# The Software Requirements Modelling in SAREL

Núria Castell and Àngels Hernández

Universitat Politècnica de Catalunya
Departament de Llenguatges i Sistemes Informàtics
Jordi Girona 1-3. Mòdul C6.
Barcelona, Spain
e-mail: {castell,ahernandez}@lsi.upc.es

**Abstract.** This paper aims to describe the software requirements modelling used by SAREL (Assistance System for Writing Software Specification in Natural Language). The initial purpose of SAREL[1] was to assist engineers in the creation of quality software specification written in natural language. The functionality of the system is twofold: vertical processing and horizontal processing. In vertical processing the input is a software specification written in natural language and the output is the associated conceptual representation. Working in horizontal processing the input is two different conceptual representations and the output is the information about the correspondence between them. All the tasks done by the system rely on the modelling of the software requirements.

## 1 Introduction

The specification phase is one of the most important and least supported parts of the software development process. In this stage it is very important to control the writing of the Software Requirements Specifications because many more mistakes can be detected if the writing is clear and concise. In this work frame other systems which tackle the problems associated with the specification phase are RA [Reubenstein et al.,91], NATURE [Jarke et al.,93] or with the quality of documents in general like EasyEnglish [Bernth97], ACE [Fuchs&Schwitter96]. Documentation writing is guided by the norms which define the linguistic restrictions required and also by the software engineering constraints related to the quality factors of the software specifications.

In order to obtain a high-quality software requirements specification (SRS) we have designed the SAREL help-system. The controls (shown in figure 1) contained with in this system can be grouped into three modules: the Style Refinement Module, the Conceptual Refinement Module and the Software Quality Control Module. The last two modules use the Knowledge Base which contains the domain knowledge and the Requirements Base which contains the set of specific requirements. In this paper we present the software requirements modelling used by SAREL and the way the system uses it.

[1] Supported by CICYT (TIC-96-1243-603-02) and CIRIT (1997SGR51)

