Analysing Natural Language Queries at INEX 2004

Xavier Tannier, Jean-Jacques Girardot and Mihaela Mathieu

École Nationale Supérieure des Mines de Saint-Etienne
158 Cours Fauriel
F-42023 Saint-Etienne, France
tannier,girardot,mathieu@emse.fr

Abstract. This article presents the contribution of the "École Nationale Supérieure des Mines de Saint-Etienne (France)" to the new Natural Language Processing special Track of the third Initiative for Evaluation of XML Retrieval (INEX 2004). It discusses the place of NLP in XML retrieval and presents a method to analyse natural language queries.

1 Introduction

If XML (eXtended Markup Language) becomes – as expected – a universally accepted standard for exchange and storage of information, it will soon turn into a necessity to query these structured documents in natural language rather than in a structured query language.

The aim of the new INEX NLPX Track (Natural Language Processing for XML Information Retrieval) is to promote "interaction among researchers in the field of Natural Language Processing (NLP) and XML Information Retrieval". Our participation to this track lies within this scope: the purpose is to add our contribution to the general reflection about the applications of NLP methods to XML retrieval.

Our objective at INEX 2004 is clearly not (yet) the demonstration of retrieval effectiveness of a system, but the implementation of a technique for analysing a natural language query.

This article considers the benefits that can be gained from using some natural language processing methods on one hand, and the specificities of structured documents on the other hand, in order to retrieve information from an XML corpus. It also presents and discusses our method that relies on structure of documents to "understand" the semantics of the request.

2 How can NLP help?

Applications of Natural Language Processing for Information Retrieval have been extensively studied in the case of textual (flat) collections (for overviews on this subject, see [1, 2, 3, 4, 5]). Linguistic analyses of the corpus and/or the query should carry out some decisive improvements in retrieval process. Nevertheless, only a few linguistic methods, as phrasal term extraction or some kinds of query expansion, are now commonly used in information retrieval systems. At present, actual results are not yet up to what we could expect [6, 5].

However we think that the spread of structured corpora can bring new hopes to NLP supporters, at least for the two following reasons:

- Benefits that can be gained from allowing requests in natural language are probably much higher in XML retrieval than in traditional IR. In the last case, a query is generally a keyword list which is quite easy to write. In XML retrieval, such a list is not enough to make queries on both content and structure; for this reason, advanced structured query languages have been devised.
  But one wants XML to be really widely used, and that implies that novice and casual users should be able to make requests on any XML corpus. In this perspective, two major difficulties arise, because we cannot expect such users to:
  - learn a complex structured and formal query language[1];
  - have a full knowledge of the DTD and its semantics.

  Note that these issues already exist in the domain of databases with the Structured Query Language (SQL); but unlike databases, XML format looks set to become used by the general public, notably through the Internet. Although unambiguously machine-readable, structured and formal query languages are necessary (in order to actually extract the answers), the need for simpler interfaces will become more and more important in the future.
- In order to perform a really effective natural language-based retrieval in a flat document, a system should "understand" the semantics of the text, and this is not feasible yet. In the case of structured documents, a well-thought and semantically strong structure formally marks up the meaning of the text; this can make easier query "understanding", at least when this query refers (partly) to the structure (VCAS task in INEX).
  However, this requires a certain amount of knowledge about the corpus, that has to be integrated into a system besides documents themselves, in order to perform a good retrieval process. We discuss this subject in Sect.6.3.

3 Description of our Approach

Our aim is to translate the `<description>` part of INEX topics, written in natural English, into a query in a formal structured language. Topics are divided into two categories:

- *Content-and-Structure* queries, which contains structural constraints.
  e.g.: *Find* paragraphs *or* figure-captions *containing the definition of Godel, Lukasiewicz or other fuzzy-logic implications.* (Query 127)

[1] In this paper we call "*formal query language*" a language with formalized semantics and grammar, as opposed to natural language.

