

# Filtering Software Specifications Written in Natural Language

Núria Castell and Àngels Hernández

Universitat Politècnica de Catalunya, Departament de Llenguatges i Sistemes Informàtics, Pau Gargallo, 5. Barcelona 08028. Spain

**Abstract.** The specification phase is one of the most important and least supported part of the software development process. We have conceived SAREL (Assistance System for Writing Software Specification in Natural Language) as a tool to improve the specification phase. SAREL is a continuation of a program of research and development called LESD (Linguistic Engineering for Software Design). The purpose of SAREL<sup>1</sup> is to assist engineers in the creation of software specifications written in natural language. It is divided into three modules: the first one controls the requirement according to the writing norms, the second one obtains a conceptual representation using the Knowledge Base, and the third one carries out a series of optional analyses taking into account the following software quality properties: consistency, completeness, traceability, verifiability and modifiability. Once a requirement has been labeled as correct, its conceptual representation is added to the Requirements Base.

## 1 Introduction

The software development process starts generally with the specification phase. At this stage it is very important to control the quality of specifications in order to detect possible mistakes as early as possible. The correction of errors in the development and implementation phases implies spending more time and effort than in the specification phase. This is the reason why developers increasingly try to identify possible mistakes in the early phases of software development.

During the initial phase, software specifications for complex systems result in bulky documents since they are often written in natural language. Documentation writing is guided by norms which define the linguistic restrictions required to satisfy the specifications. These norms are of two types: those relating to the use of natural language in general (for example, [2] and [3]); and those that are based on terminological restrictions related to a particular domain (for example, the ESA - European Space Agency - norms). Both of them restrict the use of natural language through a set of rules which limit various irregularities (polysemy, paraphrase, ambiguity, vagueness..). Even though the norms define linguistically precise restrictions, the frequent failure to observe them makes it difficult for the consequences of such breaches to be detected afterwards. In addition to linguistic restrictions, the norms also include Software Engineering constraints related

---

<sup>1</sup> This work is supported by CICYT (TIC93-420). email: castell,ahernandez@lsi.upc.es

to the quality factors of the specifications, such as consistency, completeness, traceability, modifiability and verifiability.

The purpose of the system we are developing (SAREL) is to assist engineers in writing specifications in natural language taking into account the writing norms and the software quality properties.

Other systems which tackle the problems associated with the specification phase are RA [13] [14], FRORL development system [17] and OICSI [11] [15]. The Requirements Apprentice (RA) assists a human analyst in the creation and modification of software requirements. The FRORL development system facilitates the specification, analysis and development of a software system. The OICSI is a system prototype that supports the analyst in the process of problem-statements acquisition, elicitation modelling and validation. Although the goals of the above systems vary slightly from SAREL system's goal, they form a reliable set of references for our research work. A comparative study can be found in [10].

In section 2, we describe our system (SAREL) in a general way. Sections 3, 4 and 5 describe each module of SAREL. The paper concludes with a discussion of the future goals of our research.

## **2 SAREL: an Assistance System for Writing Software Specifications in Natural Language**

The SAREL system is a continuation of a program of research and development called LESD (Linguistic Engineering for Software Design). This project was instigated by the ARAMIIHS center in Toulouse (France) and was carried out by French and Spanish researchers. LESD aimed [4], [5], [16] to develop computational tools which would (1) allow conceptual interpretation of functional or preliminary software aerospace specifications written in English; (2) permit evaluation of quality factors by means of reasoning algorithms applied to the conceptual representation; and (3) help the engineers handle documentation.

In this context, the main goal of SAREL is to assist an engineer in the creation of software specifications written in natural language. To be exact, the specifications are written in English because this is the most common language in the aerospace domain (the LESD domain). The examples we present in this paper are taken from aerospace system documents. A preliminary version of our system was described in [10]. The assistance process, broken down into several steps, validates every requirement introduced by the engineer taking into account the writing norms (for instance [2], [3]) and the quality properties [9]. This process incrementally constructs a conceptual representation of the specification. The controls, shown in figure 1, can be grouped into three modules: the Style Refinement Module, the Conceptual Refinement Module and the Software Quality Control Module. Once a requirement has been checked and is correct, its conceptual representation is added to the Requirements Base.

