AN ONTOLOGY-BASED APPROACH TO KNOWLEDGE ACQUISITION FROM TEXT

Ricardo Sánchez Carreño
Jesualdo *Fernández Breis*
Rodrigo *Martinez* Béjar
Departamento de Informática. Inteligencia Artificial y Electrónica
Universidad *de Murcia*

Pascual *Cantos Gómez*
Departamento de Filología Inglesa
Universidad de Murcia

ABSTRACT

Knowledge extraction from texts/corpora can simplify the knowledge acquisition process as the participation of knowledge engineers would not be required and systems would acquire knowledge directly from experts. This work presents a system that uses techniques from knowledge acquisition and natural language recognition research areas for acquiring knowledge from text. The knowledge acquisition process, which is represented by means of ontologies, is described in this paper as well as the validation of the tool in a specific linguistic domain.

KEY WORDS: *ontologies, knowledge acquisition, natural languages*

RESUMEN

Lo extracción de conocimiento a partir de textos corpus lingüísticos puede simplificar el proceso de adquisición de conocimento ya que no se precisará de ingenieros del conocimiento y los sistemas informáticos podrán adquirir conocimiento directamente de expertos. Este trabajo presenta un sistema que integra procedimientos propios de las áreas de adquisición de conocimiento y de reconocmiento de lenguajes naturales. Además de la descripción del proceso de adqusición de conocimento, representado mediante ontologías, se valida la herramienta implementada para un dominio lingüístico específico

PALABRAS CLAVE: *ontologías, adquisición de conocimiento, lenguajes naturales*

I. INTRODUCTION

Extracting knowledge directly from *natural language texts* is *an interesting and* challenging *task as it* might *help to extract knowledge* easily *and without the involvement of* knowledge *engineers.* In *addition. we are interested in tools* capable of both *extracting knowledge fronitext and interacting directly with experts* within specific linguistic domains.

This paper presents a technique for generating knowledge from text. combining techniques and approaches from knowledge *acquisition and natural language recognition (two* completely different *disciplines). Knowledge is input by experts into the* system. *specifying* where it *resides within the text* while *the* system *associates text and knowledge.* Both *language* and *knowledge can* repeatedly *appear* along *the text. Whenever this* situation *arises.* the *system has to make decisions concerning the nature of the language as it can refer to knowledge that* has *already* been *taken in by the system. This* work also addresses the usage *of associations between text and knowledge to derive knowledge from text* fragments.

The whole *process can be divided into two main* steps. *The* first *one* is *completely human-guided:* an *expert reviews a text. and knowledge is generated froni scratch. This* is *the search phase. The second one* is *setting a* context *(the language* chosen *for setting this technique* is English).

Amongst *the* many different features of *natural languages those that make them* particularly *difficult for understanding* and *processing are polysemy and ambiguity. In a* multi-*domain environment. a same word can refer to* different *things or knowledge units. This explains* partially *why techniques based* solely *on natural languages cannot be* perfect due *to tlie* intrinsic problems of *natural languages. The technique presented* here attempts *to* arrange *knowledge associated to a single* linguistic *expression. The system can make wrong knowledgr* associations but *the tool has* been designed *and* implemented *in* such *a way that the user can* modify *decisions* arrived at *by the tool whenever they are considered to be inappropriate or wrong.*

The basic idea of our approach is *that the system stores knowledge* found *by the expert in* order *to be* able *to* automatically *identify this knowledge* when it *appears thereafter. For* instance. if *tlie user* recognises *the expressioniword* "flue" *as a concept. whenever* it *reappears* in *the text. the system* will realise *that this* expression/word *has already* some *associated knowledge. presenting* it to *tlie user. who* will *then* have *to decide whether this knowledge* association is *correct or* not.

Knowledge has been *represented in this work by* means of ontologies. In *the literature. ontologies are* commonly explained *as* specifications of domain *knowledge conceptualisations (Van* Heijst *et al. 1997). Due to tlie* very nature of ontologies. *there* is *not a* unique (valid) way for defining *them* (Musen *1997). Moreover.* several different definitions have been historically *assigned to* ontology. Succinctly. an ontology *is* commonly *considered to be an* enumeration *of*

relevant concepts in an application area. as well as a definition of classes oí' concepts and relationships among these classes (Martínez-Béjar and Martín-Rubio 1997). A positive contribution to ontologies has to do with the possibility of studying their formal and mathematical properties (see Martínez-Béjar and Martin-Rubio 1997). In the present work we have used the operators proposed and formalised in Martínez-Béjar and Martin-Rubio (1997) and iinpleinented them to build a domain ontology.

I.1. Knowledge Acquisition

One of the most popular knowledge elicitation technique is the *interview* with domain experts. Knowledge engineers usually. first. get knowledge from experts by interviewing them and. then. formalise that knowledge. using. for instance. ontologies.

However. there are several problems related to interviews. Interviews contain two phases: (1) knowledge engineers acquire knowledge by "becoming experts" and (3) the formalisation of the previously acquired knowledge. Consequently. the expert system built will never be as an expert as the domain expert. but as an expert as the knowledge engineer is.

Another flaw is derived from the imperfect nature of interviews. Performing interviews is usually slow. as it can lead to misunderstandings in the communication process between thr expert and the knowledge engineer. and there can always be mistakes in the knowledge formalisation process. Furthermore. according to Jackson (1990). interviews have a low knowledge per hour ratio. In order to overcome these drawbacks in knowledge acquisition. a possible solution might be *automation*: an attempt to reduce the human component in the knowledge acquisition process. The approach presented here aims at this automation tendency. removing interviews from the knowledge acquisition process: it is the domain expert that builds the ontology from text by means ofinteracting directly with the system.

