

Requirement Engineering Assistance Diagnostic (READ)

Airin Ahia-Tabibi

Department of Software Engineering
Concordia University

Abstract

This report describes the *Requirement Engineering Assistance Diagnostic* project, READ, along with my responsibilities during my first work term, supervised by Prof. Olga Ormandjieva at Department of Computer Science and Software Engineering, Concordia University, May to August 2008.

The ultimate goal of this research is to develop a Web-based tool which will guide the developers in their in-depth study of the quality of requirements documents and in the timely identification of risks that poorly specified requirements might introduce into the project. The READ project's aim is to automatically generate a graphical representation of a requirements text in the form of a UML model and its XML representation. This is accomplished by applying Natural Language Processing (NLP) techniques within the context of a Requirements Engineering (RE) process.

As an undergraduate student involved in the project, I was mainly in charge of providing the proper documentations expressing different parts of the project. Therefore, I studied the project in-depth, gained a detailed insight to the project which was an asset for me to improve my research capabilities. I was also responsible for implementing the rules together with developing some tools for automatic processing.

Acknowledgment

First, I would like to gratefully express my gratitude to my project supervisor, Prof. Olga Ormandjieva, for her support and patience, for providing enthusiasm for research and for her continuous supervision in preceding this project.

I would like to warmly thank my coordinator, Mr. Jean Michel Paquette, for assisting me with his comments and suggestions that allowed me to improve the quality of this report.

I am particularly appreciative of the help I received from colleague and my good friend, Shadi Moradi-Seresht, who helped me with her knowledge in understanding the concept.

Last but not least, I am truly indebted to my common in law, Babak Khosravifar who has always giving me the most needed support and encouragement.

1. Introduction

This report is completed as a partial requirement of the CDMP award. The report deals with the description of the *Requirement Engineering Assistance Diagnostic project* (READ), supervised by Prof. Olga Ormandjieva along with my responsibilities, at Concordia University from May 1, 2008 to August 31, 2008.

Before starting my job I had been selected as a recipient of the *Natural Science and Engineering Research Council* (NSERC), and *Canadian Distributed Mentor Project* (CDMP), awards to be involved in a research project with a female professor. Thus, Prof. Olga Ormandjieva accepted having me in her research group and supervising me for the research and the work term.

When I began my work, I was really concerned about my qualification as a second year undergraduate student joining a research group, doing real programming, writing technical documentation and in general, being involved in a serious Software Engineering project. Soon enough, however, I was given many opportunities to express myself, learn about how to do research, work in an academic environment and become more self-confident.

2. Project Description

2.1. Purposes

The application of a systematic approach to the development of software is defined as Software Engineering. It needs techniques and procedures, the so-called software development process, with the purpose of improving the reliability and maintainability of software systems. The discipline of software engineering includes knowledge, tools and methods for measurement, design, development, construction, and testing. Requirement Engineering (RE), is a sub-discipline within Software Engineering which addresses specific challenges that exist in understanding the nature of the problem or, in other words, stakeholders' needs and wishes (illustrated in Figure 1).

Analyzing, understanding, classifying, and modeling user requirements are the key activities in the RE. The success of a software project is highly dependent upon thoroughly understanding the stakeholder's needs which are revealed and identified in the software requirement text [1] [2]. The research is motivated by the industrial importance of correctly documenting requirements and of reducing the costs inherent in identifying underlying requirements problems and potential system solutions. The ultimate goal of this research is to develop a web-based tool which will guide the developers in their in-depth study of the quality of requirements documents and in the timely identification of risks that poorly specified requirements might introduce into the project.

