Towards Automation

Andrei Mikheev and Costanza Navarretta

August 1994

ET-12 Project
Methodologies for Constructing Knowledge Bases
for Natural Language Processing Systems
Report 7

Contents

Preface

This report (Report 7) is the deliverable for Work Package 5 of the project *Methodologies for Constructing Knowledge Bases for Natural Language Processing Systems*. In the first two phases of the project we made the state-of-the-art of knowledge engineering for natural language processing systems (Report 1), we determined which *kinds* of knowledge should be encoded in natural language understanding systems (Report 2) and which characteristics support material should have (Report 3).

On the basis of the theoretical framework for conceptual analysis and knowledge elicitation outlined under Work Package 2 we proposed a revised methodology for knowledge acquisition and elicitation (Report 4, interim and final version) and we evaluated the (re)-usability potential of existing linguistic resources (Report 5).

In Report 6 we described applications of the methodology to a large text corpus.

The objective of the present Work Package are to investigate to which extent the methodology for knowledge acquisition can be automatized.

1 Introduction

In Report 4 (Mikheev and Navarretta 1994) we presented a method for knowledge elicitation and acquisition for constructing knowledge bases for natural language understanding systems. We supplied the method with strategies and/or techniques for shoring up the knowledge elicitation and acquisition process.

In Report 5 we evaluated the reusability potential of existing resources. We considered both linguistic resources which are available on computer (i.a. machine-readable dictionaries, term banks, thesauri, lexical knowledge bases) and tools and techniques for analysing and processing large text corpora. We also looked at some tools used in the framework of other expert systems.

The objectives of the actual phase are to more systematically investigate the automation of the methodology for knowledge acquisition. To this purpose we will both consider existing tools and define tools specialized to our working framework.

The acquisition of semantic features of a domain is one of the well-known bottlenecks in the porting of KB-based systems to new domains. It is well recognized that since much of expert knowledge about almost any topic is encoded in texts, automatically channeling some of that knowledge out of the text and into the knowledge base would greatly aid knowledge acquisition. Apart from regular texts in knowledge engineering for expert systems, interviews, observations and other standard knowledge acquisition techniques tend to result in large quantities of textual data that need to be analyzed and from which knowledge still needs to be extracted. Therefore application independent **acquisition of domain knowledge from textual resources** becomes an important issue.

In recent years, this issue has gained a lot of attention and some progress has been made. However, techniques developed in the expert system field are suffering from a lack of linguistic sophistication and usually are not extended beyond interactive tools which in the best case are sensitive to syntactic categories of words.

For example, knowledge elicitation tools like KRITON (Diederich and Linster 1989) incorporate a text analysis feature: texts are read from a file, nouns are made mouse-sensitive, and the knowledge engineer can include these nouns into a structured representation representing the way the text presents the organization of domain concepts or the way the knowledge engineer perceives them.

It is quite obvious that although this and similar methods have proved to be useful for working with small texts they are not suitable for dealing with large textual data. Therefore a new approach needs to be taken. This approach must uncover linguistically represented ontological knowledge in a highly automated maneer. Research on the processing of large text corpora which has been carried out in the fields of computational linguistics and information retrieval is a very good candidate for the purpose in question. This research explores linguo-statistical methods for uncovering linguistically anchored structural similarities in the text which tend to represent important semantical features of the domain.

The technology used for knowledge base development and the development of information retrieval (IR) systems looks very similar; both require the ability to find lists of domain dependent words (in IR terms, features), and NLP techniques can be used to good effect both to preprocess a corpus to find features (e.g., stemmed words, sense disambiguated words, relevant passages), and to provide the basic units for information retrieval (e.g. noun phrases and groups (e.g. Finch 1994)). Moreover, IR methods themselves can help the knowledge engineer to find documents and passages relevant to the part of the domain he or she is currently trying to engineer.

For both these reasons, it is valuable to provide for IR modules in the knowledge acquisition architecture. These modules should include a passage identification module and a document retrieval module.

One of the most popular methods for extraction of domain semantic features (so called domain schema) from texts is known as Distributional Analysis (Hirshman 1986). It is based on the identification of the sublanguage specific co-occurrence properties of the words in the syntactic relations in which they occur in the texts. These co-occurrence properties indicate important semantic characteristics of the domain: classes of objects and their hierarchical inclusion, properties of these classes, relations among them, lexico-semantic patterns for referring to certain conceptual propositions, etc. These semantic features in the form they are extracted are not quite suitable to

be included into the knowledge base and require a post-processing of the knowledge engineer which is known as conceptual analysis.

The domain schema is usually generated manually after a careful evaluation of the domain which is a time consuming process and often requires the help of a domain expert. However, it seems to be possible to automate this process and facilitate human intervention in many parts using a combination of NLP and statistical techniques for data extraction, type oriented patterns for conceptual characterization of this data and an intuitive user interface.

All these resources can be put together into a Knowledge Acquisition Workbench (KAWB). The workbench supports a spiral process of corpus analysis starting from a rough automatic extraction and organization of lexico-semantic regularities and ending with a computer supported analysis of extracted data and a semi-automatic refinement of obtained hypotheses.

This paper describes a toolkit for a semi-automatic construction of application-independent ontologies using **corpus processing methods**.

2 Towards a Workbench

A workbench for knowledge engineering should integrate computational tools and the user interface to support a spiral process of data extraction, data analysis and hypotheses refinement.

A **data extraction module** provides the knowledge engineer with manageable units of lexical data (words, phrases etc.) grouped together according to certain semantically important properties. The data extraction phase can be subdivided into a stage of semantic category identification and a stage of lexico-semantic pattern extraction. Both of these stages complement each other: a discovery of semantic categories allows the system to look for patterns and found patterns serve as diagnostic units for further extraction of these categories. Thus both these activities can be applied one after another until a certain level of precision and coverage is achieved.

The **word class identification component** encompasses tools for the linguistic annotation of texts, word clustering tools and tools for access to external linguistic and semantic sources like thesauri, machine-readable dictionaries and lexical data bases. First, the corpus can be tagged and each word is assigned with its part of speech information by a special NLP tool known as a tagger. Then a statistical clustering can be applied separately to nouns, adjectives, verbs etc. Found clusters of words can be automatically checked and subcategorized with a help of external linguistic and semantic sources.

The **pattern finder component** makes use of phrasal annotations of texts produced by a general robust partial parser. First, the corpus is checked for stable phrasal collocations for single words and entire semantic clusters by a special tool - collocator. After collocations are collected another tool - generalizer tries automatically deduce regularities and contract multiple patterns into their general representations. Such patterns are then presented for a conceptual characterization to the knowledge engineer. In order to facilitate the analysis some predefined generic conceptual structures are suggested for specialization by a special component.