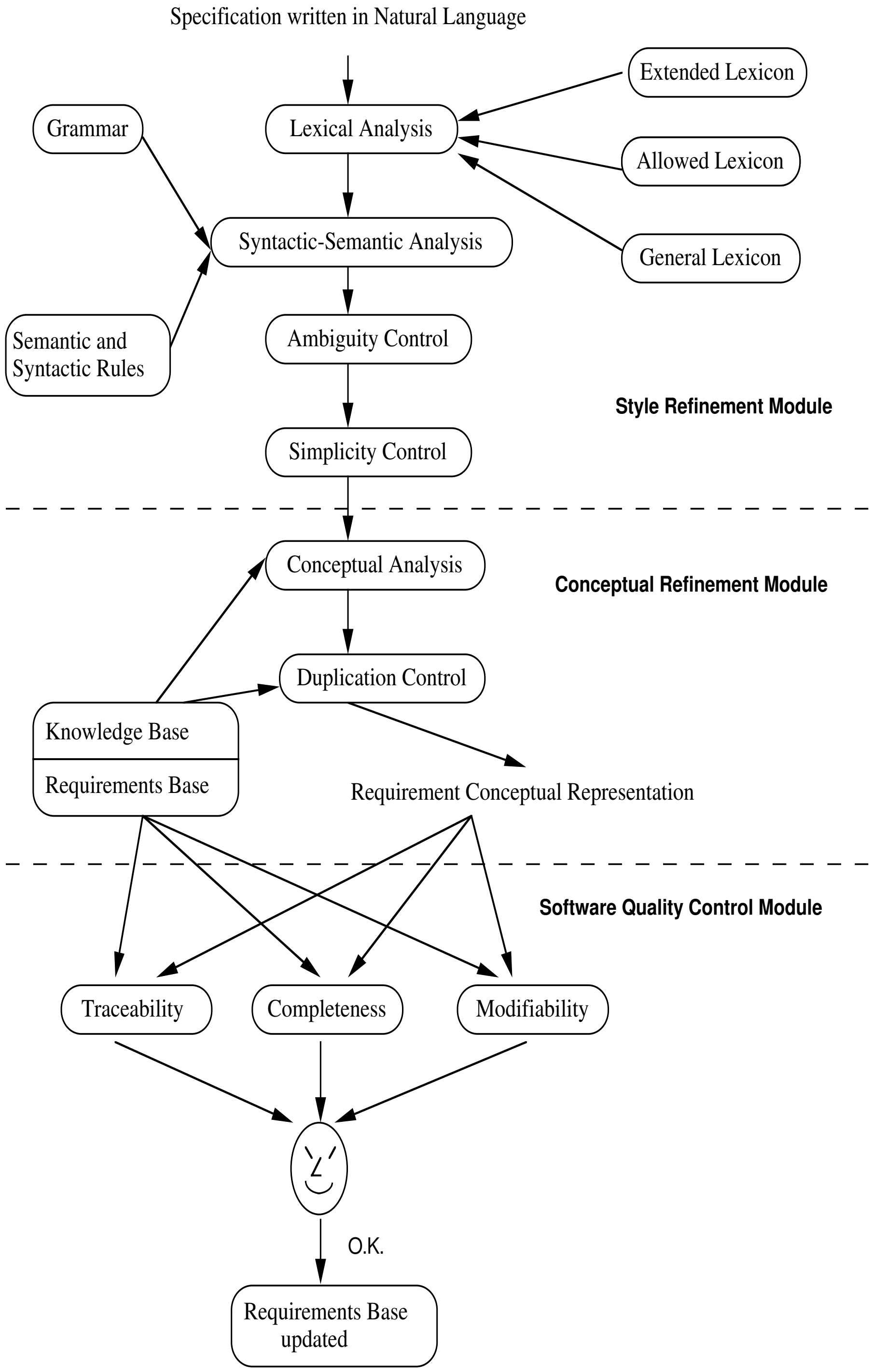


Figure 1. Modules of SAREL

In section 2 we present briefly the different modules mentioned above and the two functionalities of SAREL. In section 3 we describe the information contained in the Knowledge Base. Section 4 presents the Requirements Base using the software requirements modelling. Following that we explain in section 5 how to use the software requirements modelling from the point of view of the two different functionalities. The conclusions are presented in section 6.

## 2 The SAREL System.

The main goal of SAREL is to assist an engineer in the creation of quality software specifications written in natural language. A preliminary version of this system was described in [Hernandez94]. The assistance process validates every requirement introduced by the engineer taking into account the writing norms (for instance [ANSI83]), [AECMA89]) and the quality properties [Castell et al.,94]: consistency, completeness, traceability, modifiability and verifiability. This process incrementally constructs a conceptual representation of the software specification. The controls, shown in figure 1, can be grouped into three modules:

- Style Refinement Module controls the requirement according to the writing norms and it is split into four steps: (1) the lexical analysis verifies that the words belong to the application domain lexicon; (2) the syntactic-semantic analysis is performed and a tree-like semantic representation is produced; (3) the ambiguity control helps the engineer to identify the correct representation between the possible interpretations; (4) the simplicity control detects whether the structure of the sentence is simple or compound.
- Conceptual Refinement Module validates the requirement in relation to the Requirements Base. At first it obtains a conceptual representation using the Knowledge Base and after that detects duplicated information.
- Software Quality Control Module carries out a series of optional analyses [Alvarez&Castell94], [Castell&Slavkova95] which validate the global Requirements Base which increaes with the new requirement. The goal is to offer information about the software quality properties which have been considered most relevant in the LESD project [Borillo et al.,92]: completeness, traceability, consistency, verifiability and modifiability.

The SAREL system has two different funcionalities depending on the user's goal: vertical processing and horizontal processing.

### 2.1 Vertical Processing

The Vertical Processing basically corresponds to the sequential application of the controls shown in figure 1. The input is a software specification written in natural language and the output is its associated Conceptual Representation. The document is processed, requirement after requirement, by the Style Refinement Module and the Conceptual Refinement Module in order to obtain the Conceptual Representation that can be optionally validated by the engineer using the Software Quality Control Module. Once a requirement has been checked, its conceptual representation is added to the Requirements Base. Using this functionality, it is possible to obtain the Conceptual Representation associated with the informal software specification. This means that the information represented can be consulted in a collective or individual way by the engineers in a more reliable format. A more precise description of this functionality can be found in [Castell&Hernández95].

### 2.2 Horizontal Processing

In Horizontal Processing the input is of two different conceptual representations, and the goal here is to offer information about the correspondence between them. The correspondence analysis search for every $requirement_1$ in $document_1$ its corresponding $requirement_2$ in $document_2$. Where $document_1$ corresponds with the user company that needs to develop a computer system and therefore, $document_2$ corresponds with the software company that will carry this out. The system will give a correspondence measure based on [Romesburg84] similarity analyses applied over the components of the requirements. Depending on the value of this measure, the correspondence will be tagged as: Correct, Excess or Excess-Insufficient. See [Castell&Hernández97] for a more precise description.

## 3 The Knowledge Base

The Conceptual Refinement Module uses a domain representation to generate the Conceptual Representation of the requirement that is being processed. Our Knowledge Base contains this domain representation using a frame-based formalism [Alvarez96]. This Knowledge Base, among other information includes the representation of the structure of a Software Requirements Specification (SRS).

