Abstract

In many software systems development projects, the documents available for requirement analysis are in natural language. However, their quality is not always optimal, owing to the presence of ambiguous expressions which may be interpreted in different ways. This may be an obstacle against understanding the problem and against the identification and subsequent modelling of requirements. It may, moreover, hamper the entire development process and negatively affect the quality of the system. This paper proposes a scheme with which to define indices of ambiguity, both structural and semantic. It also investigates the feasibility of calculating these indices using the knowledge base of a natural language processing system and the outputs produced during the text parsing phase. In this way it may be possible rapidly to signal the terms and phrases open to several interpretations, thereby improving the quality of the requirements analysis process.

Keywords: Requirements Engineering, Ambiguity Levels, Natural Language Processing

1. Introduction

Ambiguity is intrinsic to natural language. Programming languages and formal specifications languages were developed essentially to obtain unambiguous commands and descriptions. Ambiguity in the code or in the specifications, especially in the case of safe-critical systems, may have disastrous consequences. No less serious are problems caused by ambiguous descriptions of requirements, which as well as influencing the success of the project at the technical level\(^1\) may also have damaging consequences at the contractual one. The results of a recent survey confirm that the majority of the documents available for comprehension of the given problem and the eliciting of requirements are in natural language. Whatever approach is used to model them, therefore, tools are needed which identify the presence of ambiguities in texts as early as possible. These tools, moreover, may provide useful guidance in the writing of requirements and their preliminary analysis, while also assisting the analyst in the application of requirements collection techniques. For example, when a questionnaire or an interview is being designed, the questions can be tested to reduce the amount of ambiguity in them.

Ambiguity has been studied in almost all the artificial intelligence and computer science topics, perhaps most notably, Vision, Signal Processing, Robotics, etc. However, for the purposes of this paper we are interested in research on natural language processing (NLP). The problem of the understanding and automatic generation of natural language, in fact, is closely bound up with the removal of ambiguity. A first point to stress is that there are various types of ambiguity, and that they are detected and handled in different ways. In particular, one can distinguish between the semantic ambiguities of words or phrases, and the structural ambiguities that arise from the various roles performed by words within a sentence and in connecting the parts of it together.

This paper introduces a scheme for identifying the various

\(^1\) Pre-print ICS2000

\(^2\) See e.g. the case described in [8], p. 126.
levels of ambiguity using the linguistic tools employed for NLP. The application of the ambiguity measures proposed will be investigated with reference to a system which permits deep analysis of texts. This analysis is obtained through a sequence of steps, the intermediate outputs of which contain information useful for identification of the various types of ambiguity. The paper is organized as follows. The next section conducts general discussion of the role of natural language and therefore of ambiguity in requirements analysis. The third section describes the various types of ambiguity, using a classification which may be of used in the pre-processing of requirements documents. The fourth section defines indices of ambiguity for the various levels identified and proposes measures based on information yielded by tools for the analysis of texts in natural language. The fifth section investigates their definition and application for the LOLITA system while also discussing the shortcomings of NLP systems. The conclusions set out some proposals for further research.

2. Natural Language and Requirements Engineering

Software engineering comprises numerous activities in which non-ambiguous artificial languages may be used. However, this is not always possible, and requirements analysis is an activity which by its nature is more closely connected to the use of natural language. The reasons for this can be summarized as follows:

- **Domain/Scope.** When requirements are collected and analysed, understanding of the problem is essential before the requirements are modelled. For this purpose the analyst must interact with users and customers who have different competences and roles, and the communication usually takes place in natural language. The scenario becomes even more complex when one considers, on the one hand, the approaches that also require modelling of business processes, and on the other, the progressive integration in information systems of apparatus that do not belong among traditional hardware. When the project changes, the domain of the problem and the technical vocabulary to describe change as well, but natural language can be used with all interlocutors.

- **Input** A recent survey of software development has shown that, in the majority of cases, the documents available for requirements analysis are provided by the user or obtained by means of interviews. Moreover, 78.1% of these documents are couched in common natural language, 15.9% of them in structured natural language (e.g. templates, forms), and only 5.3% in formalised language.

- **Process** The RA process is iterative, and software developers view it as one of the most critical of activities. Since it is based on communication and on the analyst’s experience, it cannot be structured beyond a certain limit. The problems that arise when requirements are defined by the user are due to various and ineradicable causes (see e.g. [13]). Enabling the user to employ natural language not only enhances the analyst’s understanding of the problem and collaboration among the members of a project, it may also facilitate the validation of requirements. Finally, of no less importance is the fact that one of the key principles of marketing – customer orientation – is thus observed.