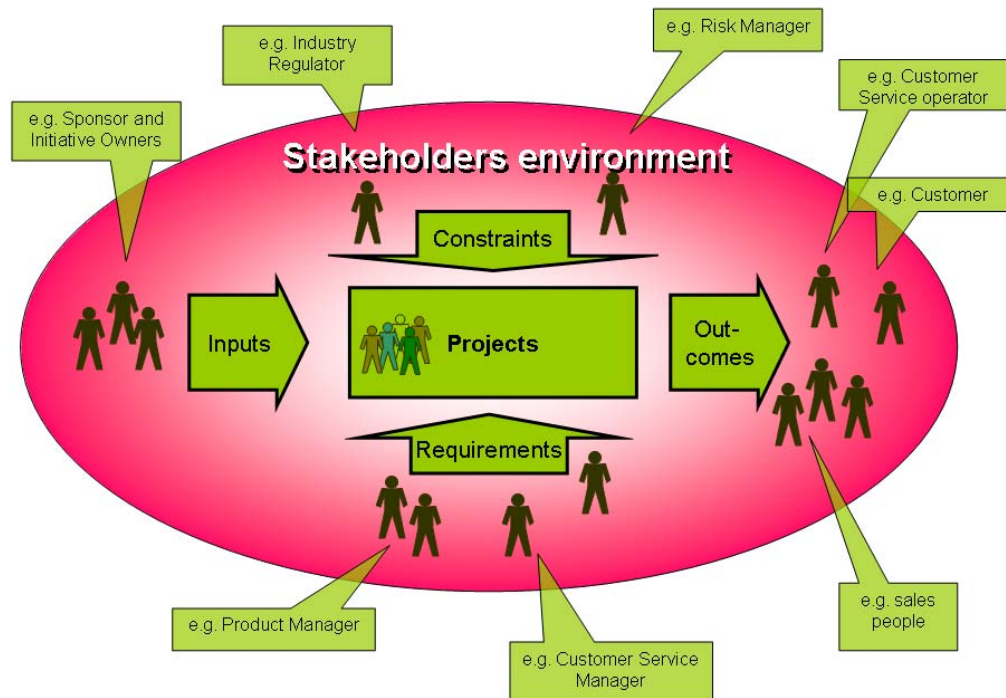


Figure 1 General Overview of the Projects Environment.

2.2. Scope

A system model can be acquired through an automated process, with a corpus of technical free text requirement documents in a form of Natural Language as input and a UML model, expressed graphically, through UML diagrams, representing its static and dynamic aspects, and textually, in an equivalent XML presentation, as output. The static and dynamic models are generated with the help of a set of pre-defined rules. In others words, the READ project's aim is to automatically generate a graphical representation of a requirements text in the form of a UML model and its XML representation. This is accomplished by applying Natural Language Processing (NLP) techniques within the context of a RE process [3].

2.3. Overall Description

Figure 2 illustrates the NLP-Based Quality Assessment in RE. This figure clearly summarizes the whole process, from getting the requirement text from the client to

generating a graphical representation. This project contains two main segments, NLP as the first part and Visualization of the Requirement Text as the second part.