– *Content-only* queries that ignore the document structure.
e.g.: *Any type of coding algorithm for text and index compression.* (Query 162)

To achieve the analysis of such requests, the steps that we perform are:

– a part-of-speech tagging of the query (3.1);
– a syntactic/semantic analysis of the query (3.2);
– with the help of specific rules (3.3):
  • a recognition of some typical constructions of a query (*e.g.: Retrieve + object*) or of the corpus (*e.g.: "an article written by [...]"* refers to the tag *au – author*);
  • and a distinction between semantic elements mapping on the structure and, respectively, mapping on the content;
– a treatment of relations existing between different elements (3.4);
– the construction of a formal language query (3.5).

3.1 Part-of-Speech Tagging

A part-of-speech (POS), or word class, is the role played by a word in the sentence (*e.g.:* noun, verb, adjective...). POS tagging is the process of marking up words in a text with their corresponding roles. To carry out this task we chose the tool TreeTagger [7]. Example 1 represents a query and its POS tagging[2] (*Find* is an imperative verb, *of* and *with* are prepositions and *that* is a relative pronoun).

(1)	*Find*	*the*	*title*	*of*	*articles*	*that*	*deal*	*with*	*semantics.*
	V(I)	DET	NOUN	PREP	NOUN(P)	REL_PRO	V	PREP	NOUN

3.2 Syntactic/Semantic Analysis

This analysis is performed with a set of context-free rules describing the most current grammatical constructions in queries and questions. As an example, we listed in Fig.1 the rules that are triggered when parsing our sample query 1.

NP → DET? NOUN	*REL_PROP → REL_PRO VP*
NP → NP PREP NP	*VP → V PREP? NP*
NP → NP REL_PROP	*S → V(IMP) NP*

Fig. 1. Examples of rules, with S = Sentence, NP = Noun Phrase, REL_PROP = relative proposition, VP = Verbal Phrase. Question mark "?" means that the element is optional in the sequence.

A recursive application of these CFG rules results in the *syntactic tree* represented in Fig.2[3].

[2] For a clearer comprehension by a non-expert reader, we changed the names of the tags into less complete but more explicit abbreviations.

[3] Note that with this set of rules two different parsings are possible: the relative proposition can be attached to the noun *"article"* (as shown in the figure) or to the noun *"title"*. In practice both trees are explored.

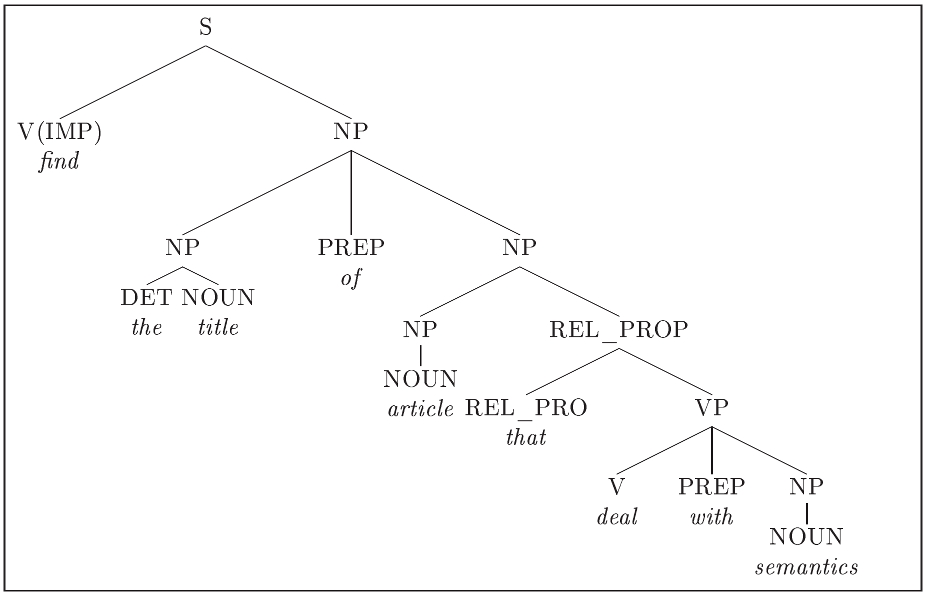


Fig. 2. Syntactic tree of sentence 1, obtained with rules of Fig.1

This operation gives us a syntactic structure, but we need some semantics to have an idea about the relations existing between words. In that aim, we use a very simple implementation of Discourse Representation Theory (DRT) [8] semantic representation of a discourse (or a part of discourse) is described with a two-level "box" called "Discourse Representation Structure" (DRS). The upper level gives the discourse referents, which are the elements introduced by the discourse; the lower level represents the conditions concerning the referents.