- **Output** Various projects have sought to use NLP systems to support the modelling of requirements on the basis of natural language texts. However, the focus of this paper is on preliminary analysis of documents to detect ambiguities and signal them to the analyst or the user, independently of adopted analysis models. Used to this end are NLP automatic tools, the main purpose of which may be described as

---

1 This survey is part of an on-line market research program of the Computer and Sciences Department of Trento University. The results are available at the site http://www.online.cs.unitn.it.
2 According to the survey cited above, requirements identification and modelling are the first and third things people would like to improve (the second is testing), but for general-purpose software, testing is in third position.
3 For a survey see e.g. [10], [1] and the papers of the NLDB conferences.
3. Ambiguity types

It is ambiguity that gives natural language its flexibility and usability, and it consequently cannot be eliminated. The nature and definition of ambiguity is a problem that has preoccupied philosophers and scientists since antiquity. Art and politics are areas in which, for different reasons, ambiguity is essential. In the case of a poem or a painting the interpretation of the reader or the observer is closely bound up with the resonances evoked in personal experience. In politics, ambiguity serves to create space for defining relationships or bargaining over shared goals – as happens, for that matter, in all human relations. Consider business decisions, which are increasingly taken by groups. Some authors have concentrated on the ‘economical’ aspects of natural language. The use of scientific terms with unequivocal meanings may help to reduce ambiguity, but it requires high cognitive effort not only by the speaker (or writers) but also by the listener (or reader). Good linguistic competence is based on the adjustment of language to pragmatic purposes in order to minimize the effort of communication. Words and sentences in natural language may correspond to a vast number of concepts. Each word in a sentence may correspond to many different meanings, and there may be more than one set of grammatical dependencies among the words in different sub-phrases of the sentence. This paper assumes the following levels of ambiguity:

- semantic ambiguity
- syntactic ambiguity.

Semantic ambiguities concern the meaning of a word or phrase, while syntactic or structural ambiguities concern the various roles performed by words in sentences and possible grammatical constructions.

To take individual words, there are words that can represent different concepts (in English a ‘bank’ is a financial institution or the edge of a river) or that can be used as both a verb and a noun (there is also the verb ‘to bank’).

At the phrase level, there may be semantic ambiguity due to the presence of ambiguous words, and structural ambiguity due to the possibility of connecting the components of the phrase in different ways (‘I saw the man in the bank’ is ambiguous if we have to decide who was in the bank).

There are also pragmatic ambiguities, and these are more difficult to detect and resolve because they concern relations more than content. For example, the appropriate pragmatic response to ‘Can you tell me the time?’ is to say what time it is, not to reply ‘yes’ or ‘no’. This paper focuses on the first two levels of ambiguity described.

4. Ambiguity measures

On the basis of the classification set out in the previous section, now introduced are a number of indices with which to measure the level of ambiguity. These measures are given general definition in this section, without reference to a particular NLP system. The next section will investigate their definition and application for the LOLITA system [7], [12].

At the level of individual words one talks of lexical ambiguity. As we have seen, a word may have different meanings (semantic ambiguity) or it may comprise different syntactic units (syntactic ambiguity).

Let us suppose that we have a finite set \( U \) of \( N \) words. The set \( W_i \) represents the possible meanings of a word \( w_i \):

\[
W_i \in U \text{ con } i = 1, \ldots, N
\]

---

6 A classic text is Grice on conversational maxims [5]. See also [6].
7 For detailed treatment of the definition and classification of ambiguity see [14].
8 For a series of studies on lexical ambiguity see [4].
9 The number of words varies over time with the introduction of new ones.
$W_i \equiv \{ m_j \mid j = 1, \ldots, n_i \}$ with $n_i \leq N$

where $W_i$ is empty if the word does not exist and coincides with $U$ for a word which refers to anything.

Corresponding to each meaning of the word $w_i$ is a possible syntactic role (e.g. noun, adjective, verb), which we denote with $r_i$.

Finally, the meanings of a word are used with differing frequencies. These frequencies can be used to weigh its ambiguity.