Taking into account [IEEE93] the three essential parts in an SRS, we can distinguish the **introduction** and the **overall description** as parts that should be used in the Knowledge Base construction. Upto a point these two parts contain all the background information needed to understand the problem as a whole. So using information extraction techniques [Turmo et al.,98] the entities involved in the application domain can be obtained.

The information contained in the **specific requirements section** will correspond with the information represented in the Requirements Base. We want to point out that all the entities and all the activities contained in a given requirement correspond with entities and activities represented in the Knowledge Base.

## 4 The Requirements Base

In this section we present in more detail the Requirements Base used by SAREL. Firstly the Software Requirements Classification is presented. This is based on the description of the specific requirements section [IEEE93]. There are other approaches that use this classification such as KARAT [Tschaitschian et al.,97]. In our approach the original classification has been adapted and it forms a selection of the most general classes. Secondly the Semantic Roles subsection contains some examples of semantic roles based on [Allen95] needed to model the different kinds of software requirements. Finally the software requirements modelling is described associating the different semantics roles required for each type of software requirement.

### 4.1 Software Requirements Classification.

The software requirements are grouped in the specific requirements section, and each of them can be classified in one of the following general classes (the only ones considered by SAREL at present):

- Functional Requirements define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs.
- Performance Requirements specify both the static and the dynamic numerical requirements placed on the software or on human interaction with the software as a whole.
- External Interface Requirements correspond to a detailed description of all inputs into and outputs from the software system.

### 4.2 The Semantic Roles

This subsection presents the notion of semantic role and a subset of all the semantic roles that must be defined in order to model the software requirements. From the following sentences [Allen95]:

- *John broke the window with the hammer*
- *The hammer broke the window*
- *The window broke*

*John, the hammer*, and *the window* play the same semantic roles in each of these sentences. *John* is the actotr (AGENT), *the window* is the object (PATIENT), and *the hammer* is the instrument (INSTR) used in the act of breaking of the window. Next there are some examples of Allen's semantic roles.

- AGENT: intentional causation
- INSTR: force/tool used in causing the event
- PATIENT: the thing affected by the event
- EXPERIENCER: the person involved in perception
- BENEFICIARY: the person for whom an act is done

Allen's set of semantic roles has been extended with new roles such as FEATURE, MEASUREMENT, UNITS ... in order to cover all the possible meanings associated with software requirements.

### 4.3 Modelling Software Requirements

In order to control the set of software requirements contained in the SRS document we have considered it necessary to establish the different roles associated to each kind of requirement. Below we present the set of semantic roles required for each class mentioned in 4.1.

1. For Functional Requirements the Semantic Roles required are: AGENT, ACTION and PATIENT. The conceptual representation associated with the functional requirement: *The transfer system shall transfer the clock-in to the communications server* will be as follows:
   (DEFINE-NODE requirement-x
   (AGENT transfer system)
   (ACTION transfer)
   (PATIENT clock-in)
   (TO-LOC[2] communications server))

2. For Performance Requirements need the following semantic roles are required: PATIENT, MEASUREMENT, AT-VALUE, (or FROM-VALUE and TO-VALUE), and UNIT. An example of performance requirement is: *An identifying card shall be made-up in less than 1 minute.* The associated conceptual representation is:
   (DEFINE-NODE requirement-y
   (PATIENT identifying card)
   (MEASUREMENT making-up time)
   (AT-VALUE $< 1$)
   (UNIT minutes))
3. The semantic roles required to represent the Interface Requirements are: PATIENT, QUALITATIVE-FEATURE (or QUANTITATIVE-FEATURE and UNITS-FEATURE). The conceptual representations associated to the Interface Requirements: *The monitor of the personal computer will be colour and 15"* are given below:
   (DEFINE-NODE requirement-z
   (PATIENT monitor)
   (QUALITATIVE-FEATURE colour))
   (DEFINE-NODE requirement-t
   (PATIENT monitor)
   (QUANTITATIVE-FEATURE 15)
   (UNITS-FEATURE inches))
   At this point it is necessary to say that the Knowledge Base contains the relation which defines the monitor as part of the personal computer.

## 5 The use of the Requirements Base

Taking into account the two functionalities of the SAREL system, the Requirements Base can be used in two different ways.

[2] This is a optional semantic role

From the Vertical Processing point of view, the Requirements Base is used by the analysis tasks contained in the Software Quality Control Module. For example the traceability analysis provide the engineer with information about the traceability links of one fixed requirement. In this case SAREL activates a set of existing algorithms which control the traceability in order to show the relationships between the introduced requirement and a subset of requirements of the Requirements Base. The relationships are based on the entities and the relations stated in the requirements. From this information the engineer can see the relationships between requirements introduced up to this point. A lack of links shows that a requirement is isolated in relation to the rest of requirements.