The main aim of the **hypotheses refinement module** is to uncover and refine structural generalities found in the previous phases. It matches in the text special patterns which represent hypotheses of the knowledge engineer, groups together and generalizes the found cases and presents them to the knowledge engineer for a final decision.

Patterns themselves can be quite complex constructions which can include strings, words, types, precedence relations and distance specifiers. The matcher evaluates how good a given piece of text matches the pattern and returns matches at various levels of exactness. We call this tool a fuzzy matcher after Sinclair (1994).

If modules are to communicate flexibly then an inter-module information representation format needs to be specified. If, for example, a tool which produces a hierarchical representation of concepts in a domain is developed, then tools which use this facility (e.g., a semantic parser) must be able to interpret this structure and having interpreted this structure to mark up text with semantic classes. Thus the representation format must allow the representation of many sorts of information (text, domain ontologies, syntactic structure). On the other hand, the representation format should be simple enough that modules which don't require a highly complex representation can ignore that markup easily so that they can be quickly and cheaply written, and rapidly incorporated into the architecture. Moreover, one of the design goals of the architecture is to allow new tools to be written by many parties and incorporated easily. In this case, it is not possible to define a representation format beforehand for the simple reason that we can't predict what information new tools will need to represent. Consequently it is important that flexible information representation schema are defined in which any information whatsoever can be easily represented.

Standard Generalized Markup Language is an international standard for marking up text. The philosophy behind SGML is that documents should be marked up using an abstract user-definable syntax so that elements of a document receiving the same markup are in some way semantically similar. The philosophy is that elements of a text should be marked up according to what they *contain*, not according to what the user wants them to look like on the printed page. In practice, with new notations being invented continually, and interactions between notations not being fully specified, it is an unfulfillable dream that a single general markup language can be invented for even mathematical formulae.

Our interest in SGML is as a way of exchanging information between modules in a knowledge acquisition system, and of storing that information in persistent store when it has been processed. For this purpose, SGML has several attractive features.

- Many corpora will be marked up using SGML, so little conversion will be necessary to handle new corpora. Moreover, it is relatively straightforward to transform any corpus into SGML.

- SGML is an international standard, so we can expect some "free" tools to become available for manipulating SGML objects (eg. DBMS software, word processors, etc) without having to write software to convert internal objects to a suitable form. We can also expect that the increasing use of SGML for text representation will make our products more attractive to the publishing industry. For example, the NLP tools developed during the LRE project MULTEXT will all be SGML-aware, so these can be relatively straightforwardly plugged into our architecture. IR tools developed during the LRE project SISTA are also SGML-aware. Thus specifying SGML to be the information representation format allows a significant amount of reusability and increases efficiency.

- It is a very flexible and configurable way of transferring information. If it becomes necessary to increase the amount of information in a message, this can be achieved conservatively, so that tools which don't require the additional information will work without modification in the face of the more complex messages. For example, if a tool is built expecting a message to look like "<TAGS> <IN> (some tag string) </IN> </TAGS>", and it is decided to add another field to this (for instance, a document identification field), then since the old type of message is properly contained in the new message (in the sense of tree unification), the tool can process it just as before. This is very useful in prototyping, so that tools can be built for a particular purpose making use only of information necessary for that task, and then extra information can be added transparently to the messages it receives and transmits. That is, a tool built to handle messages using a particular descriptor can also handle messages conformant to a more general one.

  In practice, each module will know some of the semantics of any SGML specified data it processes and ignores what it doesn't know about. For example, a POS tagger might look for all <s> elements and interpret them as sentences.

- In principle, using the features of SGML such as the "Short tag" feature, tag omission and minimization, default attribute values, and so on, it is possible to reduce the length of the message which needs to

be transferred between processes to a minimum. However, if canonical form SGML is used, then this can be parsed very quickly. Consequently, SGML is a very flexible representation language, and adaptable to the changing requirements of the information channel which uses it to represent information.

After this description of the corpus engineering process and main workbench components we will embark on a more detailed characterization of tools themselves. We, however, will not present any technical details and suggestions on an actual implementation because the main idea about the workbench is that it should be able to incorporate different implementations, for example, different taggers or different clustering tools. Some of the tools are already implemented and can be reused in the workbench but others still need implementation or reimplementation in terms of the open architecture of the workbench.

3 NLP Linguistic Annotation Tools

The simplest form of linguistic description of the content of a machine-readable document is in the form of a sequence (or a set) of words. More sophisticated linguistic information comes in several forms, all of which may need to be represented if performance in an automatic acquisition of lexical regularities is to be improved. From a linguist's view of the text, there are several fairly easily identified relationships which might be of considerable use to knowledge engineers:

Part of Speech: (Morpho-Syntactic) Words have different parts of speech according to their linguistic context of occurrence. On its own, syntactic categories are useful in identifying likely content words (e.g. nouns and verbs) which the knowledge engineer might consider importing into the knowledge base. More importantly, much phrasal information is expressible in terms of sequences of word classes much more readily than it is in terms of constraints over sequences of words. The word class is the building block of any parsing strategy, and finding the class of words in a text is the first stage to adding linguistic structure.

Root Form (Morpho-Syntactic) Different words often share the same stem. For example, *tags, tagging, tagged* are all *inflected* forms of the stem "tag". The meaning of words with the same stem are often closely related, the main difference being the syntactic role they play in the sentence in which they are used. The rules for inflection interact with the category of the word; if a word is inflected with '-ed', for example, it must be a verb (or perhaps an adjective); if it is inflected with '-est', it must be a (superlative) adjective. Frequently, especially when the distinct forms of words are uncommon in a corpus, it will be useful to identify the inflected forms of a common stem. This is frequently called *stemming*. Stemming serves to unify related word forms which would otherwise be treated as separate. Although this might seem a reasonable thing to do, engineers should be careful in applying stemming blindly; results in several related areas (e.g. IR) of applying stemming are very mixed, often leading to significant reductions in performance, even when the stemming algorithm is very accurate.

Group and Phrase (Syntactic) Sequences of words are arranged into linguistic groups of (usually) continuous *constituents*. For example, "a

can of worms" has a specifier constituent ("a"), and a noun-group constituent ("can of worms"). This latter phrase consists of two subgroups: "can" and the prepositional phrase "of worms". This latter group itself consists of a preposition ("of") and a noun-phrase ("worms"). This latter noun-phrase consists of the single noun "worms". Various linguistic dependencies can be described in terms of the relationship between groups and phrases. It is intuitively likely, and empirically borne out, that words which are in the same group are more informative both (a) of each other and (b) taken together as a group, of the subject tags of the abstract. Consequently, uncovering grouping information is likely to be a valuable asset in any attempt to find a reliable tagging strategy.