**Fig.1.** The Modules of SAREL

### 3 Style Refinement Module

This module controls the requirement according to the writing norms and is broken down into four steps: lexical analysis, syntactic-semantic analysis, ambiguity control and simplicity control. This module first controls the lexicon used in a requirement, then analyses its coherence and finally validates its surface form.

#### 3.1 Lexical Analysis

This analysis carries out the control of lexicon used in the specification. Given a requirement, the lexical analysis verifies that the words belong to the application domain lexicon. To do so, SAREL uses three different lexicons:

- The extended lexicon includes all words related to the application domain.
- The allowed lexicon is an extraction of the previous one. Every word in the allowed lexicon represents a set of synonyms on the extended one.
- The general lexicon contains general English words.

Firstly the analyzer will ensure that all words contained in the requirements belong to one of the three lexicons. After that, all words belonging to the extended lexicon will be substituted by synonyms from the allowed lexicon. The lexicon division offers the engineer some kind of writing flexibility, because he can use an extended lexicon instead of the allowed lexicon which is more restrictive.

At present we are adapting an On-line Lexical Database (WordNet) [12] to the aerospace domain in order to build the above lexicons. But the lexical validation process is independent of the application domain.

#### 3.2 Syntactic-semantic Analysis

In this step, the Alvey parser [6] is used. This parser is based on Montague's logic and its output is a tree-like semantic representation. This environment has been modified and adapted to the aerospace domain and it provides satisfactory results.

Although semantic rules are associated with syntactic ones, the GDE (Grammar Development Environment) is not able to capture the whole meaning of a requirement and a conceptual analysis must subsequently be performed.

#### 3.3 Ambiguity Control

It is possible to obtain more than one semantic representation from a requirement. This situation appears when the requirement contains some kind of ambiguity. The goal here is to identify the representation which corresponds to the engineer's idea. For example the requirement "*The AEROS will monitor the computer on-board and the status of the space-vehicle*" can give rise to two possible interpretations:

- a) "*The AEROS will monitor the computer on-board the space-vehicle and the AEROS will monitor the status of the space vehicle*"

- b) *“The AEROS will monitor the computer on-board and the AEROS will monitor the status of the space vehicle”*

The ambiguity controller applies a series of rules to the set of semantic representations in order to qualify them. An example of a possible rule is the one which asserts that the preposition *of* is always related only with the last nominal group. This controller cooperates with the engineer to select the correct semantic representation.

### 3.4 Simplicity Control

One of the properties expressed by the writing norms is simplicity. From the semantic representation of the requirement the controller detects whether the structure of the sentence is simple or compound. The following requirement does not have this property.

**Req.:** *“The AEROS will control the computer on-board the space-vehicle and the AEROS will control the automatic systems of the flight configuration”*

If the requirement is composed of two simple requirements, the validation process can continue with these two in a sequential way.

## 4 Conceptual Refinement Module

This module validates the requirement in relation to the Requirements Base (RB). At first it obtains a conceptual representation using the Knowledge Base (KB). Both RB and KB use a frame-based formalism [16]. From the conceptual representation this module detects duplicated information. At present we have already defined the Knowledge Base, and the Requirements Base is manually incremented when requirements have been analysed. The automatic process to integrate requirements is still at a development phase.

### 4.1 Conceptual Analysis

This analysis identifies in the Knowledge Base those entities involved in the semantic representation and constructs the requirement conceptual representation. Figure 2, from [16], shows the conceptual representation of two requirements:

**Req-1:** *“The AEROS will monitor the systems of the space-vehicle”*

**Req-2:** *“The AEROS will control the automatic systems of the space-vehicle”*

### 4.2 Duplication Control

The function of the duplication controller is to verify that the requirement introduced by the engineer contains new information. To do so, it matches the requirement conceptual representation to the Requirements Base in order to discover possible duplications. The following example shows a set of equivalent requirements since the system obtains the same conceptual representation for them.