A typical example of DRS is given in Fig.3. In this example, the terms *"Napoléon"*, *"Austerlitz"*, *"battle"* and the verb *"to win"* (event e) are discourse referents. Referents are represented by letters in the upper level and described by logical predicates in the lower level. The other conditions are about the agent and the object of the event (respectively *"Napoléon"* and *"battle"* for the event *"to win"*) and the location of the battle.

To compute a DRS representing a whole sentence, we attribute a basic DRS to each word, depending on its class (POS). Syntactic rules are then enriched with semantic actions. Figure 4 shows how the following rule:

$$VP \rightarrow VERB \;\; PREP? \;\; NP,$$

applies to basic DRSs. Moreover, two semantic actions are associated with this rule: $e_1 = e_2$ and $x = y$.

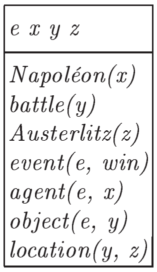


Fig. 3. DRT representation of sentence: *"Napoleon wins a battle in Austerlitz"*

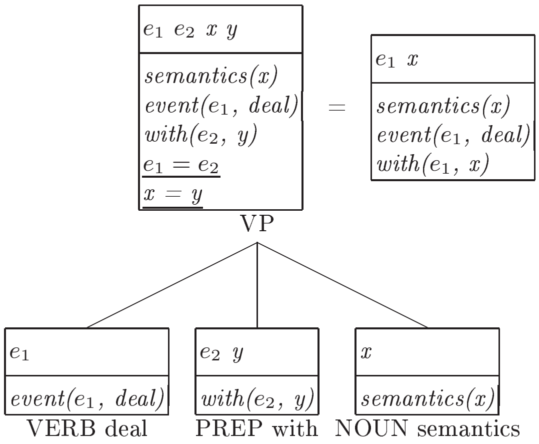


Fig. 4. Example of DRS for the verbal phrase "*deal with semantics*"

Due to a lack of space we cannot show the full semantic tree obtained for the example. Figure 5 gives the final DRS.

N.B.: The set of syntactic/semantic rules that we use is made up of about 50 rules. It is obviously not intended to describe the whole language. The stress has been put on noun phrases, which are often much more meaningful than verbal phrases (at least in terms of Information Retrieval). Relative propositions, prepositional phrases are also very important because they mark a query structure that we do not want to miss. For complex demands, an entire parsing is often impossible. In that case only NPs are analyzed and verbs are left out.

The DRS that we obtain at this stage cannot be used to build a formal query yet. Some IR-specific rules have to be set up.

$x\ y\ z\ s\ e_1\ e_2$
event(e_1, *find*) *event*(e_2, *deal*) *title*(x) *article*(y) *semantics*(z) *object*(e_1, x) *of*(x, y) *agent*(e_2, y) *with*(e_2, z)

Fig. 5. DRS for sentence 1

3.3 Specific Rules

The DRS can be reduced by considering some special cases, among which:

1. **"Query verbs"** like *"to want"*, *"to find"*... With the help of a dictionary describing semantic relations between those verbs and queries, we set a particular flag on the concerned element. This flag means that the element should be selected as a good answer to the query, as shown in example 2.
   (2) I **want** an article. *(see Fig.6.a)*
   Here we know that the verb *"to want"* means that its object (*"article"*) has to be selected. This new information is represented by a framed referent. These verbs and their agents (here the *speaker*, or *"I"*) are then left out.
2. **Description verbs** like *"to deal with"*, *"to concern"*... An other dictionary helps to add a new relation called *about*:
   (3) an article that **deals with** semantics. *(see Fig.6.b)*
   The verb referent is removed as well in this case.
3. **Verbs of topological relation** like *"to contain"*, *"to include"*... If such a verb has an agent and an object, then an appropriate relation is set up between those two elements. The verb is deleted:
   (4) a section that **contains** a figure... *(see Fig.6.c)*
4. Some corpus-specific **semantic rules** have to be added in order to recognize some precise linguistic constructions.
   (5) a document written by Jiawei Han.
   Here an *ad-hoc* rule imposes the transformation given in Fig.6.d[4].
5. **Words or phrases in quotation marks** are considered as non-separable expressions and are grouped together in a single variable.