Subject-Predicate (Syntactic - Semantic) From school grammar, each simple sentence has a subject and a predicate. Thus "John loves Mary" can be re-written as "loves-Mary(John)", or "loves(John, Mary)". This sort of information will be very useful to the knowledge engineer seeking to build a comprehensive ontology of semantic relationships within the domain being modelled, and it will also be very useful in finding how concepts in a knowledge base are typically related in real natural language texts (and hence experts' knowledge) about that domain. Subject-predicate relationships are defined from the syntactic form of the sentence.

The NLP module of the KAWB consists of a word tagger, a specialized partial robust parser and a case attachment module. The tagger assigns categorial features to words. This is not a straightforward process due to the general lexical ambiguity of any natural language but state-of-the-art taggers do this quite well using different strategies usually based on an application of Hidden Markov Models (HMMs).

It is well-known that a general text parsing is very fragile and ambiguous by its nature. Syntactic ambiguity can lead to hundreds of parses even for fairly simple sentences. This is clearly inappropriate. However, general and full scale parsing is not required for knowledge acquisition purposes but rather a robust identification of certain text segments is needed. Among these segments are compound noun phrases, verb phrases etc. To increase a precision of knowledge extraction in some cases it is quite important to resolve references of pronominal anaphora.

It is worth mentioning that there is no requirements to obtain a hundred percent precision for the parser since its results will be post-processed by the knowledge engineer. Identified by the parser phrases are allowed to be underspecified: relations inside a phrase may be unresolved, phrases can lack clause boundaries etc. Another source of robustness for a partial parsing are structural regularities of technico-scientific texts. These texts usually have a fairly regular structure and use a high percentage of monosemus words (terms).

The parser supplies information to a *case attachement module*. This module using semantically driven role filler expectations for verb frames provides a more precise attachment of noun phrases to verbs. A semantically-driven case grammar obviously requires more lexical information that can be provided by a tagger. This information can be obtained from a core knowledge base and external lexical data bases which are described elsewhere in the text.

4 The Statistical Module

The statistical module employs different models/methods for extraction and rough structuring (clustering) data from the underlying text corpus. Most of the existing statistics-based methods not only address tagging and parsing of unrestricted texts, but also automatical extraction of lexical semantic information, disambiguation of prepositional phrase attachment and rough clustering of data.

A method for building sublanguage lexicons from text corpora via syntactic and statistic analysis combined with a lexical semantic theory is described in Pustejovsky (1992). In the method different word senses are conflated into a single meta-entry, called lexical conceptual paradigm (LCP), so that the regularities of word behaviour dependent on context can be systematically encoded. The notion of a LCP helps in capturing systematic ambiguities in language such as count/mass alternations, container/containee alternations, figure/ground reversals, product/producer diathesis, plant/food alternations, process/result diathesis and place/people diathesis. The LPCs can be learned from both untagged and tagged corpora.

Hindle and Rooth (1991) propose to use the relative strength of association of the preposition with verb and noun, estimated on the basis of word distribution in a large corpus, for resolving ambiguous prepositional phrase attachment. The used procedure is promising in showing the existence of a relation between nouns and verbs with prepositional phrases[1].

In report 4 (Mikheev and Navarretta 1994a) we reported different statistical techniques for automatically classifying words according to their contexts of use, so called word clustering techniques (pp. 19–24). Many of these strategies/techniques have only been applied to particular cases or have been tested in only one domain. They should be extended to other cases or their generality should be proved.

In particular the first experiments of the distributional clustering technique described by Pereira *et al.* (1993) have addressed the specific problem of classifying nouns, using the relation between a transitive main verb and the head noun of its direct object. This techniques should be applied to other relations (e.g. classifying the transitive main verb using the head noun of its direct object, classifying adjectives using the modified noun (with a dif-

[1] The method cannot be used to determine what sort of relationship is involved.

ferent technique this is at the basis of the technique for clustering adjectives proposed by Hatzivassiloglou and McKeown (1993)). The clustering techniques based on N-gram models (Brown *et al.* 1992, Lafferty and Mercer 1993) should be tested on more domains.

5 MRD Access Tools

Research on machine readable dictionaries has especially addressed the problem of automatically extracting lexical and semantic information from word sense definitions (both genus and differentia). Some methods have been also developed for disambiguating word senses in unrestricted texts using the information contained in machine readable dictionaries and/or in thesauri. Some methods require tools (parsers etc.) which are specially developed to process a particular electronic resources, others do not require any specialized tool.

Many strategies for extracting taxonomies from general language machine-readable dictionaries have been described in Boguraev and Briscoe (1989). Most of these techniques are specialized to a particular resource, *Longman Dictionary of Contemporary English* (LDOCE)[2], but some of the ideas behind them can be adapted to other resources.

5.1 Tools Extracting Semantic Information from MRDs

In Wilks *et al.* (1989) three different approaches for extracting semantic information from LDOCE are described. All three approaches assume that dictionaries do contain sufficient knowledge for at least some NLP applications, and that such knowledge is extricable, but they are based on different assumptions about bootstrapping, i.e. what initial knowledge is necessary. The first approach argues that no prior knowledge is needed at all. The second approach argues that there is a small set of words for which it is impossible to extract the sort of semantic information the authors are looking after, without having prior information available. Both the second and the third approach assume the prior existence of a grammar (one uses case information in the parser, the other extracts case information from human subjects).

[2]This fact is due to the special characteristics of the machine-readable version of LDOCE and in particular to the fact that in LDOCE a restricted vocabulary, called *controlled vocabulary* and containing approx. 2000 words, is used in the meaning definitions.

In the first approach all sentences that contain a word are used as sources of information about the use of that word, rather than just the definition of the word. The basic assumption is that the frequency of co-occurrence of a pair of words provides a reasonable measure of the strength of the semantic relationship between them.

At the beginning information about all the words from the controlled vocabulary is extracted. The interpretation of conditional probability occurrences is taken as a measure of semantic relatedness and measures of semantic relatedness are used as a basis from which to construct useful information about a word and its senses.

Pragmatic knowledge is not considered and the amount of data obtained is reduced investigating the co-occurrences of all the pair of primitives in LDOCE with a program, the PATHFINDER, developed to discover the network structure in psychological data. A program, BROWSE enables to select groups of words using thresholds of various probability functions and write out sub-matrices of the co-occurrence matrix. The PATHFINDER converts the almost completely connected network represented by the sub-matrix into a sparsely connected network. To validate the results obtained they were compared with matrices of judgements of relatedness made by human subjects. It was found that conditional probability of co-occurrence is strongly related to human judgements of semantic relatedness.

In the second approach a set of 1,200 words, called the Key Definig Vocabulary (KDV), is extracted to define the controlled vocabulary of LDOCE, and then all the remaining 27,758 words defined in the vocabulary are processed. The assumption behind this approach is that the words in LDOCE can be defined by the KDV in a series of "four defining cycles" which add progressively more of the LDOCE vocabulary to the KDV.