**Fig. 2.** Conceptual Representation

**Req-1:**“*The AEROS will monitor the computer on-board the space-vehicle*”

**Req-2:**“*The computer on-board the space-vehicle will be monitored by the AEROS*”

**Req-3:**“*In the space-vehicle, the computer on-board will be monitored by the AEROS*”

**Req-4:**“*The computer on-board the space-vehicle will be monitored*”

At first the duplication controller searches on the Requirements Base all requirements containing the activity which appears on the new requirement. All of them will be tested in order to discover if both the agent and object are the same. When the controller detects duplication, it will offer the engineer the possibility of refining it.

## 5 Software Quality Control Module

This module carries out a series of optional analyses which validate the global Requirements Base incremented with the new requirement. The goal is to offer information about the software quality properties which have been considered most relevant in LESD (*completeness, traceability, consistency, verifiability and modifiability*).

Once a requirement has been validated and the information about software quality has been presented to the engineer, he decides if the conceptual representation of requirements must be added to the Requirements Base. This process incrementally constructs the conceptual representation of the specification written in natural language.

Taking into account that these analyses are optional, the engineer could decide not to use them (it is a normal decision when adding the first requirements). In any case these modules can be used in the future to control the global quality software specification at that point. The analyses related to the quality properties already studied are described in the following paragraphs.

### 5.1 Analysis of Traceability

The goal here is to provide the engineer with information about the traceability links of the requirement. To do so, SAREL activates a set of existing algorithms [16] which control the traceability in order to show the relationships between the introduced requirement and a subset of requirements of the RB (either more specific or more general). From this information the engineer can see the relationships between requirements introduced up to this point. From the next requirement :

**Req-1:** “*The AEROS will receive the data of the space vehicle*”

the system displays, among others, the following more general requirement:

**Req-2:** “*The system will monitor the data of the space vehicle*”

The second requirement is more general than the first because AEROS is the main instance of “system” and “receive” is part-of “monitor”.

The lack of links shows that the requirement is isolated in relation to the rest of requirements.

### 5.2 Analysis of Completeness

At this stage SAREL activates reasoning mechanisms which are being developed [1] taking into account several aspects of completeness. In order to control completeness, it is necessary to have available a general hierarchy of actions-subactions that can be associated to any set of specifications related with an aerospace system. An example of this hierarchy is present in the activity *monitor*, which comprises three subactivities: *receive*, *analyse* and *display*. Given the next requirement:

**Req.:** “*During the launch phase, the AEROS will analyse and display the status of the space vehicle*”

taking into account the hierarchy described above, the system analyses the relationships among requirements in the Requirements Base. If there is no requirement containing the *monitor* activity, it will inform the engineer. In the same way, if the system notices that there is a requirement that contains the *monitor* activity but there is no requirement containing the subactivities in the hierarchy, it will inform the engineer that the current specification could be incomplete.

### 5.3 Analysis of Modifiability

The complexity and consistency of future modifications of software specifications depends on the level of propagation of a given modification in all requirements affected by that modification. The concept of modifiability in LESD has been formalized [7] [8] according to the levels of interconnection between the specifications requirements. The range of possible values is  $[0..1]$ , but the range of acceptable values is a subinterval of this. In case the obtained measure does not fall within this second range, the engineer must study a possible problem of excessive or insufficient interconnection.

## 6 Conclusion and Future Research

Our research deals with the control of quality in specifications written in natural language. Since the specification phase is performed by a human analyst, we have designed an assistance system for writing software specifications in natural language. The quality of the specification is studied from two points of view: writing norms and software quality properties.

The assistance process is divided into three modules: the Style Refinement Module, the Conceptual Refinement Module and the Software Quality Control Module. Once a requirement has been validated by these modules, it is added to the Requirements Base. During this process we incrementally obtain a conceptual representation of the specification.