[4] *N.B.:* This kind of rules no longer represents *linguistic* features, but IR-specific rules. The two *contains* predicates in this example do not have a real linguistic meaning, but express a structural constraint in the XML document.

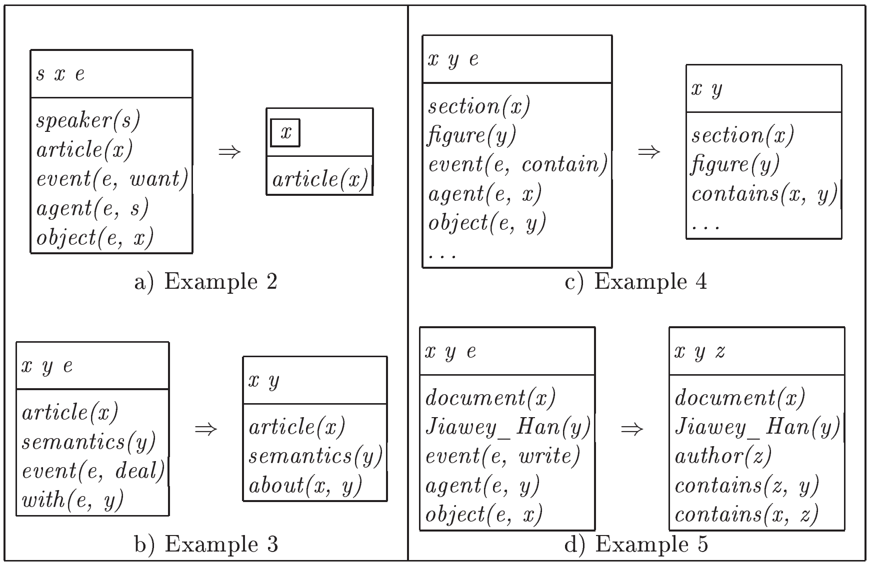
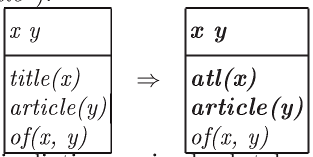


Fig. 6. Semantic representation of examples 2, 3, 4 and 5

(6) We are looking for sections in articles, whose abstracts contain "spatial join," that describe "performance evaluation." *(Topic 156)*

6. And above all, **a term recognized as a DTD-tag (or synonym)** is changed into this tag name and is marked as such (here by a bold predicate in the DRS, with *atl* standing for *"title"*).

   (7) the **title** of an **article**...

   A dictionary of synonyms is used. This dictionary is absolutely not intended to have a general purpose. Indeed DTD tag names are rarely real words, but abbreviations instead (*st* for *section title*, *p* for *paragraph* etc.).

Figure 7 shows the application of some of these specific rules on our sample DRS. Let us remind the initial request:

(1) *Find the title of articles that deal with semantics.*

We suppose that our dictionary tells us that the words *"title"* and *"article"* respectively stand for tag names *atl* and *article*.

In this new DRS, we can clearly distinguish a *tag name*, which is related to the document structure (in bold type), from what we will now call a *term*, as *"semantics"*, which is supposed to be a part of textual contents of the document.

$$\begin{array}{|l|}\hline x\ y\ z\ s\ e_1\ e_2 \\ \hline \mathit{event}(e_1,\ \mathit{find}) \\ \mathit{event}(e_2,\ \mathit{deal}) \\ \mathit{title}(x) \\ \mathit{article}(y) \\ \mathit{semantics}(z) \\ \mathit{object}(e_1,\ x) \\ \mathit{of}(x,\ y) \\ \mathit{agent}(e_2,\ y) \\ \mathit{with}(e_2,\ z) \\ \hline \end{array} \Rightarrow \begin{array}{|l|}\hline \boxed{\boldsymbol{x}}\ y\ z \\ \hline \boldsymbol{atl(x)} \\ \boldsymbol{article(y)} \\ \mathit{semantics}(z) \\ \mathit{of}(x,\ y) \\ \mathit{about}(y,\ z) \\ \hline \end{array}$$

Fig. 7. DRS for sentence 1 before and after application of specific rules. Referent x is selected (rule 1), Article y is *about "semantics"* (rule 2) and terms *"article"* and *"title"* are recognized as tag identifiers *atl* and *article* (rule 6).