1.3. Natural language recognition

Natural language recognition has traditionally been viewed as a purely linguistic issue. based mostly just on grammars. However. grammars present several problenis. For instance. they are unable to handle common key natural language properties such as ambiguity. imprecision. variability. etc. Automatic data transferring from natural language sources to knowledge base entries is qualitatively quite poor (Sánchez-Carreño 1999). too. An example showing a dependency grammar-based approach for recognising natural language in medical domains is presented in Steimann (1998). This author recognises the difficulty of constructing such

grammars and presents the limitations and problems grammars constructed this way offer. A possible solution for solving this grammar constraint is enabling experts to decide whenever unclear or difficult cases arise. For instance. in Van Heijst et al. (1998). the authors proposed an NLA algorithm in which the expert is asked each time ambiguity is encountered.

Consequently. we believe *non-automated* approaches to be more appropriate for processing natural language. introducing a human agent that acts in some parts ol'the process to solve specific natural language problems. Jn Schmidt and Wetter (1998). the authors state that the underlying reason for having problems and deficiencies with direct data transferring has to do with the assumptions a text writer/author makes about herihis reader's knowledge. If the receiver does not satis./ those assumptions then any of the following three elements need improvement: the producer. the receiver and/or the channel.

The improvement ofthe producer is not a good option whenever knowledge is being acquired from text. There are some reasons that support this judgement. Firstly. text authors will not rewrite texts keeping in mind that these are to be read by an automatic apent. Secondly. some unsuccessful experiments have revealed that the fact that experts thinking in what the system might understand. continuously interferes with what (s)he wants to communicate. The consequence is that the quality oi'the acquisition process is drastically reduced.

Improving the receiver implies implementing tools able to understand the language used by experts. However. as already mentioned. the efforts of producing automatic tools able to understand natural languages have not provided good enough results yet.

Finally. the third option is improving the channel. This has been the option chosen here. The channel can be improved by introducing a mediator between the text and the system. The system atternpts to build an ontology by itself and the expert supervises this process. havinp the possibility of correcting mistakes made by the system.

In what follows. we shall give an overview of the approach (Section 2) and explain the search phase ol'the approach (Section 3). In Section 4. the other main phase of the method. namely. setting a context. is presented. The phases described in the sections 3 and 4 are then compared in Section 5. Section 6 describes the tool itself. Section 7 presents the benefits ol'the approach and in Section 8 the conclusions are drawn.

11. AN OVERVIEW OF TIJE APPROACH

The aim ofthis work is to describe a method for extracting knowledge from natural language texts. More specifically. building an ontology from a given *text*. Texts can be ven long -to facilitate their processing they can be divided into *fragments*-. covering a domain or *task*. that is. its content is about a specific application domain. The ontology is built by an *expert* who must have some expertise on the specific task described in the text. Furthermore. the experts interact with the system by means of the text and its implicitely stated task.

Knowledge contained in text is said to reside inside it. Ontologies divide knowledge into classes such as concepts. attributes. relationships. rules. etc. These knowledge entities (i.e.. classes) can explicitly appear in the text although sometimes they are only referred to implicitly. The process proposed here attempts to Gnd precisely this explicit knowledge occurring in the text.

She starting point is an empty *knowledge base*. In this initial stage the system is unable to find any existing knowledge in the text and it is the expert's task to input it. Additionally. experts do not just find knowledge in a single fragment. they might also need to identify expressions derived from specific knowledge.

The expert tries to identify all knowledge entities found in the fragment. telling the system the *expressions* in which they appear. These expression-knowledge association schemas are then stored by the system in order to re-use them in forthcoming new knowledge Gndings. The expert only has to identify these associations once. thereafter the system will proceed automatically and the expert will only have to confirm the associations offered or found by the system. In principle. systems with large knowledge bases need little or less expert intervention. concentrating primarily on dividing texts into fragments and confirming the system'sproposals. Unfortunately. the process is not that simple. as while the system searches for fragments or expressions with associated knowledge. words with associated knowledge can appear in different forms (types): plural or singular. replaced by a pronoun or with a specific form (verbal tense. etc.).

In large knowledge bases. we might iind expressions with several different types of associated knowledge. e.g. polysemous words having multiple knowledge associations for the same expression. Similarly. we might find knowledge fragments referring to other pieces. For instance. attributes do not exist on their own. they belong to a concept. A relationship implies the existence of at least two concepts. Thus. the system has to identify knowledge in the fragment as well as knowledge referred to it. This process brings along some intrinsic problems: (1) searching ior expressions in a fragment. (2) deciding what to do when an expression has more than one knowledge association in the knowledge base and (3) identifying knowledge referred to by non-concepts. The iirst two problems are tackled in the *search phase* whereas the third one is studied in the *context setting phase*.

These two phases can be approached from different points of view. Here. we shall t n to overcome the above problems applying the solutions proposed by Musen (1997). Musen does not deal with implicit knowledge. and it is the expert's task to identify implicit knowledge in every fragment and not the system`s one.

III. THE SEARCII PHASE

The first goal of this phase is to find eapressions with associaied knowledge in the knowledge base. Next. whenever an expression has more than one knowledge association. its potential association has to be decidedichosen. Given that texts can be too large. this search procedure is performed within each fragment just. The search process is quite simple and can be formalized with the following algorithm:

```
Fragment expressions with associated knowledge (FE) = O
While there are non-analysed words in the current fragment do:
        current_word = next non analysed word;
        For all the expressions in the knowledge base (KB) which are similar to current_word.
                If the expression is acceptable then
                        Obtain the associated knowledge the expression has in the KB;
                        ordered_knowledge_list = Sort this knowledge;
                        fragment_expression =
                                new expression that matches the KB expression;
                        fragment_expression.possible_knowledge =
                                                        ordered_knowledge_list;
                        FE = FE U{ fragment_expression};
                End If.
        End For.
End While.
```

The result is a list containing all expressions of the fragment already included in the knowledge base, Nen expressions are not associated to any knowledge yet and are stored in a knowledge list of "possible candidates". This list holds all potential knowledge associations for the expression found in the knowledge base. After the "context setting" phase. this list will be displayed to the user. offering her/him the possibility of choosing a possible knowledge from the list.