Hence, associated with each word may be set of terms thus constituted:

$$w_i \equiv \{ <m_j, r_j, \nu_j> \mid j = 1, \ldots, n_i \}$$

We now introduce the following measures of lexical ambiguity:

$$\alpha(w_i) = n_i \quad \text{Semantic ambiguity: number of possible meanings.}$$

$$\alpha^*(w_i) = f(n_i, \nu_i) \quad \text{Weighted semantic ambiguity: function of the number of possible meanings weighted according to their frequency.}$$

$$\beta(w_i) = n(r_i) \quad \text{Syntactic ambiguity: number of possible syntactic roles.}$$

$$\beta^*(w_i) = f(r_i, \nu_i) \quad \text{Weighted syntactic ambiguity: function of the number of possible roles weighted according to their frequency.}$$

Using these definitions, in the case of a single meaning or a single role for a sentence unitary ambiguities are obtained, while nil values are obtained for unknown words. If it is wished to emphasise non-ambiguity with a nil value of the ambiguities, it is sufficient to redefine them by subtracting 1. In this case, the measures assume a negative value for words devoid of meaning. The same applies to the ambiguity measures of a sentence.

The meaning of a sentence depends on the interpretation given both to the words comprised in it and to the number of possible parsing trees (parsing forest).

Let $S_i$ be the set of the parsing trees for the sentence $s_i$, where $S_i$ is empty if the sentence is not syntactically acceptable.

$$S_i \equiv \{ t_{i,l} \mid l = 1, \ldots, n_i \}$$

It is possible to assign a penalty each of these trees so that the trees obtained by parsing the sentence can be ordered, taking account of the choices made in their construction. Hence, associated with each sentence is a set of terms thus constituted:

$$s_i \equiv \{ <t_{i,l}, p_{i,l}, g(\alpha_{i,l})> \mid l = 1, \ldots, n_i \}$$

Bearing in mind that ambiguity is combinatorial - that is, to obtain the ambiguity of a sentence, the numbers of alternatives for each locus of ambiguity must be multiplied rather than added -, we may define:

$$\gamma(s_i) = f(g(\alpha_i)) \quad \text{Semantic ambiguity: this depends on the number of possible interpretations of the sentence obtained by permuting the meanings of the ambiguous words in the parsing trees.}$$

$$\gamma^*(s_i) = f(g(\alpha^*_i)) \quad \text{Weighted semantic ambiguity: this takes account of the weighted semantic ambiguity of the words.}$$

$$\delta(s_i) = n_i \quad \text{Syntactic ambiguity: number of possible parsing trees.}$$

$$\delta^*(s_i) = f(n_i, p_i) \quad \text{Weighted syntactic ambiguity: function of the number of possible parsing trees and the penalty associated with them.}$$

Some of the ambiguity measures introduced here are more intuitive than others: for example, semantic ambiguity for a word and syntactic ambiguity for a sentence. And they are also those which can be more easily obtained using NLP systems. Choice of the system to use should be based on a cost/benefit analysis: there exist, in fact, linguistic corpora and parsing and NLP tools of very different
complexity, and which are characterized by the type of language that they accept as input (English, German, French, etc.), by their degree of dependence on domain and applications, by the type of analysis carried out, and by their performance.\footnote{See for example the proceedings of the Message Understanding Conferences organized by DARPA.}

5. Ambiguity measures with the LOLITA NLP system

In order to assess the ambiguity measures introduced in the previous section, we shall refer to the LOLITA NLP system. This is a domain and application independent NLP core system designed for use in realistic situations \[2\]. All the data (morphological, grammatical, semantic, pragmatic, etc) are stored in a large semantic net, a kind of conceptual graph \[7\], which is the knowledge base of the system. The version used to test the ambiguity measures is based on a net of around 150,000 nodes connected in hierarchies.\footnote{It is now being integrated with a dictionary, the Cambridge International Dictionary of English (http://www.books.lit/CAMBRIDGE/inner.htm), which will greatly increase its size.} LOLITA accepts input in English, although it also contains data for Spanish, Chinese and Italian. Natural Language documents are morphologically, syntactically and then semantically analysed. Semantic analysis assigns a part of graph to syntactic parse trees. This sub-graph is then passed to the pragmatic analysis, which before adding the new information checks if it is consistent with the rest of the semantic network.

5.1. Lexical ambiguity

In order to assess lexical ambiguity, there is a LOLITA command, \texttt{lc}, which shows all the meanings associated with a term contained in the net. For the term \texttt{bank}, for example, there are 13 different meanings, of which 7 are nouns and the other 6 are verbs. Thus we have: $\alpha$ (bank) = 13 (semantic ambiguity) and $\beta$(bank) = 2 (syntactic ambiguity). The high number of meanings is due to the type of representation adopted for LOLITA: the nodes represent concepts, so that there is, for example, a meaning of \texttt{bank} as \textit{financial institution} and one as \textit{building}. If the domain is fixed, for example by applications which concern businesses, one may assume that the meaning of \texttt{bank} as \textit{river bank}, for example, has very low or nil frequency and the semantic ambiguity decreases to 11.