3.4 Structure Analysis

At this stage, some binary relations between referents have not been treated by any specific rule (in Fig.7 the relation *of(x, y)*). These relations have all a particular meaning, and a system cannot have the knowledge of each of these meanings. We only implemented a semantic representation of some important relations (and particularly "temporal" relations – *after*, *included*, etc., that should be understood here as order constraints in XML file).

Let $R(x, y)$ be a DRS condition. To handle "known" relations as well as "unknown" ones, we apply a heuristic according to the following cases:

1. x and y refer to two tag names (representing structural elements):
   (a) if the relation R is known, no action is needed.
   (8) A **paragraph** <u>after</u> a **figure** (*see Fig.8.a*).
   (b) if the relation R is unknown, the fact that a relation exists is in itself an information: the structure given by the DTD allows to guess which relation(s) it can be. In our example, the DTD will tell us that an element *atl* (title x) is *contained by* an element *article* (y).
   (9) A **title** <u>of</u> **article** (*see Fig.8.b*).
2. The relation links a tag (let us say x) and a term (y):
   (a) if R is known, we add a tag that can match with any name (called '*'). This tag is *about* y, and the relation is transfered to this new tag:
   (10) A **paragraph** <u>before</u> the *conclusion*[5] (*see Fig.8.c*).
   The paragraph should be written in the XML file before a tag that contains the word *"conclusion"*.
   (b) if R is unknown, we keep it without change (see comments for rule 3).

[5] We suppose in this example that the word *"conclusion"* cannot be assimilated to a tag name, as it is the case in IEEE collection.

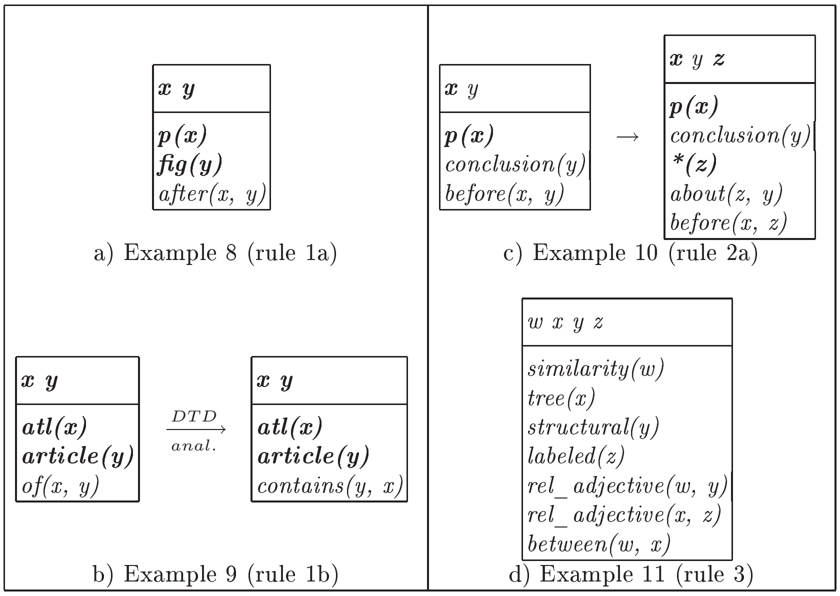


Fig. 8. Application of different structural rules.

3. R holds between two terms (*term* is here used as opposed to *DTD tag name*): in that case the relation does not apply to the structure, because x and y refer to content elements. We do not have any particular treatment to do, but the relation can be a useful linguistic information. We keep it for the retrieval process (which can use it or not).

   (11) The structural similarity between labeled trees (*see Fig.8.d*).

   If the search engine is able to handle such relations (*adjectives* and *between*), it is useful to know that, for example, the whole phrase *"similarity between trees"* is preferable to the separate words *"similarity"* and *"trees"*.

Among these rules, only rule 1b applies in our running example, and the final DRS is shown in Fig.9.