Obviously. it might also happen that no good options are found. In that case. it is the user who has to provide andior define new knowledge units associaied to already existing expressions or to new ones. Moreover. the user has also the choice to simply ignore expressions.

The various functions underlined in the above algorithm are now described:

Similar

This function is in charge of identifying which word/expression in the current test fragment is similar to those in the knowledge base. The simplest case would be an "equal" function. Nevertheless. this function cannot deal with compound expressions. therefore a function of the type "isPrefix" is needed. It would also be desirable that these functions could deal with different types associated to the same lemma (for instance. identifying an expression/word. such as "swam" as identical to a current fragment containing the word "swims"; this could partially be solved by means of a lemmatizer and/or a part-of-speech tagger). Musen (1997) uses the

"isPrefix" function. whenever there is a word in the current fragment "similar" to an expression in the knowledge base that starts with the current word.

Acceptablc

This Sunction is an extension of the "similar" function. As the "similar" Sunction can be very permissive. the "acceptable" function is introduced in order to determine whether the current word and a similar existing expression are not just "similar by chance". The "isPrefix" function has an important drawback: if the current woi-d is the article "a". any expression starting with "a". such as "assurance". "added value". "a hundred" or "advert" will be considered to be similar. The "acceptable" function limits the number of acceptable options amongst the similar ones. It has been designed having strong requirements. An expression existing in the database is acceptable if it occurs in the current fragment.

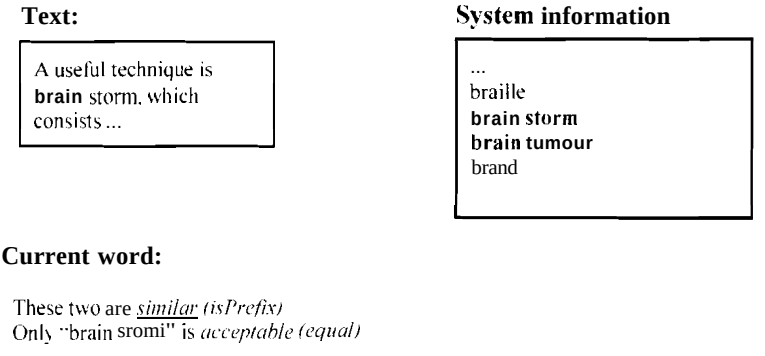

Figure 1. Combining the "similar" and "acceptable" functions

Let us now illustrate this with an example that combines the "similar" and "acceptable" functions (see Figure 1). Let us suppose that the current word is "brain". When looking up the knowledge base. two expressions are identified as similar: "brain storm" and "brain tumour". The following step is to look at the current fragment and check whether any of the previous expressions could be accepted. If the word that follows "brain" in the current fragment is "storm". then the first expression will be considered acceptable whereas if the following word is "tumour". the second one will be accepted. Else. none of them will be considered as acceptable. Constraining this way reduces the applicability of one of the benefits of the similar function: identifying words with the same root but different suffix as similar.

Creating new expressions

Current words/expressions in a text fragment are always single coiistituents. However. database expressions can contain more than one word. If an expression is acceptable. then the current Sragnient uill contain all tlie words of the database expression. That is. the current word need to be enlarged to cover all the words ofthe database expression. creating a new object containing all the words.

Obtaining associated knowledge

The correspondence between expressions recognised by the expert and their associated knowledge is stored in the database. After obtainiiig an expression that nieets certain requirements (similar and acceptable). knowledge associated to that expression is searched in the database (additionally. also in other texts and in tests from other experts). Whenever different association possibilities in the database exist. these are sorted and displayed to the expert. An exaniple is presented in Figure 2.

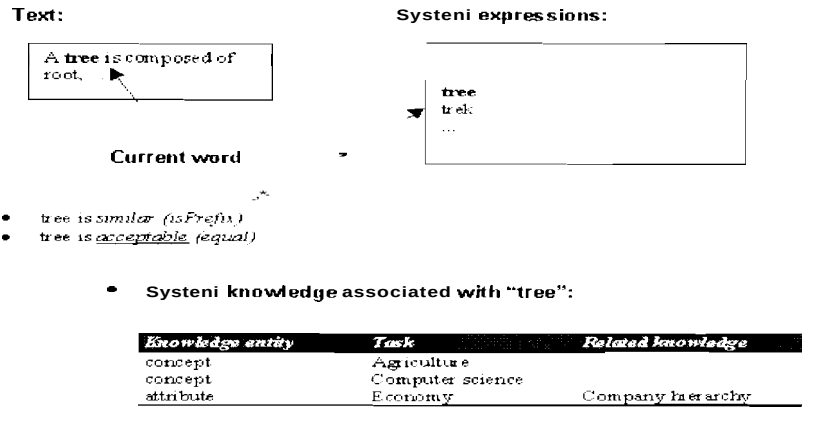

Figure 2. An example of ohtaining the associated knowledge

Sorting the knowledge

According to the above description of the search phase. we stated that there niight be instances where we niight get a set of possible associated knowledge for a single expression. The existence of more than one possibility for associating knowledge is likely due to the following reasons:

- Doniain dependency: tlie different meanings given to a term can vary according to the doniain in which it is used. In a doniain such as physics. tlie expression/word "velocity" is associated to a concept whereas in other domains it becomes an attribute.
- Person dependency: it is likely that various experts assign different meanings to the same

expression. For instance, in philosophy, expressions such as "idea" are restricted to certain authors, assigning very specific meanings to it.