When a candidate word enters a defining cycle, the stems of the words used in the definitions of the first three senses of the word are examinated. If all the word stems in the definitions occur in the KDV then the candidate word is put in a success file and added to the KDV at the end of the defining cycle. If not the word is put in a "fail" file and its addition to KDV postponed.

The use of a limited number of KDV in the beginning reduces the number of knowledge structures to be manually encoded. The defining cycles help in discovering circular definitions that must be eliminated in a machine tractable dictionary. The initial KDV primitives were empirically found by

intersecting the words of LDOCE controlled vocabulary with the most frequent words and basic words used in The General Basic English Dictionary. A criterion of conceptual simplicity was applied so that only words that express a single main concept were selected. The expansion of the initial set of KDV was made empirically.

The knowledge structure used, ISU (integrated semantic unit), is an enriched representation of word senses. It incorporates linguistic knowledge with general world knowledge in the representation of each word sense. It consists of "Wordsense", a word sense of any entry word in LDOCE, "belong(Superordinate)", which introduces a hierarchical relationship between Wordsense and Superordinate, and "ik" introducing integrated linguistic and word knowledge related to "Wordsense". KDVs arrange themselves in hierarchies that are not tangled (two or more genus words) because each different word sense is considered a different concept (bank1, bank2 etc.).

The third approach consists of a lexicon-producer/consumer system. The lexicon-producer system converts LDOCE entries into lexical semantic structures (frame-based knowledge representation) intended for knowledge based parsing. Each lexical semantic structure (frame) is part of one or more hierarchies. These hierarchies are used i.a. when preference breaking usage happens. In this case it is necessary to relax the grammatical or semantical constraints travelling up a hierarchy of constraints. The frames are in the beginning constructed using the LDOCE's grammatical codes and the general semantic and pragmatic information that is easy to extract from the dictionary. When the lexicon-consumer (a knowledge-based parser operating over non-dictionary texts) needs more information (for resolving lexical-ambiguity or making non-trivial attachment decisions) the lexical semantic phrases are enriched with information extracted from definitions of the words. A parser built especially to analyze the LDOCE's definitions (only content words are analyzed) is used. It produces phrase-structure definition trees which are passed to an interpreter for patter-matching and inferencing. Genus and differentia are extracted and information about them is given to the frame structure (genus-slot is filled, IS-A relations between a word and its definitions are constructed etc.). The frame created for each word-sense from its definition represents the intension of that word-sense. The information so collected is used in the two kinds of computational semantics: Preference Semantics and Collative Semantics.

A) The job of a Preference Semantics Parser is to consider the various competing semantic interpretations for a sentence or constituent and to choose among them by finding the one that is most semantically dense.

The external bootstrapping used to extract the information contained in LDOCE is a grammar for LDOCE sense definitions and a set of meaningful tree patterns.

B) Collative Semantics has four components: "sense frames" and "semantic vectors" (representations), "collation" and "screening" (processes). Sense-frames are the knowledge representation scheme and represent individual word senses. Collation matches the sense-frames of two word senses as a complex system of mappings between their sense-frames. A sense-frame contains genus and differentiae and belongs to a semantic network which is a hierarchy of genus forms. Sense-frames consist of the "node" and the "arcs". The arcs contain a labelled arc to its genus term. The arcs of all the sense-frames comprise a densely structured semantic network of word-senses called the sense-network.

The node part is the differentiae that provides a "definition" of the word sense. Nodes consist of cells which have a syntax modelled on that of English. There are three types of nodes: a) type-0 nodes (nouns) represent their structural and functional properties, b) type-1 (adjectives, adverbs, determines, ordinals and other one-place predicates) contain a preference and an assertion: the former contains semantic information expressing a restriction on the local context, the latter contains semantic information to be imposed onto the local context, c) type-2 (verbs, prepositions, comparatives, conjunctions) are case frames containing case subparts filled by case roles such as agent, object and instrument. Case subparts contain preferences and assertions if a state change is described.

Vossen *et al.* (1989) describe the development of a semantic database (LINKS) in which the meaning descriptions in LDOCE are stored in a systematically related way. The strategy followed is that of stepwise lexical decomposition. The method is the following:

1. A grammatical coding is applied to the words of the restricted vocabulary and their inflected forms.

2. This coding is automatically inserted in all the meaning descriptions. The output is a grammatically-coded corpus of meaning descriptions.

3. A syntactic typology is developed for the structures of the meaning descriptions of each of the major parts of speech (POS), resulting in parser-grammars for each of them. (Applying these grammars to the corpus should lead to syntactically analyzed meaning descriptions).

4. Finally a semantic typology is developed.

The authors want to incorporate the syntactic and the semantic typologies into a relational database system in order to be able to trace the "horizontal" and the "vertical" links between words.

They have created files containing the different codes and types of information contained in the LDOCE tape version:

1. Subject field codes: the domain to which a sense is restricted.

2. Box codes: stylistic, sociolinguistic and semantic information about the sense of the entry.

3. The ortographic form of the entry.

4. The POS code of the entry.

5. The meaning descriptions.

First the words in the controlled vocabulary were manually tagged. In a second step the most important (frequent) of related tokens (multi-words, derivatives etc.) were also encoded. The meaning descriptions (MD) were called nominal meaning descriptions because usually the meaning descriptions of nouns have the structure of noun phrases.

The interpretation of nominal meaning descriptions consisted of four levels: a) word sequence b) POS sequence c) syntactic pattern d) semantic pattern.

The basic structure of a nominal phrase consists of: a) a determiner component (optional), b) a modifier component, pre- and/or post-kernel modifiers (optional), c) a syntactic kernel (obligatory).

There are different types of structures: a) the syntactic kernel is a hyperonym of the entry-word, the pre- and post-modifying elements express restrictions imposed by the meaning of the entry-word on the extension of the hyperonym. In this case the syntactic kernel is called a Link. b) The syntactic kernel is a synonym (no modifying elements). c) The noun of the

modifier (complement, often of-complement) carries more information than the kernel. The relation between the complement and the entry is often expressed by the syntactic kernel that is therefore called Linker (part-of, component-of etc.). d) the syntactic kernel designates a very general class and the post-kernel phrase should be read as the complete filling in of the specific type of the class the entry-word stands for. The complement is not a noun phrase but a verb phrase in the progressive form. The entries having a MD with such a structure are directly associated with a VP that expresses a relation between, or a property of entities. These kernels that shunt information from a nominal form to a non-nominal form are called Shunters.

Problems arise mainly with multiple POS-codes and ambiguous structures of the MDs. The main problems with a semantic analyzer are multiple senses of the words in the MD and circularity of MDs.