3.5 Formal Language Query

At the end of the linguistic phase, our aim is to obtain a formal language query that could easily be translated into an existing structured language. From those languages (and initially from SQL) we take the idea of clause pattern (SELECT-FROM-WHERE in SQL) for restructuring the request. We chose the following four-clause pattern (expressed in an XML syntax):

– `from` clause: tag names and indications on paths (XPath [9] expressions);

$\boxed{x}$ y z
atl(x)
article(y)
semantics(z)
contains(x, y)
about(y, z)

Fig. 9. final DRS for sentence 1.

- `select` clause: elements to be returned to the user; those elements must be refered in the `from` clause[6];
- `where` clause: relations between tags or variables (before(x,y), about(a,b)...);
- `variables`: identifiers replacing terms in the other clauses.

Transformation process from DRS to formal language is straightforward: tag names are already flagged (in bold type in our representation), as well as selected elements (framed referents in the DRS). Variables are the unary predicates that are not tag names, and `where` clause corresponds to the other conditions.

In this manner, our system automatically generates a query in XML. Not surprisingly, this form is quite verbose and hardly readable. We rather chose to show a SQL-like representation in Fig.10 (from DRS of Fig.9 – we can obtain it with a simple XSL Transformation [11]).

```
FROM y = /article,
     x = y//atl,
WHERE about(y, z)
VARIABLES z = "semantics"
SELECT x
```

Fig. 10. Example of formal query obtained from DRS 9 (query 1). `y` is an *article* tag, `x` is an *atl* tag (title) contained in `y`, and `z` represents the term *"semantics"*.

4 Examples

We give here three significant examples of topic analyses. We already explained that several syntactic parsings could be possible for the same sentence. In prac-

[6] `select` clause only contains element names, we do not provide (as in XQuery-like [10] query languages) any possibility of formatting the output. Besides this is neither the purpose of INEX nor of IR in general.

tice a "score" is attributed to each rule release, depending on several parameters (among which distance between words that are linked, length of phrases, type of relations... Unfortunately we lack space to explain more precisely this process). In our sample topics only the best scored result is given.

4.1 Topic 127

(127) Find paragraphs or figure captions containing the definition of Godel, Lukasiewicz or other fuzzy-logic implications.

Figure 11 shows the first DRS, obtained before the application of IR rules. The final query is given in Fig.12. If we reassemble terms with their relations, an approximate (manual) translation into NEXI could be:

```
//*//(p| fgc)[about( .//*, definition of godel definition of lukasiewicz)
             OR about(.//*, other fuzzy-logic implication)]
```

But this translation is less expressive and we loose some information.

c1 c2 c3 c4 c5 c6 c7 c8 c9 c10 c11 c12 c13
event(c1, sym:find, object:c12) *event(c2, sym:contain, object:c11, agent:c12)* *of(c3,* $c9 \wedge c10$*)* *definition(c3)* $c9 = godel$, $c10 = lukasiewicz$ *'fuzzy-logic'(c4)* *implication(c5)* *rel_ nmodifier(c5, c4)* *rel_ adjective(c5, other)* *paragraph(c6)* *figure(c7)* *caption(c8)* *id_ nmodifier(c8, c7)* $c12 = c6 \vee c8$, $c11 = c3 \vee c5$

Fig. 11. DRS for topic 127. The predicate "`rel_adjective(x, `*`text`*`)`" means that the word *`text`* qualifies the variable `x`. The relation "`rel_nmodifier(x, y)`" stands for nominal compounds (here, *"fuzzy-logic implication"* and *"figure caption"*).