- Spatial location: if an expression has been used recently with a speciiic meaning and the same expression appears again, then it is likely to have the sanie meaning.

Whenever different possibilities are considered as inferred knomledge from an expression, the systein orders them according to the previous three factors. Amongst those factors, tlie spatial location interacts in two different ways. The system considers whether an expression has already been used in the same text file and/or in the current fragment (this case is given tlie highest priority). The various sorting criteria are:

Know ledge that has been recognised by the same expert (person dependency), ior the same type of domain (domain dependency), in the same fragment and test (spatial location).
Knowledge recognised by the same expert for the same type of domain and text.
Knowledge recognised by a different expert for the same type of domain, text and fragment.
Knowledge recognised by the sanie expert for a different type of domain in the same text and fragment.
Knowledge recognised by a different expert for a different type of domain in the same text and fragment.
Knowledge recognised by a different expert ior the same type of domain and text.
Knowledge recognised by the same expert for a different type of domain and in the same text.
Know ledge recognised by a different expert for a different type of domain and in the same test.
Knowledge recognised by the same expert ior the same type of domain but in a different text.
Knowledge recognised by a different expert for the same type of domain in a different text.
Know ledge recognised by the same expert Sor a dif'ferent type ofdoniain and a different text.
Know ledge recognised by a different expert for a dif'ferent type of domainand in a different test.

After sorting out knowledge, the search phase ends. At this point, the systeni would have processed the current fragment and expressions present in its database. Additionally, inferred knowledge found would have been sorted out according to the criteria above in a attempt to overcome ambiguity.

IV. SETTING A CONTEXT

Once the searcli phase has been carried out. the system will have a list of expressions with associated knowledge. However. the system's task has not Iinished yet. only if the inferred knowledge is a concept. else, that is. if the inferred knowledpe is a different knowledge entity some operatioiis still need to be perfornied.

IV.1. Attributes

In English. attributes usually follow a concept. This property is used by the system to look for concepts attrihutes belong to. Therefore. current fragments are processed backwards froni the current expression on until an expression which is labelled as a concept is found. For exaniple: "... *the high system* ***cost*** *is the cause of problems like ...*". If expression "cost" appears in the database linked to an attribute. then it will be recognised in the search phase. Next. tlie prograni will search for the most left-nearby concept of "cost". The systeni looks for expressions for which knowledge has already been inferred in the current working session. In this way. if the user had inferred the concept. say. "theory" from the expression "system". the systeni would find it and would immediately associate "theory" to the attribute "cost". It is also likely to find no expression M ith associated knowledge on the left of the attribute. In this case. the system would search in the database for the corresponding concept.

In practice. if an attribute is far from its corresponding concept. then it is unlikely to be associated with it. Moreover. it is also possible that the systeni finds during the search phase an espression nearby an attribute for which a concept in the database already exists. Therelbre. after checking a predetermined number of expressions for which knowledge has been inferred by the user. if no concept has been inferred from them. the system searches for expressions that were found during the search process. looking for an expression with an inferred concept in It.

Another heuristics is applied every time the system is looking for a concept and a different attribute is found. In these cases. the systeni can use tlie concept associated to tlie second attribute. This heuristics is not used with those expressions obtained in the search phase due to their lack of stability and reliability.

IV.2. Values

Let us now consider a further example: "... *because of the* ***low*** *lithium atomic number ...*". It is difficult to know where the attribute is going to appear in a text. although it often appears on

the right handside of its value. This means that, normally, there will not be many expressions on the right handside of "low" with associated knowledge. This implies that the system must be guided by means of the expressions found during the search phase. Let us suppose that the user has just started with a fragment and knowledge has only been associated to expressions on the left handside of "low". If the system searches on the right handside of "low" for an expression with an attribute inferred by the user, the system will not be able to find it. Only if the tool had been previously used in a chemistry domain and, consequently, the concept "lithium" and the attribute "atomic number" already existed in the database. In this instance, the system will find the espression "atomic number" in the search phase, and the value "low" will be associated to the attribute "atomic number". The system will attempt to set the attribute "atomic number" in a context and associate the concept "lithium".

In this particular case, the context for the expression "low" proposed by the system would be: "*lithium.atomic number.low*". This type of results are frequently obtained in practice. As well as with concepts, attributes are not the unique elements considered when providing contexts for values. If the system finds any expression with any inferred value while searching on the right handside of the value, the attribute associated to the latter value is assigned to the value in process. As well as with concepts, if after analysing a pre-determined number of expressions, for which the user has associated knowledge, nothing is found, the system searches amongst the expressions found in the search phase.

IV.3 Relations

Relations are assumed to be a binary. That is, two elements must be found. Let us consider the example: "*Drugs affect human behaviour*". This type of structure is most frequent between relations: one of the candidates is on the left of the expression, inferring the relation, and the other one on the right handside. The system searches for expressions with inferred knowledge on the left and right handside and candidates are selected according to various criteria:

- If the current expression is associated to a relation of the type "is-a" or "part-of", any ontological category can be chosen as a candidate as these relations can only exist between concepts. Therefore, the system searches for two concepts, one on the left and one on the right handside of the current expression.
- It is very rare that any of the candidates of a relation is a value (the system is designed to ignore values).
- If an attribute is found, the process of searching for a related concept is the same as the one described to provide a context for attributes.
- The search process is similar to the one described in previous sections. Candidates are searched for (1) in a pre-determined number of expressions for which the user has

associated *knowledge.* (2) *in tlie expressions obtained in the search phase and* finally **(3)** *in the expressions* of *the* user.