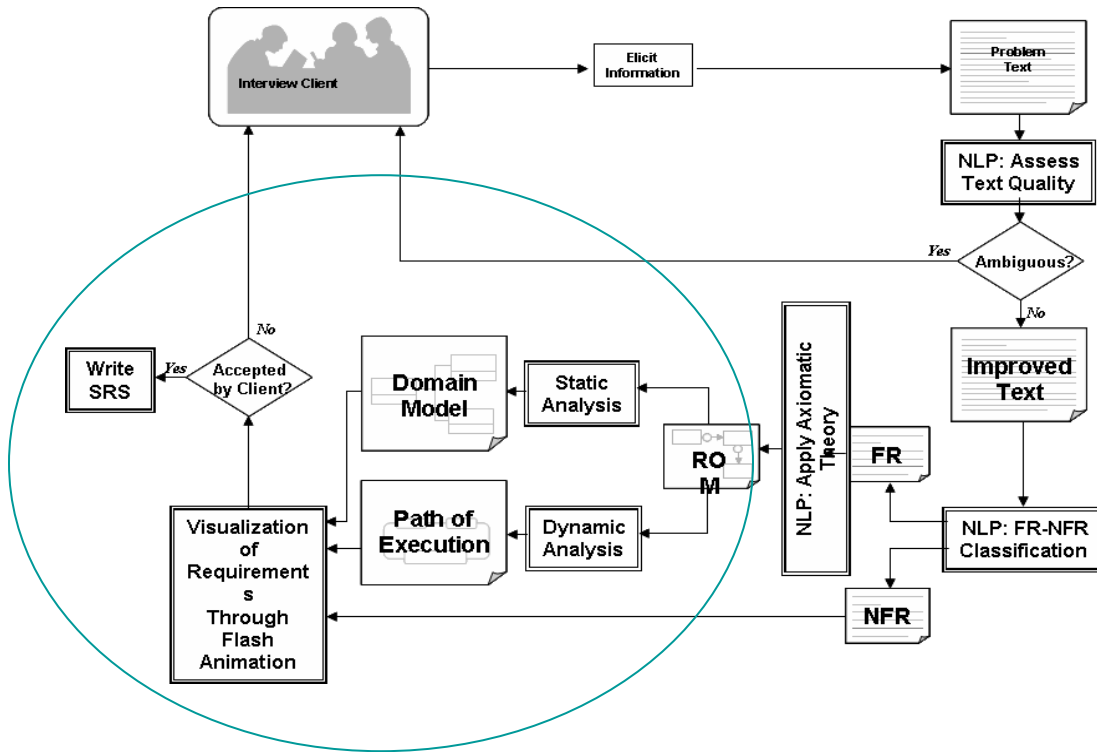


Figure 2. NLP-Based Quality Assessment in Requirements Engineering

First, the client input the requirement text as paragraphs in the form of natural language to the system. Then the paragraphs are divided to sentences in order to be checked by NLP methods [4] [5]. Doing this would find the ambiguities of the text and report them back to the client to improve the text and change the suggested modifications. This step continues to the point that there is no ambiguity from the point of view of the system. For instance, consider the following sentences:

1. We need a very good system.
2. The system must work very efficiently.

From NLP point of view, these two sentences are both ambiguous since it is not clear for the system what good and efficient mean exactly. Usually when a sentence contains an adjective or adverb, it is most probably ambiguous and must be explicitly stated. Therefore, this will be report back to the client for farther modifications. Another example is that the user may use a noun such as “ATM” and its pronoun “It” interchangeably, the so-called co-referencing, which is not acceptable for the system and would also be report back to the client.

1. Customer shall request the CBMSys to place an order.
2. CBMSys retrieves customer's credit record from the Customer Persistent Storage (CPS).
3. If the customer's credit record is good then the CBMSys places the order.
4. CBMSys creates and sends the purchase order to the publisher.
5. The CBMSys receives invoice from the publisher.
6. If the purchase invoice's details are correct then the CBMSys accepts and sends it to the Account Payable System.
7. The CBMSys prepares publisher's payment and sends the check to the publisher.
8. The CBMSys assigns shipment to the orders.
9. CBMSys generates sale invoice for customer.
10. The customer provides payment information to the CBMSys and the CBMSys sends it to the Account Receivable System.
11. Customer shall view the order status from the CBMSys.
12. Customer enters order Id to the CBMSys and it shows the order status.
13. Customer and publisher shall be able to update their personal profile from the CBMSys.

Figure 3. Improved Requirement Text by NLP

Figure 3 provide a sample improves requirement text by NLP. Once the improved version of the text is ready, it would be passed to the second part of the system which is circled in Figure 2. In this part, static and dynamic analysis takes place by using some predefined rules some of which are shown as examples in Table 1. These rules are implemented by using Object-Oriented methodologies and Java programming language in Eclipse plug-in platform. The Eclipse Visualization Plug-in (EVP), would provide the user with the graphical representation of the requirements text in the form of a UML model and its XML representation. Figure 4 exposes a screenshot of the output form on of the graphical representations. One of the important aspects of EVP which is worth mentioning is that the client is able to modify the out put as desire in the diagrams [5]. Should the graphical representation is acceptable by the client, will be used for farther software engineering development.

