Software Requirements Specification

Extracting Summary-level Use-Cases Descriptions

by Text- Partitioning

Version 3

Prepared by: Airin Ahia-Tabibi

Supervised by: Dr. Ormandjieva

Concordia University

July 2008

Table of Content

1. Introduction

A conceptual 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. Then, an improved static view is generated by injecting domain-related missing information provided by Expert Comparable Contextual (ECC) models, which are extracted from reusable domain-specific data models. Figure 1 illustrates the NLP-Based Quality Assessment in Requirements Engineering.

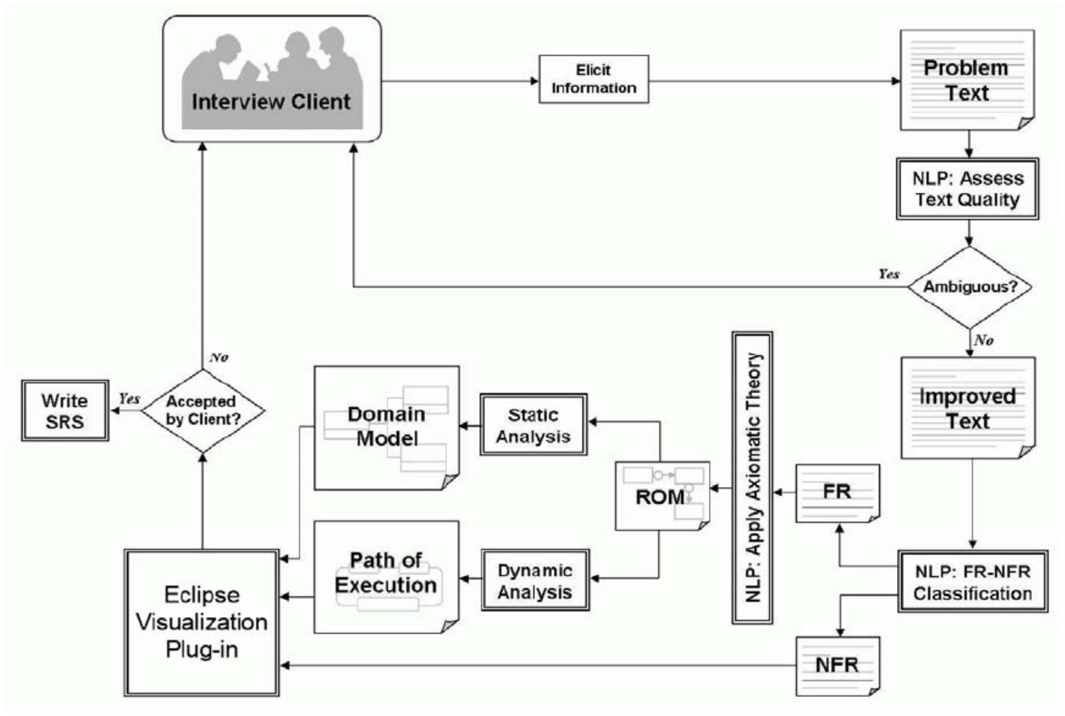

Figure 1. NLP-Based Quality Assessment in Requirements Engineering

1.1. Purpose

This Software Requirements Document (SRD) defines the requirement of the Metric-based text- partitioning algorithm in order to facilitate the drawing of Summary-level Use-case diagrams which are high-level descriptions of the service provided by the system under development, SUD, primarily by measuring the distance between sentences. This is part of the large Requirement Engineering Assistance Diagnostic (READ) project. The ultimate goal of the READ 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.

1.2. Intended Audience and Reading Suggestions

This SRD describes how the Summary-level Use-Cases descriptions are extracted from requirement text by using text partitioning algorithm. It is primarily written for the developers of the READ Project and future references as well as the software users.

1.3. Scope

The goal is to group the sentences in the requirement text by the Metric-based text-partitioning algorithm in order to facilitate the identification of Summary-level Use-cases description from the text.

The Program essential functionality is getting the improved requirement text from which the ambiguity has been removed with the help of the client [5], saving it in XML format and generating the similarity and dissimilarity tables as well as measuring the distances. It is worth mentioning that the text has already undergone the NLP text Quality assessment.

Such a grouping increases the visibility of a service in the text paragraphs of the original text. This will facilitate the job of the analysts in ensuring the completeness of the use-case descriptions and verifying the text for possible inconsistencies between otherwise scattered statements.