V. COMPARING *THE TWO PHASES*

The search phase is *a* semantic *process:* words *are* selected from *a text and their meanings are* looked up *in* a *database. The* database and *the knowledge contained are prime* for correctly associating *knowledge* to *expressions. On the other* hand. *context* setting is *a* syntactic *process:* this *phase* starts *once* all *knowledge has* been found *in tlie database. Furthermore. this knowledge* will only *be used once a context* is *found. derived lrom the f'ormer. Concepts are associated to attributes and participants* to *relations through a* simple *linear search* method.

The search phase is language-independent. *That is. knowledge is searched in the* database *and the* meaning of words *is looked up in a* dictionary: *Both knowledge and dictionary share the* same *structure independently of the language* used. On *the other* hand. providing *a context for a knowledge entity is language-dependent. Concepts are searched for* on *tlie* left *handside* of *the attributes as this is* how they normally *appear in* English. *and attributes are searched* for *on* the right handside of the *values.* If the *language chosen had* been *Spanish. then* concepts *would* have been *searched* for on *the* right *handside of the attributes and attributes on the* left *handside* of the *values.*

VI. DESCRIBING *THE TOOL*

A tool based *on the* approach *described* above *has* been designed *and implemented* for acquiring *knouledge* from *texts (text needs to be* specified *in a text* file. i.e.. in *ASCII* format). *Text length* is irrelevant *as it can be splitted into different* minor fragments. *Test* samples might *belong to one* or more specific domains or *tasks. The* distinction of domains is important *as* meanings of words *depend* heavily *on the* domain they *are* used. *The final user ol'the tool* is an *expert.* Each *expert* is *acquainted with knowledge* of one *or more domains. The* system also *accounts* for *tlie* associations *betueen experts and tasks.*

The knowledge acquisition process is made *in* sessions. An *expert* with *knowledge on a* specific *task* specifies *the* file to work with *and a* new session is *created associated* to *this expert.* task *and lile. While* processing *the* fragments. *tlie expert* finds or recognises *knowledge. This* knowledge *can appear* explicitly or implicitly *in tlie* fragment. If the knowledge appears *explicitly in tlie* fragment. *then* the *expert has* to identify the *espression* in *which this*

knowledge appears. associating expressions to knowledge or inferring knowledge from expressions. Recall that xpressions and knowledge do not necessarily coincide.

This tool has two distinct working modes: (1) the query mode and (3) the maintenance one. In the maintenance mode. users are provided with the full functionality of the tool (adding nem experts and tasks. associating experts to tasks. saving work/session in the database. loading previously saved work/session(s) etc.). The query niode has a reduced functionality. Tlie user cannot perlorni nianagement activities nor save work/sessions in the database. Other differences between both modes include: (1) in the maintenance mode. the user inserts knowledge with the help ofthe tool: the systeni proposes knowledge to the user by making use of natural language recognition techniques: and (3) in the query mode. the user cannot insert new knouledge as ontologies are built automatically.

It must be pointed out that the user can neither input knowledge into the system. select the expert. the task nor the text to be recognised. Else. the system will "understand" the user to be an expert. believing that the task corresponds to a domain from which knowledge has been previously acquired.

The systeni is able to infer concepts. attributes. values and relations. however. axioms cannot be automatically inferred. The main problem with axioms is that the numbei of participant elements is unlimited. Which and how many participants are part of an axiom is jet unexplored. The quantity of axionis present in a text is not huge compared to the quantity of ontological categories. Under these circumstances. the system has been designed not to recognize axioms in texts. just concepts. attributes. values and relations. However. users can define those axioms they consider necessary or relevant for the application doinain.

VI.1. Ontologies in the tool

CommonKADS (Schreiber et al. 1998) uses ontologies as a way of structuring. sharing and reusing domain knowledge. In CommonKADS. six ontological categories are distinguished: concepts. attributes. values. instances. relations and expressions (axioms). In this tool. only five of these categories are used. namely:

- Concept: it represents a class of objects in the domain.
- Attribute: concept's property.
- Value: it is detined Sor an attribute. For instance. length has a numeric value whereas colours are enumerated. The elements oí' the domains are the possible values attributes can take. The tool is oriented to cope with qualitative values.

- Relations: relations in a domain ontology play the same role as in a relation/entity model. although some constraints have been imposed. In this tool. relations are binary and are predefined:
  1. IS-A: this taxonomic relation allows for establishing conceptual hierarchies. For example: *A man is a human being.*
  2. PART-OF: this mereological relation indicates that a concept is comprised of other ones. For example: ***The*** *engine is part of the car.*
  3. ASSOCIATION: whatever relation between two concepts that is neither taxonomic nor mereological. For example: *Hair colour is related to skin colour.*
  4. INFLUENCE: association relation in which a concept can influence the existence of another concept.

The taxonomic and inereological relations only exist between two concepts. The remaining relations can exist between whatever two ontological categories although a relation cannot participate of another relation.

- Axioms: An axiom is a domain rule that includes a relational operator. For instance. Force = mass * acceleration.

In this tool. the ontologies are shown as trees with three branches: one for concepts. one for relations and another for axioms (see Figure 3).