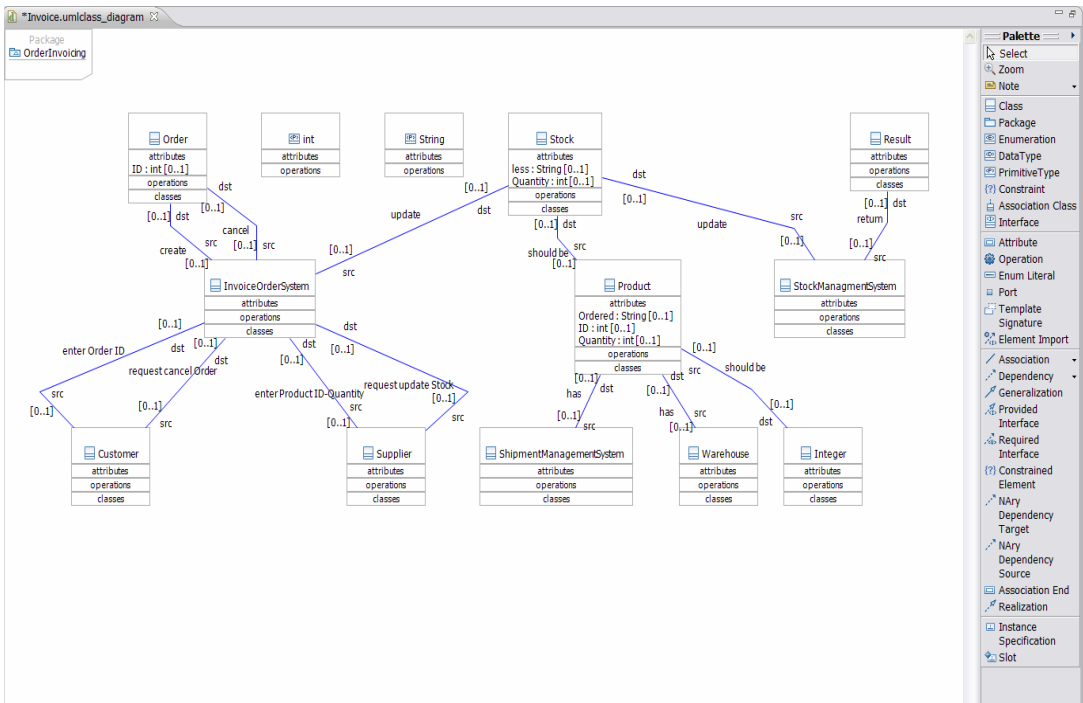


Figure 4. Graphical Representation Screenshot [5]



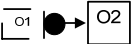


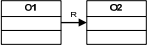
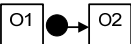


Formal presentation	Equivalent Structural View	Rule description
		<p>1. Each solid box can be considered either as a concept or an attribute for another concept in the structural view. However, not all boxes are valid concepts in the domain so later they will be refined according to the defined rules.</p>
		<p>2. If O1 is an adverb and an auxiliary verb, it will be ignored but if it is an adjective, it can be a candidate as an attribute for O2 or imply that there is an attribute for O2 called O'1. In first-cut view all adjectives are put as an attributes and later they can be substituted by their appropriate attributes in consultation with Wordnet manually.</p>
		<p>3. If R=transitive verb, a verbal (e.g. conform to) and both boxes have solid lines: It will be mapped directly to the static view. However, not all boxes are valid concepts in the domain so later they should be refined according to the defined rules.</p>
	 	<p>4. O1=noun and O2=noun : O2 can be considered as an attribute for O1 if O2 is found in the Predefined list of keywords otherwise it can imply the “has” relationship between the O1 and O2</p> <p>5. O1=noun and O2=noun: O1O2 can be considered as one concept if they always appear next to each other in the text or if none of them exists alone in the text.</p>

Table 1. Sample Rules and their Descriptions [1] [2]

3. Future Work

The READ project is a very large project modulated to different segments each of which requires lots of research attempt, development, and implementation. As we continue with the development of the visualization functionality of the project, we are aiming to improve the system and possibly add more functionality to the software.