4.2 Topic 141

(141) Retrieve sections about threads from articles about java

This is a textbook case, a short and syntacticaly unambiguous query, that works perfectly. See Fig.13. The translation into NEXI is straightforward:

```
//article[about(., java)]//sec[about(., thread)]
```

```
from   v2 = *, v3 = *, c12 = *, v4 = *, v5 = *
       c6 = v4//p
       c8 = v5//fgc
where  c12 = v4 or v5, c12 = v2 or v3
       ling_adj(c5, v1)
       ling_np_relation(c3, c10, of)
       ling_np_relation(c3, c9, of)
       ling_np_relation(c5, c4, nmodifier)
       about(v3, c5), about(v2, c3)
select c12
var    c5 = implication, c3 = definition, v1 = other
       c10 = lukasiewicz, c9 = godel, c4 = fuzzy-logic
```

Fig. 12. Final structured query for topic 127. Relations beginning by "`ling_`" are linguistic relations that it would be interesting (but not essential) to treat (in order to take into account possible variations in multi-word terms [12, 13]).

```
from   c3 = /article
       c1 = c3//sec
where  about(c3, c4), about(c1, c2)
select c1
var    c4 = java, c2 = thread
```

Fig. 13. Final structured query for topic 141

4.3 Topic 164

(164) I am surveying the area of knowledge management for a report I am writing, and am looking for knowledge management frameworks and technologies for organizational memories.

This topic raises some problems described in Sect.6.2. Most of the proposed relations (see Fig.14) are interesting, but some terms (and an erroneous analysis of "I am writing...") generates noise (especially *"write" (c4)* and *"report" (c3)*). A CO-NEXI reconstitution would be:

```
knowledge management for report - write for knowledge management framework
for organizational memory - write for technology for organizational memory
```

5 Retrieval Process

At this stage we have a formal query that can be translated into an existing and implemented language. But while doing that we should keep in mind that Information Retrieval is yet to be done:

```
from   v1 = *, v2 = *
where  ling_np_relation(c1, c3, for)
       ling_np_relation(c1, c2, nmodifier)
       ling_np_relation(c7, c6, nmodifier)
       ling_np_relation(c6, c5, nmodifier)
       ling_np_relation(c4, c8, for)
       ling_np_relation(c4, c7, for)
       ling_np_relation(c8, c9, for)
       ling_np_relation(c7, c9, for)
       ling_adj(c9, v6)
       about(v2, c4), about(v1, c1)
select v1
var    c4 = verb_write, c8 = technology, c7 = framework,
       c9 = memory, c1 = management, v6 = organizational,
       c3 = report, c2 = knowledge, c6 = management, c5 = knowledge
```

Fig. 14. Final structured query for topic 164

- In our example the *about* relation is to be implemented in one way or another.
- It would be great if the search engine could "understand" linguistic constraints that we generated (see Sect. 3.4 and 4).
- As the request is in natural language, the `from` clause should not necessarily be considered as strict paths. If a user's request is *"give me a paragraph about semantics"*, a section or a figure could also be relevant (maybe not even *less* relevant). This remark corresponds to the INEX choice of assessing CAS topics as *"Vague CAS"* rather than *"Strict CAS"*.

6 Comments

6.1 Structural Density

A comparison between our work on IEEE collection for INEX and our other studies can give us some indications about the relation between density of mark-up in the corpus and ability to analyse properly the requests on this corpus. While we are using the structure to perform the analysis, it makes sense to consider that the higher this density is, the easier our work should be. And this is right to a certain extent. But if a corpus is more *data-centric*, closer to a database format (as address books or flight schedules), then the users' requests on this corpus are very strict and precise (much more than INEX topics). Then the limits of NLP (so far) are reached, and any imperfection in the system, any ambiguity could lead to an erroneous interpretation. We need in such cases to consider a "restricted" natural language, limited to unambiguous structures and pre-defined terms. As a matter of fact, the INEX collection seems to be an "ideal" in-between format to apply our method.

6.2 Topic Complexity

Topics proposed by participants at INEX 2004 are very diverse.

(185) Find articles about gesture recognitions.
(201) ranked retrieval in the WWW.
(187) We are looking for articles containing description of dimension reduction methods. These methods allow us to lessen effects of the "curse of dimensionality" and make retrieval of documents faster. Examples of this methods are well-known latent semantic indexing (LSI) which improves recall of the retrieval system and random projection which does not modify distances too much. We are not interested in stoplist filtration which does reduce the dimension a bit as a byproduct of term removal.

The main problem is not so much the length of the request (from 5 to 76 words) as the fact that language changes when the request is long. Indeed topics 185 and 201 are typical IR requests: a *"Find + Noun Phrase"* construction and a single noun phrase. On the other hand, the last example consists of common language sentences, with many anaphoras and pragmatic insinuations; a proper analysis of all the niceties of language is not conceivable. This leads to a lot of noise in results; for instance words *"effects"* or *"projection"* will be considered by our system as terms to find in the documents as well as the others. If these terms are related to the topic, their importance is much lower. Finally the system does not handle negations.