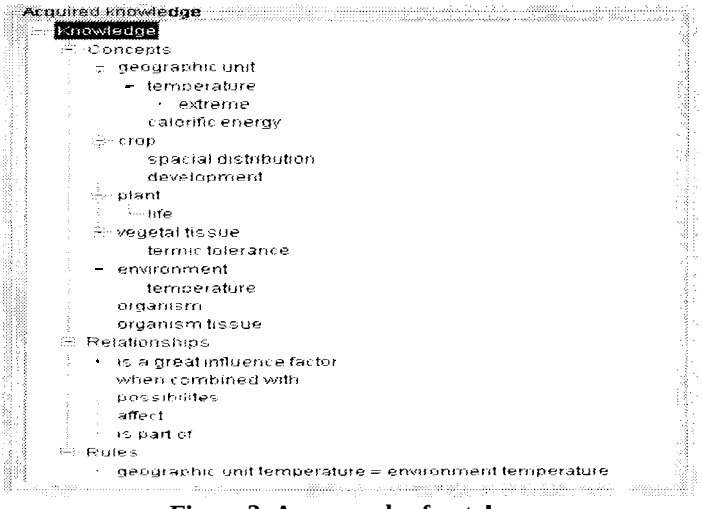

Figure 3. An example of ontology

In Figure 1. axioms are iepresented as branclies of the "rules" node. Each concept has branclies for its attributes and each attribute has branches for its values. The relations are branches of the "relationships" node. The instances of the relations are displayed on the right handside of the screen (Figure 4).

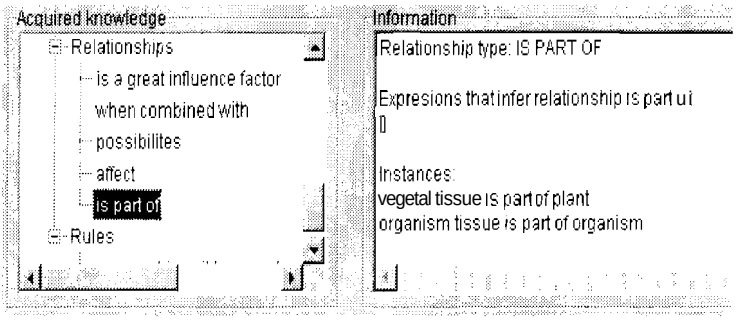

Figure 3. Relationships

Figure 5 shows a suggestion made by the system. In it. a relationship between the attribute "calorific energy" of the concept "geographic unit" and the concept "plant" is suggested. The user has the possibility of modifying it.

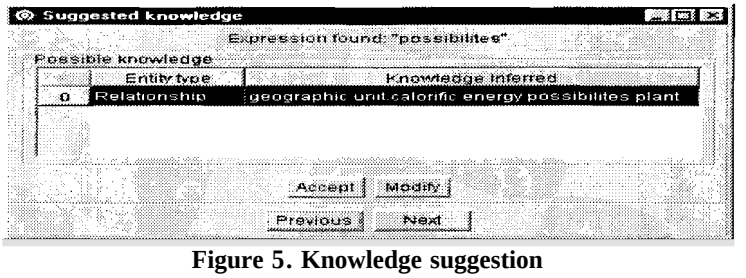

Figure 5. Knowledge suggestion

Figure 6 shows the screen that allows modifying the relation inferred by the system. The participants and the name of the relation can be chosen.

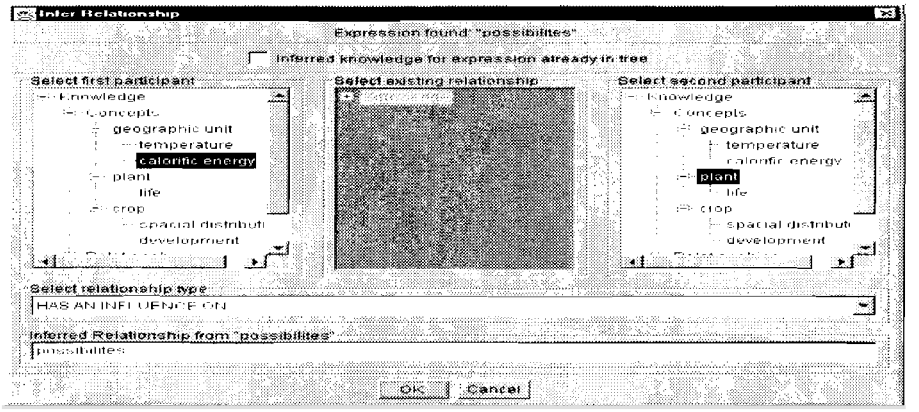

Figure 6. Modifying the relation

VI.2. Validation

The tool has been validated for a specific domain. namely agriculture. Figure 7 shows a text fragment iised for validating the tool. Figure 8 displays an analysed fragment of the test belonging to the studied application domain. The screen shows some acquired knowledge. We can see. for instance. that from the fragment and the acquired knowledge depicted in Figure 8. "calorific enrrgy" has been inferred as an attribute of the concept "geographic unit" and the relationship "calorific energy possibilities plant life" has been inferred from the text chunk "*when combined with calorific energy -light hours-, possibilities plan life*".