The future work will focus on extending the predefined rules for identifying concepts and relations in structural view and implementing them in EVP. We aim to enhance the NLP part to be able to accept co-referencing and wider type of sentences as an input. Consider the following sentences:

1. I give you the book.
2. You get the book form me.

These two sentences both mean the same and they are clear for the client; however, they are not yet acceptable by the system. They will be taking care of in future. User Interface design, in addition, will be our next concern to provide a more improved output and more user friendly environment for the client.

4. Responsibility and Learning

I studied the project in-depth, attended project related presentations, learned different programming languages such as Java and XML as well as software engineering design and development methods. For instance, I indeed learned about drawing use case diagram, writing scenario, doing static and dynamic analyses which all will be part of my future courses in the program. I also became familiar with a new software platform, Eclipse, and gained a detailed insight to the project which was an asset for me to improve my research capability. In fact, gaining insight about research was the main reason for being selected as a recipient for NSERC and CDMP research awards; therefore, it counts as part of my responsibility.

Although I prepared some flowcharts for a program in the NLP part and studied that part as a section of the project, I was not so much involved in it. The specific aspect of my job as a Software Engineering student involved in the project was preparing proper software documentations expressing different parts of the project. Software Requirement Specification (SRS), for instance, was one of the documents I prepared for the project. Figure 5 illustrates small part of this document [3]. In addition, being involved in the implementation of rules and developing tools for the automatic process was also part of my job description.