1.4. References

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

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

[3] Software Requirement Specification Template, exerted on July, 2008, form www.processimpact.com/process_assets/**srs_template**.doc

[4] Moradi Seresht, Sh., Ormandjieva, O., Sabra, S., "Automatic Conceptual Analysis of User Requirements with the Requirements Engineering Assistance Diagnostic (READ) Tool". Accepted at the 2008 Software Engineering Research, Management and Applications (SERA 2008), 20-22 of August 2008, Prague, Czech Republic, (2008)

[5] 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.

2. Overall Description

2.1. Perspective

As we continue with the development of the visualization functionality of the READ project [2], the Eclipse platform is used to code the algorithm by using Object-oriented methodology and the Java programming language.

2.2. Characteristics

User: The system has only one user which has access to all functions.

Hardware: The Eclipses platform must be able to write and read files to a workplace.

Software: The OS must allow the Eclipse platform to write and read files to a workplace.

Security: There is no security feature implemented in the system and any user can access all the functions.

Reliability: The system will ensure that the generated tables and distances are correct.

2.3. Assumptions and Dependencies

The format of the input file has been assumed to be an improved pure text. This document, however, is not meant to be exhaustive elaboration of all functions. It is just a clear and concise outline.

2.4. Contents

After the section 1 and 2 which were the Introduction and Overall Description, the rest of the document is designed as follow. Section 3 describes the Object Oriented Requirement Analysis. Section 4 is about the interfaces and section 5 describes the Non-functional recruitments. The document finishes by section 6 which is the Glossary.

3. Object Oriented Requirement Analysis

3.1. Use Cases

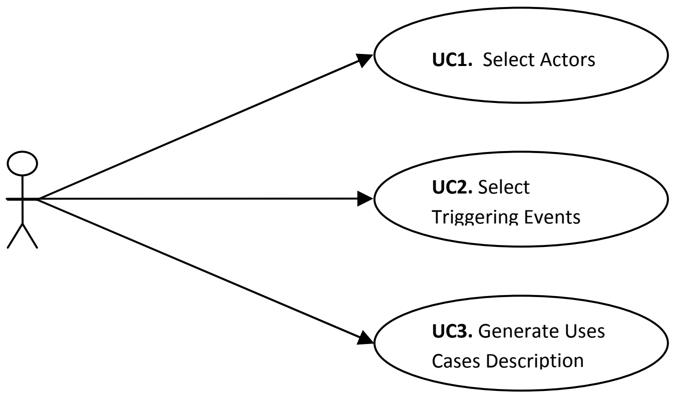
UC1. Select Actors
UC2. Select Triggering Events
UC3. Generate Uses Cases Description

Figure 2. Use Cases

UC1. User chooses the actors from the corresponding XML file which is already created.

UC2. User chooses the triggering events from the corresponding XML file which is already created.

UC3. The partitioning algorithm consists of 2 parts. First part is to use some predefined patterns to recognize the main sentences which identify the summary-level-use-cases accomplishing certain goals (system usages). The second part is to partition the original problem statement description around the chosen sentences into summery-level-use-cases descriptions, one partition per main sentence. Each partition groups the sentences related to one goal. The equivalence criterion is the closeness of the sentences to the corresponding main sentence.

The extended functions, in addition, are separating sentences in the paragraph by using sentence delimiter program, eliminating redundant words, and recognizing the primary and secondary actors as well as the verbs.

4. Interfaces

This portion of the project is internal to the system and therefore does not considerably involve interaction with the user and as a result there is no direct interface. Once it gets integrated, the interface will be that of the EVP [2].

5. Non-Functional Requirements

5.1. Design Constraints

As it is mentioned, the Extracting Summary-level Use-Cases Descriptions by Text-Partitioning is only one portion of the READ project and it suppose to interact with the rest other parts. Therefore, it will be implemented as an Eclipse application. Its design constraints are defined by the Eclipse frameworks that it built on. In Particular, it depends on the UML2 component and will be developed using Java.

5.2. Quality Attributes

Maintainability: Ensure that the Extracting Summary-level Use-Cases Descriptions by Text- Partitioning is maintainable through documentation and good design.

Usability: Ensure that the software is appealing, user friendly and easy to fallow.

5.3. Acceptable Criteria

5.3.1 Essential Functionality

- Ability to get the Improved Requirement text as an input.
- Ability to save it as an XML file.
- Ability to generate the summary level descriptions of the use cases from the requirement text.

6. Glossary

- **ECC:** Expert Comparable Contextual
- **READ:** Requirement Engineering Assistance Diagnostic
- **SRS:** Software Requirement Document
- **SUD:** System Under Development
- **XML:** Extensible Markup Language