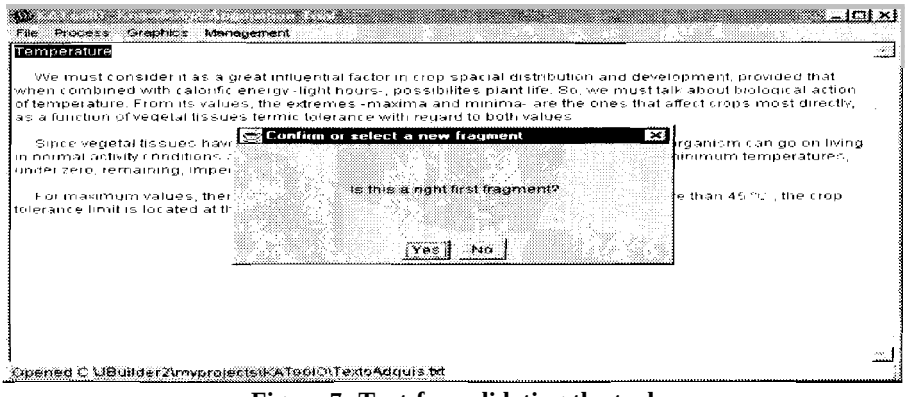

Figure 7. Text for validating the tool

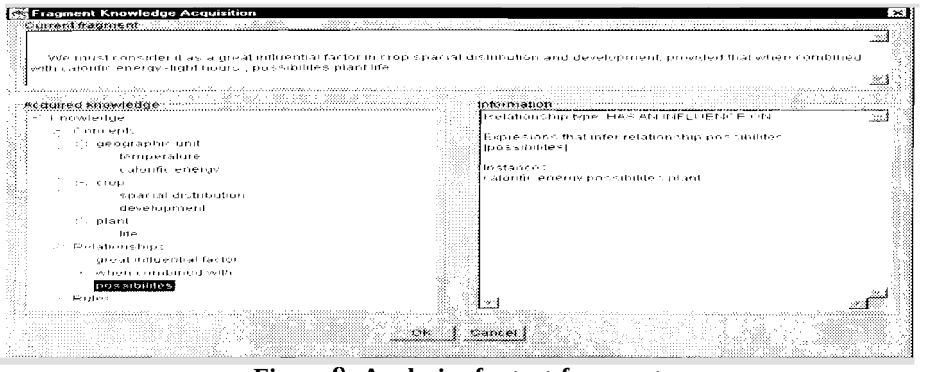

Figure 8. Analysis of a text fragment

Figure 9 shows a learning diagram and several knowledge options that can be selected. The user can view the learning diagram curve for the ontological entities discussed here (concepts. attributes. relations. rules and values) relative to text fragments analysed. Additionally. the user can also decide whether (s)he wants to see the knowledge associations found/identified by the user. by the systeni or by both.

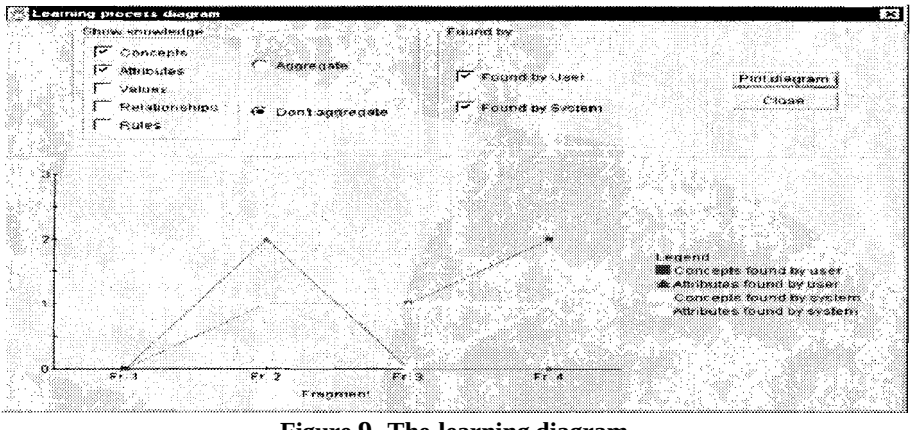

Figure 9. The learning diagram

VII. ADVANTAGES OF THE APPROACH

We are confident that this approach of recognising natural language offers some advantages with respect to pure linguistic methods as:

- Ambiguity is taken into account: the meaning of an expression/word does not only depend on its syntactic function/relation but also on the domain it is used.
- Another two types of ambiguity are considered. too: person dependency and spatial location.
- Rhetoric is not considered in our approach as the database only stores those expressions for which some associates knowledpe exists.
- A strategy for identifying implicit knowledge is included in the approach. allowing the user to add it.
- The system is incremental and automatic. It contrasts with grammars. whose rules need to be introduced and cannot be modified. Our system can be distributed with an empty database as it will incorporate knowledge from scratch and can be used for recognising natural language thereatter. If new scientific disciplines or new expressions appear or if an expression changes its meaning. the system could be easily adapted to these new requirements/conditions.
- Simplicity: the design of a grammar for recognising natural language sentences for multiple domains is very complex. but designing it for creating an ontology during the recognising process is very difficult. Our approach is not perfect but its results evidence a positive efficiency/complexity ratio.

VIII. CONCLUSIONS

In this work. a system for extracting knowledge from natural language text has been presented. The approach described combines different techniques conciliating two distinct research areas. namely. knowledge acquisition and natural language recognition. The tool offers a friendly and intuitive interface. different working modes and handles both implicit and explicit knowledge. The system suggests/aids the user. using the knowledge previously acquired. The tool itself satisfies many requirements and we can conclude that it seems adequate Ior acquiring knowledge. Furthermore. the techniques for acquiring natural language presented in this work offer a different and interesting method for using natural language for knowledge acquisition.

In Hahn and Schnattinger (1997). concepts are acquired from natural language texts. although the approach and thr way in which knowledge is structured are different. These authors used a distinct terminology (concepts. roles. individuals and axioms). in contrast to the one we used for our knowledge entities (concepts. attributes. relations. rules and values). For Hahn and Schnattinger. the concept acquisition process comprises three parts: (1) generating quality labels

Ior hypotheses. (2) estimating the credibility of the hypotheses and (3) computing the order of preference of the hypotheses. However, in our approach. the system's suggestions can be viewed as hypotheses that can be accepted or rejected by the user. Jones and Paton's (1997) approach for acquiring causal knowledge from scientific theories is presented and we are confident to have shown that our approach can be easily adapted to new requirements.