Montemagni and Vanderwende (1992) show that a general, broad-coverage parser can be applied to definitions of more dictionaries[3] to recognize structural patterns necessary for extracting semantic information from the definition text (both genus and differentiae).

Research for extracting concept-lattice semantic nets from a general language thesaurus, *Roget's International Thesauri*, 3rd ed., is currently done by more groups in the US (Sedelow and Sedelow 1994, Kent 1994).

By analysing the BSI/ISO 2382 Vocabulary of Information Technology Data Processing (1990) Nkwenti-Azeh (1994) has shown that technical definitions can be used to construct basic knowledge structures using coherent sets of technical definitions. These definitions contain a number of extractable and identifiable relations to other concepts, e.g. *result*, *purpose*, *contiguity*, as well as explicitly named relationships, e.g. *part-of*, *method*, which can be extracted analysing the link elements among the arguments of the defining preposition.

Contrasting the link elements with the facets of the British Standard Institutions ROOT Thesaurus (BSI, 1981) Nkwenti-Azeh shows that a terminological thesaurus containing the key concepts of the domain can be constructed using the terms and the relational phrases extracted from the definitions.

[3] In practice they worked with an English and an Italian parser applied respectively to English and Italian dictionaries.

5.2 Tools for Disambiguating Word Senses

David Yarowsky (1992) describes a program for disambiguating English word senses in unrestricted text using statistical models of the main conceptual categories of the Roget International Thesaurus[4]. The method can also make use of conceptual hierarchies taken from other electronic resources such as *WordNet* and *LDOCE* (subject codes) and consists of the following three steps:

> "For each of the Roget Categories
>
> - Collect contexts which are representative of the Roget category
> - Identify salient words in the collective context and determine weights for each word, and
> - Use the resulting weights to predict the appropriate category for a polysemous word occurring in novel text." (p. 455)

The saliency of words is calculated with a mutual-information-like estimate: $\frac{Pr(w|RCat)}{Pr(w)}$,
i.e. the probability of a word (w) appearing in the context of a Roget category divided by its overall probability in the corpus. The maximum likelihood estimate is not used directly because it can be unreliable, particularly when the word does not appear often in the collective context. Because estimates from the local context are subject to measurement errors and estimates obtained from the global context are subject to being irrelevant, one can reduce both sources of error interpolating between the two. In the present case the local estimates of $Pr(w \mid RCat)$ were smoothed with global estimates of $Pr(w)$.

When a salient word appear in the context of an ambiguous word, there is evidence that the word belongs to the given category. If several of such words appear, the evidence is compounded. In the algorithm Bayses' rule is used to determine the category of words for which the sum of weights is greatest.

[4]The fourth edition of 1977 was used.

The method performs best on words with senses which can be distinguished by their broad context. In the remaining cases the described procedure should be accompanied by other tools.

In Guthrie *et al.* (1991) a method for obtaining subject-dependent associated word sets, using the subject classifications of Longman's Dictionary of Contemporary English (LDOCE)[5] is described. In particular the subset of LDOCE definitions that consists of those sense definitions which share a subject code is considered. These definitions are put into a file and co-occurrence data for their defining vocabulary is created. Many words in LDOCE have no subject code associated to them, this lack is called the "null code" and is considered as a particular subject code. Often the "null code neighborhood" reveals the most generic or common sense of a word.

One of the possible applications of the subject-dependent co-occurrence neighborhoods is the disambiguation of word senses in texts. The procedure consists of following: for each of the subject codes which appear with a word sense to be disambiguated intersect the corresponding subject-dependent co-occurrence neighborhood with the text being considered. The intersection must contain a pre-selected minimum number of words to be considered. If none of the neighborhoods intersect at greater than this decided level the neighborhood N is replaced by the neighborhood $N(1)$, which consists of N together with the first word from each neighborhood of words in N, using the same subject code. If necessary the second and then the following most associated words are added, forming $N(2)$, $N(3)$ etc. until a subject-dependent co-occurrence neighborhood has intersection above the threshold level. Then the appropriate sense or senses is selected. If more than one sense has the selected code, their definitions are used as cores to build distinguishing neighborhoods for them.

[5]In practice they adopt a restructed hierarchy of the LDOCE's subject codes elaborated by Slator and consisting of a top node (all definitions), followed by six fundamental categories, then furthermore subdivided. The maximum depth of the category tree is five levels.

The above described method for lexical disambiguation of texts has been improved (Cowie *et al.* 1992) so that it does not only operate on single words, but on complete sentences, i.e. it attempts to select the optimal combinations of word senses for all the words in the sentence simulataneously in a computational effective way. The lexical disambiguation algorithm has been combined with the technique of simulated annealing[6].

[6]Simulated annealing is a technique for solving large scale problems of combinatorial minimization. The algorithm takes its name from the process by which metals cool and anneal. Slow cooling usually allows metals to reach a uniform composition and a minimum energy state. In simulated annealing a parameter T is decreased slowly enough to allow the system to find its minimum. A function E of configurations of the systems corresponds to the energy that one seeks to minimize. From a starting configuration a new configuration is chosen at random, and a new value of E is computed. If the new value of E is lower than the old one, it is chosen, if it is higher it may be chosen (probabilistically) to avoid settling on a local minimum which is not the actual minimum. The final configuration is an approximation to the best solution.

6 Lexical Database Access Tool

Already existing lexical databases are an important source of information about constituent words of domain texts. One of the most popular publicly available lexical databases is WordNet (Beckwith *et al.* 1990) described in Report 5. It contains information about words, their senses, synonymy, antonymy, hierarchical inclusion etc. A typical query to WordNet looks as follows:

---

wn anterior -antsa - *give information about antonyms for "anterior"*
2 senses of anterior

Sense 1
anterior (vs. posterior)

posterior (vs. anterior)
→back(prenominal), hind(prenominal), hinder(prenominal), hindmost, rear(prenominal)
→retral

Sense 2
anterior, earlier, prior(prenominal) INDIRECT (VIA antecedent) →subsequent, succeeding

wn anterior -synsa - *give information about synonyms for "anterior"*
2 senses of anterior

Sense 1
anterior (vs. posterior)
→ventral
→fore(prenominal), front(prenominal)
→frontal
Also See →front(prenominal)

Sense 2
anterior, earlier, prior(prenominal)
→antecedent (vs. subsequent), preceding

The workbench can be equipped with a specialized parser which parses output of WordNet queries into a structure similar to Feng *et al.* (1994) but with more fields and using SGML descriptors:

word	anterior
POS	adjective
pert.noun	NO
direct antonyms	posterior
sem.related	ventral, fore, front, frontal, prenominal

Such structures can be used for many purposes ranging from word sense disambiguation to a subclustering of word clusters.

