, D. (2010) On the role of ambiguity in RE. In International Working Conference on *Requirements Engineering: Foundation for Software Quality* pp. 248-254.

Hopgood, A. A. (2012). Intelligent systems for engineers and scientists. *CRC Press*.

ISO/IEC/IEEE International Standard - Systems and software engineering(2011) -- Life cycle processes --Requirements engineering," in ISO/IEC/IEEE 29148:2011(E), vol., no., pp.1-94, Dec. 1, 2011, doi: 10.1109/IEEESTD.2011.6146379

IEEE ISO, (1995). Quality Management and Quality Assurance. 8402:1995.

Johnson, J. (2000). The chaos report. West Yarmouth, MA: The Standish Group.

Kirikova, M., and Grundspenkis, J. (2000) Using knowledge distribution in requirements engineering. In *Knowledge-Based Systems* pp. 149-184.

Kumar, M., Shukla, M., and Agarwal, S. (2013) A hybrid approach of requirement engineering in agile software development. In *Machine Intelligence and Research Advancement (ICMIRA), 2013 International Conference* on pp. 515-519.

Kumari, A. C., Srinivas, K., and Gupta, M. P. (2012) Software requirements selection using quantum-inspired elitist multi-objective evolutionary algorithm. *In Advances in Engineering, Science and Management (ICAESM)*, 2012 International Conference on pp. 782-787.

Macaulay, L. A. (2012) Requirements engineering. London: Springer Science & Business Media.

Malik, M. U., Chaudhry, N. M., and Malik, K. S. (2013) Evaluation of efficient requirement engineering techniques in *agile software development. International Journal of Computer Applications*, pp. 83-85.

Massey, A. K., Rutledge, R. L., Antón , A. I., and Swire, P. P. (2014) Identifying and classifying ambiguity for regulatory requirements. In *Requirements Engineering Conference (RE), 2014 IEEE 22nd International* pp. 83-92.

Meziane, F., and Vadera, S. (2010). Artificial Intelligence Applications for Improved Software Engineering Development: New Prospects. Information science reference.

Nigam, A., Arya, N., Nigam, B., and Jain, D. (2012) Tool for automatic discovery of ambiguity in requirements. *International Journal of Computer Science Issues (IJCSI)*, pp. 350-355.

Nguyen, T. H., Vo, B. Q., Lumpe, M., and Grundy, J. (2012) REInDetector: a framework for knowledge-based requirements engineering. In *Proceedings of the 27th IEEE/ACM international conference on automated software engineering* pp. 386-389.

Pandey, D., Suman, U., & Ramani, A. K. (2010) An effective requirement engineering process model for software development and requirements management. *In Advances in recent technologies in communication and computing (artcom), 2010 international conference* on pp. 287-291.

Parra, E., Dimou, C., Llorens, J., Moreno, V., & Fraga, A. (2015) A methodology for the classification of quality of requirements using machine learning techniques. *Information and Software Technology*, 67, pp. 180-195.

Perini, A., Susi, A., & Avesani, P. (2013) A machine learning approach to software requirements prioritization. I*EEE Transactions on Software Engineering,* 39(4), pp. 445-461.

Poscher, R. (2010) Ambiguity and vagueness in legal interpretation. *Revista de Estudos Constitucionais, Hermenêutica e Teoria do Direito*, pp. 272-280.

Prasetyo, N. A., and Bandung, Y. (2015) A design of software requirement engineering framework based on knowledge management and Service- Oriented Architecture Decision (SOAD) modeling framework. *In Information Technology Systems and Innovation (ICITSI), 2015 International Conference* on pp. 1-6.

Pressman, R. S. (2005) Software engineering: a practitioner's approach. Palgrave Macmillan.

Rehman, T., Khan, M. N. A., and Riaz, N. (2013) Analysis of requirement engineering processes, tools/techniques and methodologies. *International Journal of Information Technology and Computer Science (IJITCS),* 5(3), pp. 40.

Runde, S., Fay, A., and Wutzke, W. O. (2009) Knowledge-based Requirement-Engineering of building automation systems by means of Semantic Web technologies. *In Industrial Informatics, 2009. INDIN 2009. 7th IEEE International Conference* on pp. 267-272.

Russell, S. J., and Norvig, P. (2002) *Artificial intelligence: a modern approach*
(International Edition): Pearson Education Limited.

Schmidt, C (2016) Agile software development. *In Agile Software Development Teams* pp. 7-35.

Shankhwar, s., Singh, V., and Panday, D. (2014) Requirements Engineering Paradigm. *Global Journal of multidisciplinary studies,* (Volume 3), pp. 1-8.

Sommerville, I (2011) Software Engineering. Boston, Massachusetts: Pearson Education.

Sher, F., Jawawi, D. N., Mohamad, R., and Babar, M. I. (2014) Requirements prioritization techniques and different aspects for prioritization a systematic literature review protocol. I*n Software Engineering Conference (MySEC), 2014 8th Malaysian* pp. 31-36.

Siddique, A. B., Qadri, S., Hussain, S., Ahmad, S., Maqbool, I., & Khan, A. K. N. (2014) Integration of requirements engineering with UML in *software engineering practices. Sci. Int.(Lahore),* 26(5), pp. 2157-2162.

Sim, W. W., and Brouse, P. S. (2014) Empowering requirements engineering activities with personas. *Procedia Computer Science*, 28, pp. 237-246.

Sim, W. W., and Brouse, P. (2015) Developing ontologies and persona to support and enhance requirements engineering activities–a case study. *Procedia computer science*, 44, pp. 275-284.

Soo Ling Lim (2010) Social Networks and Collaborative Filtering in Large-Scale Requirements Elicitation. PhD Thesis. School of Computer Science and Engineering, University of New South Wales, Sydney, Australia.

Umber, A., and Bajwa, I. S. (2011) Minimizing ambiguity in natural language software requirements specification. *In Digital Information Management (ICDIM), 2011 Sixth International Conference* on pp. 102-107.

Vara, J. L., Falessi, D., and Verhulst, E. (2013) Specifying a framework for evaluating requirements Engineering Technology: Challenges and lessons learned. *In Empirical Requirements Engineering (EmpiRE), 2013 IEEE Third International Workshop* on pp. 1-8.

Van Lamsweerde, A. (2009) *Requirements engineering: From system goals to UML models to software* (Vol. 10). Chichester, UK: John Wiley & Sons.

Wang, Y., Gutiérrez, I. L. M., Winbladh, K., and Fang, H. (2013) Automatic detection of ambiguous terminology for software requirements. *In International Conference on Application of Natural Language to Information Systems* pp. 25-37.

Wiegers, K., and Beatty, J. (2013) *Software requirements.* Pearson Education.

Würfel, D., Lutz, R., and Diehl, S. (2016) Grounded requirements engineering: An approach to use case driven requirements engineering. *Journal of Systems and Software*, 117, pp. 645-657.

Yang, J. D., and Lee-Kwang, H. (2000) Treating Uncertain Knowledge-Based Databases. In *Knowledge-Based Systems* pp. 327-351.

Ying-ying, Y., Zong-yong, L., and Zhi-xue, W. (2008) Domain knowledge consistency checking for ontology-based requirement engineering. *In Computer Science and Software Engineering, 2008 International Conference* on Vol. 2, pp. 302-305.