6.3 Corpus Knowledge

During the stages that we detailed in the previous sections, we used several kinds of information about the explored corpus. This knowledge has to be modeled for each new set of documents[7] to study. Even if our track is not concerned by heterogeneous documents, we tried to reduce this necessity to the minimum. But despite this, we think that the following points are the minimum knowledge to handle in order to perform an appropriate query analysis:

- The DTD
- Because DTD tag names are not "real" words and also because several words can be used for a single concept, we need a dictionary of acceptable synonyms for tag names (*e.g.:* paper = article = document, title = atl, etc.);
- Semantic locutions (*e.g.:* "a list of keywords" = "keywords"), in order to avoid noise generated by an erroneous detection of terms (here, *list* is neither a tag name nor a term to look for in the text);
- Some very simple ontologic structures (*e.g.:* "a novel written by Marcel Proust" = "a novel of which the author is Marcel Proust" – if we suppose that *novel* and *author* represent tag names);

[7] We call a *new* set of documents a corpus with a different DTD and different subject.

7 Conclusion

The Graal of Information Retrieval, which is to *"build software that will analyse, understand, and generate results in response to queries that humans express naturally"* (INEX NLPX track presentation), is far from being reached. It requests a detailed syntactic, semantic and pragmatic understanding of both queries and documents; this is impossible in the present state of our knowledge and resources. In this paper we focused on a first step, a technique to translate natural language queries into a structured formal query language. On top of the ameliorations of this system, we now intend to work on the use of the linguistic information provided by our method, in order to improve an XML retrieval system.

References

[1] Smeaton, A.F.: Information Retrieval: Still Butting Heads with Natural Language Processing? In Pazienza, M., ed.: Information Extraction – A Multidisciplinary Approach to an Emerging Information Technology. Volume 1299 of Lecture Notes in Computer Science. Springer-Verlag (1997) 115–138

[2] Feldman, S.: NLP Meets the Jabberwocky: Natural Language Processing in Information Retrieval. Online (1999) http://www.onlinemag.net/OL1999/feldman5.html.

[3] Smeaton, A.F.: Using NLP or NLP Resources for Information Retrieval Tasks. [14] 99–111

[4] Arampatzis, A., van der Weide, T., Koster, C., van Bommel, P.: Linguistically-motivated Information Retrieval. In Kent, A., ed.: Encyclopedia of Library and Information Science. Volume 69. Marcel Dekker, Inc., New York, Basel (2000) 201–222

[5] Sparck Jones, K.: What is the role of NLP in text retrieval? [14] 1–24

[6] Strzalkowski, T., Lin, F., Wang, J., Perz-Carballo, J.: Evaluating Natural Language Processing Techniques in Information Retrieval. [14] 113–145

[7] Schmid, H.: Probabilistic Part-of-Speech Tagging Using Decision Trees. In: International Conference on New Methods in Language Processing. (1994)

[8] Kamp, H., Reyle, U.: From discourse to logic. Kluwer Academic Publisher (1993)

[9] : (XML Path Language (XPath). World Wide Web Consortium (W3C) Recommandation) http://www.w3.org/TR/xpath.

[10] : (XQuery 1.0: An XML Query Language. World Wide Web Consortium (W3C) Working Draft) http://www.w3.org/TR/xquery.

[11] : (XSL Transformation (XSL). World Wide Web Consortium (W3C) Recommandation) http://www.w3.org/TR/xslt/.

[12] Fabre, C., Jacquemin, C.: Boosting Variant Recognition with Light Semantics. In: Proceedings of the 18th International Conference on Computational Linguistics, COLING 2000, Saarbrücken (2000) 264–270

[13] Sparck Jones, K., Tait, J.I.: Automatic Search Term Variant Generation. Journal of Documentation **40** (1984) 50–66

[14] Strzalkowski, T., ed.: Natural Language Information Retrieval. Kluwer Academic Publisher, Dordrecht, NL (1999)