7 External Thesaurus Access Tool

For some domains there already exist terminological banks available on-line. These banks vary in their linguistic coverage - some list all possible forms (singular, plural etc.) for terms while others just a canonical one, and in a conceptual coverage - some provide an extensive set of different relations among terms (concepts) others just a subsumption hierarchical inclusion. A module which supports access to an external thesaurus performs:

- thesaurus multidirectional browsing. There can be many attributes for browsing: alphabetical order, a particular relation, a lexicographical resemblance etc. Browsing can be supported by a graphic user interface with a structural term representation.
- word or phrase search in the attached thesaurus: for an input word or phrase the module gives an answer whether it is a term in the thesaurus or not and if it is not - gives a lexicographically resembling set of term which have words in common. For instance, for a phrase *"left anterior discending artery"* the module can give the following set: *"left anterior discending coronary artery"*, *"left coronary artery"...* with a different scores of matching.
- retrieving a main domain category for a term: for any term the module gives its main domain category, for example, for the term *myocardial infarction* the category is *disease*. Note that a domain category is not an immediate supertype but the most general domain type.
- subsumption checking: for a pair of terms the module gives an answer whether one is a subclass of the other.
- term marking facility: a text is checked against the thesaurus and all found in the text terms are marked in it.

Of course, different thesauri have different data formats and different software (if any) for working with them. So the module should be equipped with thesaurus-independent functions and provide generic functions for thesauri access which can be reprogrammed for a particular thesaurus. For example, a graphic interface with a structural term representation can be implemented as data independent and rely on particular functions of the attached thesaurus which give it information to represent.

8 Collocator

This tool finds significant co-occurrence of lexical items (words and phrases) in the corpus and composes so-called lexico-semantic patterns. To do that this module uses a linguistically annotated text and counts frequencies of co-occurrence for a given word, type, phrase etc. with other linguistical items. Here is an example query for the type DISEASE:

Num	Freq	Annotated Phrase
$136	373	myocardial//BODY-PART infarction//DISEASE
$234	475	anterior myocardial//BODY-PART infarction//DISEASE
$467	550	inferior myocardial//BODY-PART infarction//DISEASE
$1109	1	established inferior myocardial//BODY-PART infarction//DISEASE
$1154	48	history//INFORMATION of ischaemic heart//BODY-PART disease//DISEASE
$2574	2	history//INFORMATION of an anterior myocardial//BODY-PART infarction//DISEASE
$2974	2	moderately severe stenosis//DISEASE
$2980	4	aortic valve stenosis//DISEASE
$3004	79	stenosis//DISEASE in the right coronary artery//BODY-PART

As one can see many terms include other terms as their components. This surface lexical structure closely corresponds to semantical relations between concepts represented by these terms. To uncover term inclusion the system scans the term bank and replaces each entry of a term which currently in focus with its number:

Num	Freq	Annotated Phrase with Replaced Terms
$136	373	myocardial//BODY-PART infarction//DISEASE
$234	475	anterior $136
$467	550	inferior $136
$1109	1	established $467
$1154	48	history//INFORMATION of ischaemic heart//BODY-PART disease//DISEASE
$2574	2	history//INFORMATION of an $467

Now the term bank is suitable for further analysis: the system can find out properties for known types and determine patterns of a structural inclusion between them. It is generally recognized that (very roughly) nouns correspond to objects, verbs to eventualities and adjectives to properties. Based on this we can make a further assumption: the head of a noun phrase corresponds to the main type and all other constituents usually represent

- *properties* if they are adjectives not pertaining to nouns, i.e. they are not adjectivized nouns. For example, the adjective “major” stands

for a property while the adjective "myocardial" which pertains to the noun "myocardium" stands for a structurally linked object.

- *components* if they are modifiers which pertain to nouns i.e. it can be direct or prepositional noun or noun phrase modifiers and adjectives which pertain to nouns. For example, "infarction of myocardium", "myocardial infarction", "heart disease".

9 Generalizer

This tool takes the output of the collocator and tries to find out regularities in available collocations. For example, for collocations:

[myocardium//BODY-COMPONENT <> adj]→(mod)→[infarction//DISEASE <> head]

[endocardium//BODY-COMPONENT <> adj]→(mod)→[infarction//DISEASE <> head]

[heart//BODY-COMPONENT <> noun]→(mod)→[disease//DISEASE <> head]

[occlusion//DISEASE <> head]→("of")→[LAD//BODY-COMPONENT <> abbr-noun]

[aorta//BODY-COMPONENT <> adj]→(mod)→[stenosis//DISEASE <> head]

[artery//BODY-COMPONENT <> noun]→(mod)→[stenosis//DISEASE <> head]

---

this module can arrive at lexico-semantic patterns as follows:

[BODY-COMPONENT]→(mod)→[DISEASE <> head]

[DISEASE <> head]→(in)→[BODY-COMPONENT]

[DISEASE <> head]→(of)→[BODY-COMPONENT]

[blood-vessel//BODY-COMPONENT <> noun, adj]→(mod)→[stenosis//DISEASE <> head]

[heart-muscle//BODY-COMPONENT <> adj]→(mod)→[infarction//DISEASE <> head]

There is a general fact that a body component somehow modifies a disease and there are two more specific facts about stenosis and infarction. Now the system can ask the knowledge engineer to characterize the type of the relation and the difference between its different sorts (adjectival, nominal, prepositional "of", "in" etc.).

Another important task of the generalizer is to classify adjectives (which as we already said mainly correspond to properties of types) into semantical clusters. So the system can apply the following strategy:

- start with terms of the minimal available length which have the type in question as a head. For example, for the type INFARCTION the systems sorts terms in the following order:

length 2: myocardial infarction, old infarction, acute infarction, limited infarction...
length 3: acute myocardial infarction, anterior myocardial infarction...
length 4: further antero-septal myocardial infarction...

- collect all adjectival modifiers for the type and separate pure adjectival modifiers from adjectivized nouns:

  infarction : inferior, old, acute, post, further, antero-lateral, lateral, infero-posterior, antero-septal, repeated, significant, large, limited // myocardial, diaphragmatic, subendocardial

  myocardial infarction : anterior, first, extensive, further, minor, small, previous, posterior, suspected.

  diaphragmatic infarction : old, small, acute, impending

  subendocardial infarction : lateral, further, anterior

  anterior myocardial infarction : small, acute, ensuing, large, recent

- cluster pure adjectival modifiers into groups using synonym-antonym information available in WordNet. However, it is not necessarily the case that related adjectives are stated together in one WordNet entry. Sometimes there is an indirect link between adjectives. For example:

---

major : major (vs. minor)
large : (vs. small), INDIRECT (VIA **major**) →**minor**;
extensive : INDIRECT (VIA **large**, big) →**small**;
significant: insignificant (vs. significant) →**small**
limited : **minor**, **small**
old : young (vs. old) →little, **small**

post : INDIRECT (VIA succeeding) →preceding
previous : INDIRECT (VIA preceding) →succeeding
ensuing : INDIRECT (VIA subsequent) →antecedent, preceding

chronic : chronic (vs. acute)

The system can assume that if there is at least one word in common in WordNet entries for two different adjectives they are clustered together.