Dividing the acquisition process into two phases offers the possibility of using this approach with different languages. not only English. although some considerations should be made regarding both phases. The search phase works smooth and it is difficult to improve it. Some improvements can be proposed concerning the knowledge's sorting strategy. such as giving greater weight to frequently inferred knowledge. In addition to this. the "similar" and "acceptable" functions can be modified. although the kernel of the algorithm should remain unmodilied. that is. dictionary look-up. The "setting a context" phase. however. can be clearly improved since it works correctly in cases that meet the syntactic assumptions initially made but less well in complex situations. Clear improvements can be proposed for further work related to the "setting a context" phase. trying to overcome some intrinsic problems/difficulties related to the natural languages:

- Attributes: sometimes. the syntactic form "attribute of the concept" such as in "due to the weight of the table" appears in sentences. where the attribute precedes the concept. These expressions are easily recognisable due to the presence of "of the". Therefore. an extra checking can be added for managing these situations.
- Values: sometimes. the attribute does not appear explicitly. For instance. in "..a big red car..". two attributes appear implicitly. namely. "size" and "colour". If there is no expression either on the right of those values. among those expressions with associated knowledge. or among those expressions Iound in the search phase from which an attribute can be inferred. the system will search in the database for attributes whose values were associated when input in the database. Therefore. an improvement could be: searching directly in the database whenever no close attribute is found.
- Relations: participants of a relation do not always appear next to its relation. For example. in the sentence "Sun makes life possible". the relation is not "makes" but "makes possible". therefore. "life". which is the second element of the relationappears embedded in the relation. These situations are difficult to solve. In the sentence "Temperature has an influence on plants development". the method employed in this mork for providing a context for arelation would identify "temperature" and "plants" as the participants of the relation. instead oí "temperature" and "development". A way of solving this problem would be to check whether the concept is directly followed by an attribute. In this case. it should be assumed that it is more likely that the attribute is the second participant and not the concept.
- Pronouns: these are not treated in this work. Pronouns should be set in a context before providing the context for knowledge found in the search phase so that they could play the

same role as concepts. attributes. relations or values represented by them. When providing a context for a pronoun. the system should only search for tlie closest expression with knowledge associated on the left of the pronoun as mostly pronouns refer to something that has already appeared in tlie text (*anaphora*). In the extract "..are tlie consequences of *brain tumour*. **It** also produces headaclie ...", the pronoun "It" refers to the expression "brain tumour" so tliat the association would be correct. **An** inlluence relation would be inferred from the same text between "brain tumour" and "headache". Naturally. there are cases in which a pronoun does not refer to the expression tliat appears just before it (e.g. *cataphora*). These cases are difficult to deal with. even for humans.

ACKNOWLEDGEMENTS

This work has been possible thanks to the financial support of the Fundación Séneca (Centro de Coordinación para la Investigación tlirough the Programa Séneca (FPI)).

REFERENCES

Hahn. U. & Scluiattinger. K. (1997). *An Empirical Evaluation of a System for Text Knowledge Acquisition*. In Proceedinps of the European Knowledge Acquisition Workshop. 129-144. Sant Feliu de Guixols. Spain.

Jackson. P. (1990). *Introduction to Expert Systems*. segunda edición. Addison-Wesley.

Jones. D.M. & Paton. R.C. (1997). *Acquisition of Conceptual Structure in Scientific Theory*. In Proceedinps of the European Knowledge Acquisition Workshop. 145-158. Sant Feliu de Guixols. Spain.

Martínez-Béjar. R.. Cadenas-Figueredo. J. M. & Rilartín-Rubio. F. (1997). *Fuzzy Logic in Landscape Assessment*. In Proceedings of the European Symposium on Intelligent Techniques. 234-238. Rari. Italia.

Martínez-Béjar. R. & Martín-Rubio. F. (1907). *A mathematical functions-based approach for analysing elicited knowledge*. In Proceedings o!' the Nintli International Conference on Software Engineering and Knowledge Engineering. Madrid.

Musen. M. A (1997). *Domain Ontologies in Software Engineering: Use of Protegé with the EON Architecture*. SMI Technical Report.

Musen. M. A (1998). *Modern Architectures for Intelligent Systems: Reusable Ontologies and Problem-Solving Methods*.. In C.G.Chute. Ed.. 1998 AMIA Annual Syniposium. Orlando. FL. 46-52.

O'Leary. D.E. (1997). Impediments in the use of explicit ontolopies for KBS development. *International Journal of Human-Computer Studies 46. 327-338.*

Sánchez-Carreño. R.I. (1999). *Herramienta para la construcción de ontologías*. University of Murcia (in Spanish).

Schmidt. G.& Wetter. T. (1996). Using Natural Language Sources in Model-based Knowledge Acquisition. In Data & Knowledge Engineering 26 (1998): 327-356.

Schreiber. A. T.. Akkermans. J. M.. Anjewierden. A. A.. De Hoog. R.. Shadbolt. N. R.. Van de Velde. W. & Wielinga. R. J. (1998). *CommonKADS. Engineering and Managing Knowledge. The CommonKADS Methodology*. Department of Social Science Informatics. University of Amsterdam.

Steimann. F. (1998). *Dependency grammar for medical language and concept representation*. Artificial Intelligence in Medicine 12:1 77-86. http://www.kbs.uni-hannover.de/~steimann/pubs.html

Tomita. M. (1986). *Efficient Parsing for Natural Language.* Kluwer Academic Publishers.

Van Heijst. G.. Schreiber. A. T.. & Wielinga. B.J. (1997). *Using explicit ontologies in KBS development.* International Journal of Human-Computer Studies. 45: 183-292.