In order to exhaustively develop our system, future research work will be focused on a) designing an automatic mechanism which provides Conceptual Representations from Semantic Representations, b) developing semiautomatic mechanisms which build domain Knowledge Bases from technical documents, and c) defining the validation process of the obtained Knowledge Base. Moreover, the study of selected quality factors will continue.

## References

1. Alvarez J., Castell N. "An Approach to the Control of Completeness Based on MetaKnowledge", Technical report, LSI-94-50-R Dept. of LSI, Universitat Politècnica de Catalunya, 1994.
2. ANSI/IEEE Std 729-1983. *IEEE Guide to Software Requirements Specifications* 1983.



3. Association Européenne des Constructeurs de Matériel Aéronautique. *AECMA Simplified English, A Guide for the preparation of aircraft maintenance documentation in the international aerospace maintenance language*, December 1989.
4. Borillo M., Borillo A., Castell N., Latour D., Toussaint Y., Verdejo M.F. "Applying Linguistic Engineering to Software Engineering: The traceability problem". In *Proceedings of the European Conference on Artificial Intelligence (ECAI92)*, pages 593-595, Viena, Austria, August 1992.
5. Borillo M., Toussaint Y., and Borillo A. "A. Motivations du project LESD". In *Conference on Linguistic Engineering'91*, Versailles, France, January 1991.
6. Briscoe T., Grover C., Boguraev B., Carroll J. "The ALVEY Natural Language Tools Project Grammar: A Large Computational Grammar". Technical report, ALVEY Documents, Cambridge Univ., Computer Laboratory, UK, 1987.
7. Castell N., Slavkova O. "The Modifiability Factor in the LESD Project: Definition and Practical Results", Technical report, LSI-95-7-R Dept. of LSI, Universitat Politècnica de Catalunya, 1993.
8. Castell N., Slavkova O. "Metrics for Quality Factors in the LESD Project". In 5th European Software Engineering Conference (ESEC'95), Sitges, Spain, 1995.
9. Castell N., Slavkova O., Toussaint Y. and Tuells A. "Quality Control of Software Specifications written in Natural Language". In *Proceedings of the Seventh International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE'94)*, Austin, Texas, USA, 1994.
10. Hernández, A. "SAREL: An assistance system for writing software specifications in natural language". In *Proceedings of the IBERAMIA '94*, ISBN 980-6168-16-X Caracas, Venezuela, 1994.
11. Jarke M., Bubenko J., Rolland C., Sutcliffe A. and Vassiliou J. "Theory Underlying Requirement Engineering: An Overview of NATURE at Genesis". In *Proceedings of the IEEE International Symposium on Requirements Engineering (RE'93)*, San Diego, California, USA, 1993.
12. Miller, G.A. "Wordnet: A Dictionary Browser" in *Information in Data, Proceedings of the First Conference of the UW Centre for the New Oxford Dictionary*, Waterloo, Canada: University of Waterloo, 1985.
13. Reubenstein H.B. and Waters R.C. "The Requirements Apprentice: Automated Assistance for Requirements Acquisition". *IEEE Transactions on Software Engineering*, 17:226-240, 1991.
14. Rich C. and Waters R.C. *The Programmer's Apprentice*. Reading, MA: Addison-Wesley, and Baltimore, MD. ACM Press, 1990.
15. Rolland C., Proix C. *A Natural Language Approach for Requirements Engineering*. Conceptual Modeling, Databases and CASE: An Integrated View of Information Systems Development, P. Loucopoulos, R. Zicari (eds.) WILEY, 1992.
16. Toussaint Y. *Méthodes Informatiques et Linguistiques pour l'Aide à la Spécification de Logiciel*. PhD thesis, Université Paul Sabatier, Toulouse, 1992.
17. Tsai J.P., Weigert T. and Jang H.C. "A Hybrid Knowledge Representation as a Basis of Requirement Specification and Specification Analysis". *IEEE Transactions on Software Engineering*, 18:1076-1100, 1992.