In our example for the type INFARCTION the following clusters were automatically obtained:

cluster 1: chronic vs. acute;
cluster 2: major, extensive, significant, large, old vs. minor, small, limited;
cluster 3: post vs. previous, ensuing;
cluster 4: anterior vs. posterior;
cluster 5: inferior vs. superior;
unclustered: suspected; lateral; recent; further; repeated;
complex: antero-lateral; antero-septal;

As we see all clusters look fairly plausible except the single adjective "old" which was misclassified; it stands for a temporal property of an infarction rather than its spreading at a myocardium.

This algorithm can be applied to all entries from the term bank and the knowledge engineer is presented with the results for the construction of semantical structures with the help of a type-oriented analysis tool.

10 Analysis Support Tool

Type oriented analysis is facilitated with generic conceptual structures which are different for different conceptual types. For example, a type oriented structure for eventualities includes their thematic roles (agent, theme ...), temporal links and properties while a type-oriented structure for objects includes their components, parts, areas and properties. The system recognizes which structure should be used and presents it to the knowledge engineer with optional explanations or a question guided strategy for filling it up. In our example since the type DISEASE is an eventuality (this, for example, can be obtained from WordNet) the system shows a default thematic case frame and the knowledge engineer decides where to put the type body-part:

[@disease:∀ <> head]→(loc)→[@body-COMPONENT]

This process goes incrementally for other constituent types.

Instead of defining a lexico-semantic pattern as a flat construction, for example, [body-component][disease] it is possible to characterize these patterns multi-dimensionally:

[@disease:∀ <> head]
→(hold)→[@time-inteval]
→(expr)→[@person:y]
→(loc)→[@body-COMPONENT]←(has-comp)←[@person:*y]
→(char)→[@dis-degree:∀]
→(char)→[@dis-ext:∀]
→(char)→[@dis-stability:∀]
→(char)→[@dis-periodicy:∀]

[@dis-degree:∀] = { @low = {"some", "mild"};
@mod = { "moderate", "moderately severe", 60-70% };
@severe = {"quite severe", "severe"};
@high = {"very severe"; "high grade", "critical", "tight", "heavy"} }

[@dis-ext:∀] = {@minor = {"small", "minor", "limited"};
@major ={"significant", "extensive", "large", "major"} }

Such structures characterize relationships between the head word and other constituents of the phrase.

11 Fuzzy Matcher

A fuzzy matcher is a tool which using a sophisticated pattern-matching language extracts text fragments at various levels of exactness and shows them to the knowledge engineer. If the knowledge engineer wants to check whether a generalization of particular lexico-semantic patterns is right s/he can use the fuzzy matcher. It matches patterns which represent hypotheses of the knowledge engineer in the text, groups together and generalizes found cases and presents them to the knowledge engineer for a final decision.

Patterns themselves can be quite complex constructions which can include strings, words, types, precedence relations and distance specifiers. In the simpliest case the knowledge engineer can examine a context for occurrences for a word or a type provided that the type exists in the term bank. Here is an excerpt from a search for the type DISEASE with a distance four to the left and two to the right :

developed an anterior myocardial	infarction	from which
an established inferior myocardial	infarction	. The
an acute inferior myocardial	infarction	with CHB
subsequent episodes of unstable	angina	including an
he has experienced unstable	angina	and was

More complex patterns can be used for the description of complex groups. For instance, there a request can be made to find all co-occurrences of the type DISEASE with the type BODY-COMPONENT when they are at the same structural group (noun phrase or verb phrase) and the disease is a head of the group:

{[disease].<>[body-component]}

curly brackets impose a context of a structural group, the "." means that the words can be distributed in the group, <> means that the body-component can be both to the left and to the right from the disease, and since the DISEASE is the first element of the pattern it is assumed to be the head.

The program matches this pattern into the following entries:

myocardial infarction, infarction of myocardium, stenosis at the origin of left coronary artery...

Eventually all words that can occur in the same group with a disease can be added to the pattern and further this pattern can be conceptually characterized as was shown above.

Another pattern can specify co-occurrence of the previous pattern with a word "evidence" and a word "of":

"evidence" "of"..>{[disease].<>[body-component]}

"..>" here means that the disease phrase must be in the same sentence to the right of the word "of". The result of this search is:

evidence of heart failure
evidence of myocardial infarction
evidence of *a small* diaphragmatic infarction
evidence of *left* atrial thrombus

however phrases like "evidence of the antero-septal infarction" failed to be matched because they don't include a body-component constituent. So in the "disease" pattern we need to specify that a body-component is optional: {[disease].<>[body-component?]} .

To be powerful enough for our purposes this pattern language should be quite complex and it is important to provide an easy way for specification of such patterns with a question-guided process.

12 Conclusion

The knowledge engineering workbench outlined in this paper encompasses a number of computational tools which facilitate different stages of knowledge extraction, analysis and refinement based on corpus processing paradigm. These tools are integrated into a coherent workbench with a common user interface and a common inter-module data flow interface based on SGML. Thus the workbench can easily integrate new tools and upgrade existing ones.

The general approach to knowledge acquisition supported by the workbench consists of a combination of methods used in knowledge engineering, information retrieval and computational linguistics. This approach has several attractive features for practical extraction of information from natural language texts.

- The models are always targetted to a particular corpus. This means that the models themselves are very closely tied to domain specific features of the texts describing the domain, and hence a variety of statistical procedures have been developed precisely to extract this domain specific information. Knowledge engineers can use these procedures more or less directly to find common domain specific terms.
- Although the models encompass many of the regularities found in natural language, building the models from a corpus is a fairly automatic process without the need for detailed linguistic knowledge beyond that which is taught in high schools.
- The models are robust. Since no assumptions are made about the grammaticality of the texts being processed, every sentence gets analyzed whether or not it includes a grammatical error. Since knowledge engineers are not concerned whether information is conveyed in a grammatically correct sentence, this NLP technology allows a wide coverage of texts in the domain.

Knowledge extracted and organized with the workbench is effectively application independent and can be used in many fields ranging from natural language processing to expert system design and information retrieval. Moreover, due to the ability of the workbench to encompass new modules

there is no restriction on the language itself. Thus, for example, knowledge extraction can be performed from several different languages provided that the workbench is equipped with language specific tools. This makes the workbench particularly useful for multi-lingual term acquisition in design of machine-translation systems.

References

[1] A. Bech, A. Mikheev, M. Moens, and C. Navarretta. Typology for Information Contents. ET-12 Project Report 2, EC, 1993.