The frequencies of the various meanings of a word is one of the controls associated with the corresponding nodes in the LOLITA semantic net. This control, called \textit{frequency}, may assume the following values: \textit{rare}, \textit{common use}, \textit{familiar}, where ‘familiar’ indicates the context in which the term is used. This means that, with the present state of the LOLITA net, evaluation of weighted semantic ambiguity is not immediate. For example, \texttt{manager} has two meanings, both nouns, of which the first has a \textit{rare} frequency and the second is in \textit{common use}, corresponding to $\alpha$=2 e $\beta$=1=\texttt{familiar}. The representation in the notation introduced is as follows:

\[
\text{manager} \equiv \{<\text{director, name, common_use}>,
<\text{coacher, name, rare}>\}
\]

Measuring the weighted semantic ambiguity requires introduction of a numerical scale and the use of a logarithmic function which takes account of the fact that terms like manager have less ambiguity than ones with two meanings of the same frequency.

By way of an example, we give an extract from the output from the LOLITA \texttt{lc} command for \texttt{manager}. When a term is specified by the analyst, this command retrieves the concepts corresponding to the main controls, together with the nodes to which it is hierarchically connected (figure 1).

\begin{verbatim}
command: lc
controls about: manager
meanings:
0
50360 : Managers. (=> Athletes. : 4257
= Coaches. : 67881, Trainers. : 64813)
rank: universal
family: job
type: entity
(...)
1
50361 : Manager. (=> Administrators. : 29657
= Managing directors. : 123814, Director. : 1276)
rank: universal
family: job
\end{verbatim}
Figure 1 – Meanings of the term manager (lc command)

This information can be used by the analyst to identify more precise terms in the presence of high values of semantic ambiguity.

5.2. Sentence ambiguity

The ambiguity of a sentence can be obtained by examining the output of the LOLITA parsing command *pasbr*. The effect of this command on an input sentence is to produce all the syntactic trees with two representations, the first of which also shows the semantic ambiguity of the terms of the sentence, while the second enables understanding of the interpretation associated with the tree. Figure 2 gives the output for the sentence “The ABC company sold the bank in London”.

command: pasbr
information:
The ABC company sold the bank in London.
0
sen
detph
definedDet THE
relprepcl
commn COMPANY [Sing,Neutral,Per3] * 6
prepp_relate
prep RELATE_
full_propernoun_simple
propernoun_not_comp ABC
transvp
verb SELL [Past] * 3
detph
definedDet THE
relprepcl
commn BANK [Sing,Neutral,Per3] * 6
prepp
preppNormRel IN
full_propernoun_simple
propernoun_indiv LONDON * 3
((The (company ABC)) (sold (the (bank)) (in London)))

Figure 2 – Example of output from the *pasbr* command

It will be noted that there are two parsing trees, the bracketing of which highlights that the ambiguity is due to the fact that the sentence can be interpreted as stating both that the sale took place in London and that the bank sold was in London. The number of possible meanings of the ambiguous terms is indicated alongside them in the parsing tree. It will also be noted that their semantic ambiguity is less than that which arises when considering the isolated term ($\alpha_k \leq \alpha_t$), given that some of the meanings have roles which are not acceptable for the parsing of the sentence. For example, *bank* has 6 meanings with respect to the 13 possible because only 6 are common nouns.

A notation useful to the analyst might be as follows (figure 3):

| The ABC company (6) sold (3) the bank (6) in London (3) $^{12}$.
| $\tau_1$: ((The (company ABC)) (sold (the (bank) (in London))))
| $p_1 = -5$
| $\alpha_t = 6 \times 3 \times 6 \times 3 = 324$

| The ABC company (6) sold (3) the bank (6) in London (3).
| $\tau_2$: ((The (company ABC)) ((sold (the bank)) (in London)))
| $p_2 = -5$
| $\alpha_t = 6 \times 3 \times 6 \times 3 = 324$

Figure 3 – Example of sentence ambiguity notation

Here the penalties are obtained using the LOLITA *tp* command. This command has the same effect as the parsing command but it yields more detailed output.

---

$^{12}$ London has three possible meanings corresponding to different cities, in UK, Kentucky, Ohio.
than that shown in figure 2. Penalties are values used by LOLITA to order the trees (the higher the value, the worse the tree). They are divided into four groups of increasing seriousness:

1. Usage penalties, to indicate less common constructs\(^{13}\) (\(p < 30\)).
2. Minor feature clashes, e.g. wrong concordance (\(30 < p < 100\)).
3. Major feature clashes, e.g. (apparent) dative or infinitive use of inappropriate verbs (\(100 < p < 1000\)).
4. Structural problems, e.g. missing or repeated parts of speech (\(p > 1000\)).