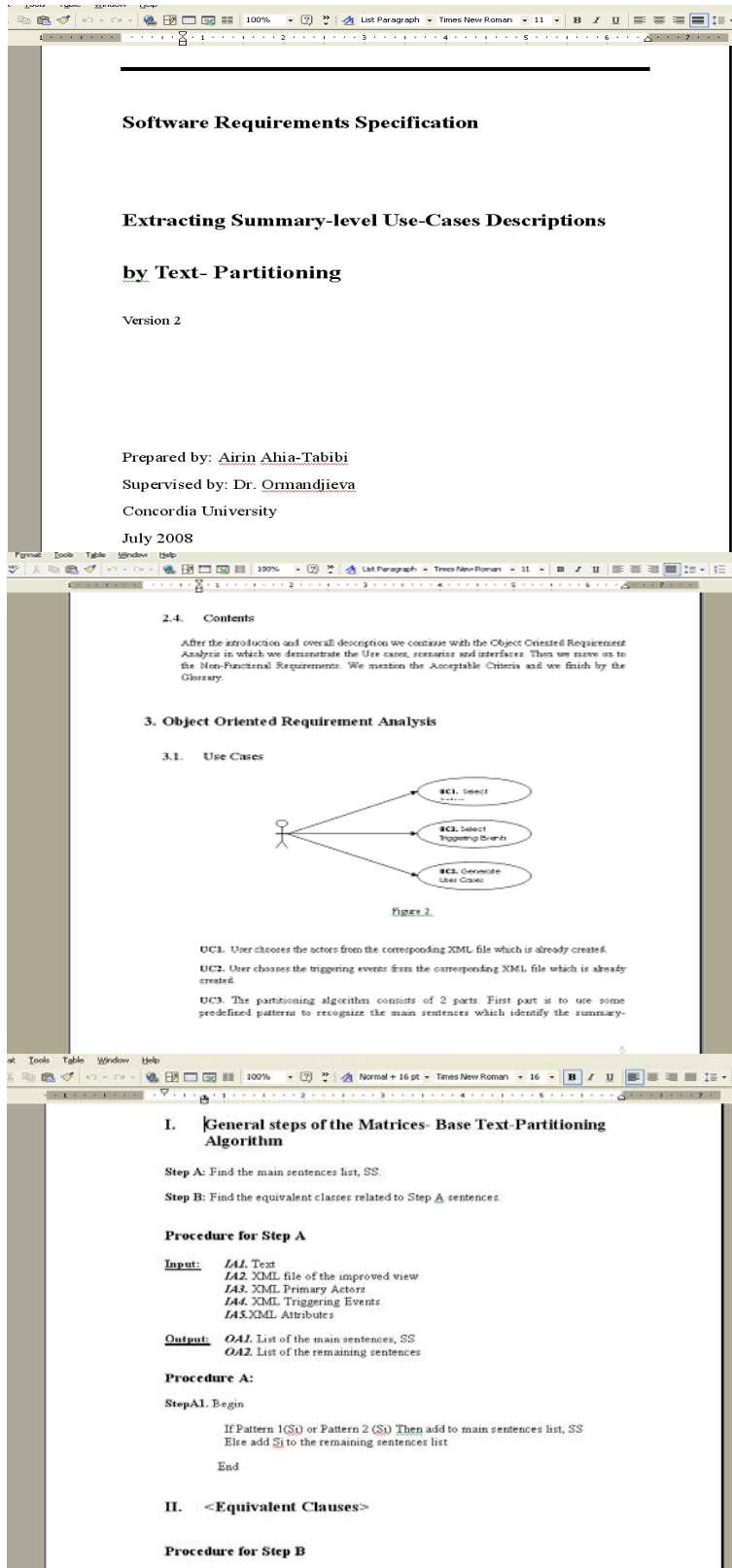


Figure 5. Prepared Software Requirement Document Screenshots [2] [3]

3. Conclusion

My primary aim was to gain a detailed knowledge to the project which is a background for future research projects. This is absolutely necessary for continuing my education and being accepted for doing direct PhD program after I graduated from Software Engineering program. As this research, in particular, was directly related to my field of study, my learning will help me to a great extent for my future courses as well and as a matter of fact, I will be able to understand the concepts more in depth.

4. References

- [1] Shadi Moradi Seresht, Olga Ormandjieva and Samer Sabra. Automatic Conceptual Analysis of User Requirements with the Requirements Engineering Assistance Diagnostic (READ) Tool. In Proceedings of 6th International Conference on Software Engineering Research, Management and Applications (SERA 2008) [Publisher: IEEE Computer Society Press. Acceptance rate: 42%]

- [2] Moradi Seresht, Sh., Ormandjieva, O., "Automated Assessment of Use Case Elicitation from Requirements Text" Accepted at 11th Workshop on Requirements Engineering, WER 2008

- [3] Ahia-Tabibi, A., "Extracting Summary-level Use-Cases Descriptions by Text- Partitioning software Requirement Document", July 2008, from <http://users.encs.concordia.ca/~nlp-se/Ayrin/>

- [4] Hussain, H.M.I. (2007). Using Text Classification To Automate Ambiguity Detection in SRS Documents, A Thesis in the Department of Computer Science & Software Engineering, Concordia University, August 2007, Montreal, Quebec Canada.

- [5] Hussain, I., Ormandjieva, O. & Kosseim, L. (2007). Quality Assessment of SRS text by Means of a Text Classification System. Proceedings of the Seventh International Conference on Quality Software (QSIC-2007), pp. 209-218. October 2007, Portland, Oregon.

[6] Saber, S., "EVP software Requirement Document", 2008, from <http://samer.sabra.ca/evp/>

6. Glossary

- **READ:** Requirement Engineering Assistance Diagnostic
- **SRS:** Software Requirement Document
- **XML:** Extensible Markup Language
- **UML:** Uniform Modeling Language
- **NLP:** Natural Language Processing
- **NSERC:** Natural Science and Engineering Research Council
- **CDMP:** Canadian Distributive Mentored Project
- **EVP:** eclipse visualization Plug-in