[2] A. Bech, M. Moens, and C. Navarretta. Strategies in NLP knowledge engineering. ET-12 Project Report 1, EC, 1993.

[3] B. Boguraev and T. Briscoe, editors. *Computational Lexicography for Natural Language Processing*. Longman Group Limited, UK, and Wiley and Sons, Inc., NY, 1989.

[4] P.F. Brown, V.J. Della Pietra, P.V. deSouza, J. C. Lai, and R.L. Mercer. Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18(4):467–479, 1992.

[5] J. Cowie, J. Cowie, and L. Guthrie. Lexical Disambiguation using Simulated Annealing. In Morgan Kaufmann, editor, *Proceedings of the Speech and Natural Language Workshop*, pages 238–242, Harriman, New York, 1992. DARPA.

[6] J. Diederich, I. Ruhman, and M. May. KRITON: a knowledge- acquisition tool for expert systems. *International Journal of Man- Machine Studies*, 26(1):29–40, January 1987.

[7] C. Feng, T. Copeck, S. Szpacowicz, and S. Matwin. Semantic Clustering. Acquisition of Partial Ontologies from Public Domain Lexical Sources. In *Proceedings of the 8th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, pages I, 5–17, Banff, Alberta, Canada, January – February 1994.

[8] S. Finch. A Methodology for Exploiting Sophisticated Representations for Classification. In *Proceeding of Intelligent Multimedia Retrieval Systems and Management (RIAO) 94*, NY, October 1994. Rockerfeller University, (forthcoming).

[9] J.A. Guthrie, L. Guthrie, Y. Wilks, and H. Aidinejad. Subject-dependent Co-occurrence and Word Sense Disambiguation. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 146–152, University of California, Berkeley, CA, USA, June 1991. ACL, ACL.

[10] V. Hatzivassiloglou and K. R. McKeown. Towards the Automatic Identification of Adjectival Scales: Clustering Adjectives According to Meaning. In *ACL Proceedings, 31st Conference*, pages 172–182, Columbus, Ohio, USA, 1993.

[11] M.A. Hearst. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of COLING-92*, volume II, pages 539–545, Nantes, august 1992.

[12] L. Hirshman. Discovering sublanguage structures. In R. Grishman and R. Kittredge, editors, *Analyzing Language in Restricted Domains*, Sublanguage Description and Processing, pages 211–234. Lawrence Erlbaum Associates, Hillsdale, N. J., 1986.

[13] Y. Huizhong. A New Technique for Identifying Scientific/Technical Terms and Describing Science Texts. (An Interim Report). *Literary and Linguistic Computing*, 1(2):93–103, 1986.

[14] R.E. Kent. Implications and Rules in Thesauri. In H. Albrechtsen and S. Ørnager, editors, *Knowledge Organization and Quality Management - Proceeding of the Third International ISKO Conference, Copenhagen, Denmark*, number 4 in Advances in Knowledge Organization, pages 154–160, Frankfurt/Main, Germany, June 1994. ISKO, Indeks Verlag.

[15] J.D. Lafferty and R.L. Mercer. Automatic Word Classification Using Features of Spellings. In *Making Sense of Words, Proceedings of the 9th Annual Conference of the* UW *Centre for the New* OED *and Text Research*, pages 89–703, Oxford, England, September 1993.

[16] R.-L. Liu and V.-W. Soo. An Empirical Study on Thematic Knowledge Acquisition Based on Syntactic Clues and Heuristics. In *ACL Proceedings, 31st Conference*, pages 243–250, Columbus, Ohio, USA, 1993.

[17] C.D. Manning. Automatic Acquisition of a Subcategorization Dictionary from Large Corpora. In *ACL Proceedings, 31st Conference*, pages 235–242, Columbus, Ohio, USA, 1993.

[18] A. Mikheev and C. Navarretta. Implementation Report. ET-12 Project Report 6, EC, 1993.

[19] A. Mikheev and C. Navarretta. Revised Methodology for Knowledge Engineering. ET-12 Project Report 4, Interim version, EC, 1993.

[20] A. Mikheev and C. Navarretta. Revised Methodology for Knowledge Engineering. ET-12 Project Report 4, Final Version, EC, 1994.

[21] S. Montemagni and L. Vanderwende. Structural Patterns vs. String Patterns for Extracting Semantic Information from Dictionaries. In *Proceedings of COLING-92*, volume II, pages 546–552, Nantes, august 1992.

[22] C. Navarretta. Criteria for 'support material'. ET-12 Project Report 3, EC, 1993.

[23] C. Navarretta. Evaluation of Existing Reusable Resources. ET-12 Project Report 5, EC, 1994.

[24] B. Nkwenti-Azeh. The Use of Thesaural Facets and Definitions for the Representation of Knowledge Structures. In H. Albrechtsen and S. Ørnager, editors, *Knowledge Organization and Quality Management - Proceeding of the Third International ISKO Conference, Copenhagen, Denmark*, number 4 in Advances in Knowledge Organization, pages 374–381, Frankfurt/Main, Germany, June 1994. ISKO, Indeks Verlag.

[25] F.P. Pereira, N. Tishby, and L. Lee. Distributional Clustering of English Words. In *ACL Proceedings, 31st Conference*, pages 183–190, Columbus, Ohio, USA, 1993.

[26] J. Pustejovsky. The Acquisition of Lexical Semantic Knowledge from Large Corpora. In Morgan Kaufmann, editor, *Proceedings of the Speech and Natural Language Workshop*, pages 243–248, Harriman, New York, 1992. DARPA.

[27] S.Y. Sedelow and W.A. Sedelow. Thesauri and Concept-Lattice Semantic Nets. In H. Albrechtsen and S. Ørnager, editors, *Knowledge Organization and Quality Management - Proceeding of Third International ISKO Conference, Copenhagen, Denmark*, number 4 in Advances in Knowledge Organization, pages 350–357, Frankfurt/Main, Germany, June 1994. ISKO, Indeks Verlag.

[28] J. Sinclair. Prospects for automatic lexicography. To appear in "Lexicography. Series Maior", Copenhagen, 1994.

[29] P. Vossen, W. Meijs, and M. den Broeder. *Computational Lexicography for Natural Language Processing*, chapter Meaning and structure in dictionary definitions, pages 171–192. Longman, UK, 1989.

[30] J. Wilks, D. Fass, C-M. Guo, J. McDonald, T. Plate, and B. Slator. *Computational Lexicography for Natural Language Processing*, chapter A tractable machine dictionary as a resource for computational semantics, pages 193–228. Longman, UK, 1989.

[31] D. Yarowsky. Word-Sense Disambiguation Using Statistical Models of Rogets' Categories Trained on Large Corpora. In *Proceedings of COLING-92*, volume II, pages 454–460, Nantes, august 1992.