From the Horizontal Processing point of view, the correspondence analysis searches in the Requirements Base for every $requirement_1$ in $document_1$ its corresponding $requirement_2$ in $document_2$. During this search it is useful to take into account information about what kind of requirement $requirement_1$ is (Functional, Performance or Interface). The measure, based on the correspondence associated to the required semantic roles will be more influential than the correspondence associated to the optional semantic roles.

## 6 Conclusions

The main goal of this paper has been to present the Software Requirements Modelling used by SAREL. This system was conceived to assist engineers in the creation of quality software specifications using a knowledge-based approach. SAREL incrementally constructs a Requirements Base where all the controls are performed. The system has been extended for helping engineers in two ways. On the one hand, in vertical processing only one document at time is considered, and the system assists the engineer in the creation of this document. In this case the Software Quality Control Module tasks are based on the representation of the requirements (their entities and their relations). On the other hand, in horizontal processing the system analyses the correspondence between two software specifications. The correspondence analyzer explores the Requirements Base taking into account the kinds of requirements and the filled roles.

## References


[Allen95] Allen, J. *Natural Language Understanding* The Benjamin/Cummings Publishing Company, Inc. 1995

[Alvarez&Castell94] 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.

[Alvarez96] Alvarez, J. *Yet Another Yet Another (YAYA)* Technical report, LSI-96-15-T Dept. of LSI, Univers. Politècnica de Catalunya 1996.

[ANSI83] ANSI/IEEE Std 729-1983: *IEEE Guide to Software Requirements Specifications* 1983.

[AECMA89] Association Européene 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.

[Bernth97] Bernth, A. *EasyEnglish: A Tool for Improving Document Quality.* In *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP'97).* pages 159-165.

[Borillo et al.,92] 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.

[Castell&Hernández95] Castell, N. & Hernández, A. *Filtering Software Specifications Written in Natural Language*, In *Proceedings of the 7th Portuguese Conference on Artificial Intelligence (EPIA'95)*, LNAI 990, pages 447-455, Funchal, Madeira Island, Portugal 1995.

[Castell et al.,94] Castell, N. & Slavkova, O. & Toussaint, Y. & 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)*, pages 37-44, Austin, Texas, USA, 1994.

[Castell&Hernández97] Castell, N. & Hernández, A. *The use of SAREL to control the correspondence between Specification Documents.*In *Actas de la VII Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA'97)*, pages 529-539, Málaga, Spain 1997.

[Castell&Slavkova95] Castell, N. & Slavkova, O. *Metrics for Quality Factors in the LESD Project.* In *5th European Software Engineering Conference (ESEC'95)*, LNCS 989, pages 423-437, Sitges, Spain, 1995.

[Fuchs&Schwitter96] Fuchs, E. & Schwitter, R. *Attempto Controlled English (ACE).* In *Proceedings of The First International Workshop On Controlled Language Applications.* Katholieke Universiteit Leuven, pages 124-136, Belgium, 1996.

[Hernandez94] Hernández A. *SAREL: An assistance system for writing software specifications in natural language.* In *Proceedings of the IBERAMIA'94*, pages 247-262 ISBN 980-6168-16-X, Venezuela, 1994.

[IEEE93] IEEE Std 830-1993: *IEEE Recommended Practice for Software Requirements Specifications* 1993.

[Jarke et al.,93] Jarke, M. & Bubenko, J. & Rolland, C. & Sutcliffe, A. & and Vassiliou, J. *Theory Underlying Requirements Engineering: An Overview of NATURE at Genesis.* In *Proceedings of the IEEE International Symposium on Requirements Engineering (RE'93)* San Diego, California, USA, 1993.

[Reubenstein et al.,91] Reubenstein, H.B. & Waters, R.C. *The Requirements Apprentice: Automated Assistance for Requirements Acquisition. IEEE Transactions on Software Engineering* 17:226-240 1991.

[Romesburg84] Romesburg, H. C. *Cluster analysis for researchers.* Belmont, Calif.:Lifetime Learning Publications, 1984.

[Tschaitschian et al.,97] Tschaitschian, B., Wenzel, C. and John, I. *Tuning the quality of informal software requirements with KARAT.* In *Proceedings of the Third International Workshop on Requirements Engineering: Foundations of Software Quality (REFSQ'97)* Spain, 1997.

[Turmo et al.,98] Turmo, J. Català, N. Rodriguez, H. *TURBIO: A System for Extracting Information from Restricted Domain Texts. (IEA/AIE'98)*, LNAI 1415, pages 708-721, Benicàssim, Spain, 1998.