These penalties are measures of the effort made by the NLP system, and therefore of the effort required to interpret sentences. As regards the measure of syntactic ambiguity, if the parsing of a sentence produces only trees with penalties higher than 1000, the analyst should adjust the requirements text. For penalties less than 1000, if trees with equal penalties are obtained for a sentence, these are essential ambiguities which even a person is unable to resolve entirely: this is what happens when both interpretations of the example sentence are possible.\(^{14}\) In situations like this following a signal from the system the analyst should intervene in order to resolve the ambiguity. As regards semantic ambiguity, this has been calculated for each tree as the product of the ambiguity of the terms contained in it, and for a sentence as an average of the possible trees. On this basis, a scheme for measuring the ambiguity of a sentence using LOLITA is the following:

\[
\gamma = \frac{1}{l} \sum_i (\Pi(\alpha_i)) \\
\delta = n_i \\
\delta^* = \frac{1}{l} \sum_p p_i
\]

Hence, to summarize, we have the following ambiguity values for the sentence analysed: \(\gamma = 324\); \(\delta = 2\); \(\delta^* = -5\).

It should be stressed that LOLITA uses the output from the parsing phase to interpret the content of the text (\(seR\) command), and in that phase it also uses pragmatic and contextual information to generate a description of the information contained in each sentence. For the example sentence, it thus highlights that a bank is sold, although, as often happens, it was probably meant that it was one of the banks owned by ABC. In this case, the analyst must modify the text by changing the article the into the determiner its (figure 4).

\[\text{(…)}\]
\[\text{universal:} \]
\[\text{event - 7688 - rank: universal \ (happen_)}\]
\[\text{subject:} \]
\[\text{ABC - 138030 - rank: named individual - family: human organisation}\]
\[\text{action:} \]
\[\text{sell - 78749 - object:} \]
\[\text{bank - 138047 - rank: individual - family: organisation}\]
\[\text{(…)}\]
\[\text{**********************************}\]
\[\text{ABC sold a bank.}\]

\[\text{Figure 4 – Extract from the output produced by the seR}\]
\[\text{command (c 10\%)}\]

The following comments can be made concerning evaluation of the ambiguity measures that can be obtained using LOLITA:

- Semantic ambiguity: the LOLOITA semantic net is larger than many NLP systems. It is possible to use other language corpora, but the main advantage of LOLITA is that it can be used to obtain the ambiguity of terms within the individual parsing trees.

- Syntactic ambiguity: given the difficulty of ordering the parsing trees, the correct interpretation does not always appear in the first positions. An analysis conducted in this regard yielded 56\% in the first three positions.\(^{15}\) This requires further effort by the analyst. It is therefore essential that the results of the analysis should be presented in a way that facilitates the identification of ambiguity.

\[\text{**********************************}\]

\(^{13}\) E.g., if a word has both a noun and an adjective form, and is used in apposition to another noun, the adjective form is usually preferred, as in 'human behaviour'.

\(^{14}\) On the basis of knowledge about the company in question and of language use there might be a preference for the second interpretation.

\(^{15}\) Better results are expected from the new version of LOLITA, Concepts, which is about to be marketed.
Evaluation of parsing with LOLITA, data are available which have been obtained from analysis of texts of differing quality and therefore characterized by various levels of ambiguity, and where 20% of the sentences have only 1 parse tree, 25% have 2-9 parses, and the rest have more (with around 3% no parses and 8% time out). One possible use of data of this kind is establishing an acceptable ambiguity threshold based on the importance of the documents concerned, and which in the case of requirements need not be absolute. Fixing reasonable thresholds for overall ambiguity requires the obtaining of data of this kind for classes of texts.

6. Conclusions

This paper has introduced semantic and structural ambiguity measures for the purpose of obtaining information to support the collection and analysis of requirements in natural language. These indices can be obtained using a NLP system, and in particular, as the paper has shown, the NLP system LOLITA. The results of this first analysis are encouraging. They show that it is possible to identify and signal the ambiguities present in a requirements document. The measures introduced bear out the principles suggested by approaches which provide support rules and schemes for the writing of requirements. In particular, an NLP system like LOLITA may provide the analyst with useful information on how to reduce semantic ambiguity with more precise terms by descending the hierarchy of concepts, and on the origin and nature of structural ambiguity. Further investigation is required to design an interface which presents this information in a manner that efficaciously supports requirements analysis.

References
11. Miles Osborne, C K MacNish, Processing Natural Language Software Requirement Specifications, Proc. of the Int. Conf. on Requirements Engineering 1996 (ICRE ’96)