Towards Developing Reusable NLP Dictionaries

Pim van der Eijk and Laura Bloksma and Mark van der Kraan

Research Institute for Language and Speech
Foundation for Language Technology
State University of Utrecht
The Netherlands
vandereijk@let.ruu.nl

Abstract

Development of reusable dictionaries for NLP applications requires a carefully designed lexicological framework, a lexical acquisition strategy, an integrated development toolbox, and facilities to generate dictionaries for client applications. This paper presents results of the LEXIC project[1], which was set up to prepare the development of large multilingual lexical resources.

Keywords: lexicons, tools, large-scale resources, typed feature structures.

1 Introduction

1.1 Common Linguistic Resources

A large amount of the investments in the development of any NLP application is spent on the construction of what one might call "large databases of lexical and grammatical resources". These resources could in principle be useful for many applications although they hardly ever are: due to the lack of agreement on the definition of basic notions and of consensus on the analysis of linguistic phenomena they are often linked too closely to specific applications. Moreover, given the generally limited size and duration of NLP projects both quantity and quality of such project-specific databases are disappointing.

In this paper we will discuss results from the LEXIC project, a feasibility study preparing large-scale development of a reusable lexical database, started by a consortium of industrial and university partners. The lexical database is designed to consist of an integrated package of two monolingual dictionaries for Dutch and Spanish and the bilingual dictionaries relating these languages. The consortium comprised a dictionary publisher as well as NLP application developers, giving it the unique opportunity of confronting the large body of experience, infrastructure and existing data of publishers with the requirements of a new class of professional users.

Another interesting aspect of the project was that it addressed the *whole* spectrum of issues in lexical database development, from lexical acquisition to serving heterogeneous client applications. In the current absence of any standard for the (grammatical) content of the dictionary (e.g. standardized sets of grammatical features) the *reusability* of a dictionary can only be evaluated in terms of *usability* for *some* target applications.

1.2 Structure of the paper

Section 2 discusses the issue of *acquisition* of lexical data. Section 3 introduces the implementation formalism and tools. The lexicon architecture is discussed in section 4. Conversion of data to client applications of the database is discussed in section 5.

2 Acquisition

2.1 Strategies

There are three potentially useful strategies to develop large lexical resources, which are not in principle mutually exclusive.

MRDs The extraction of data from machine-readable dictionaries has received much attention in the past decade. In our view the usefulness of existing material for NLP application has been somewhat overestimated. Traditional dictionaries are oriented towards a market of human consumers, who consult the dictionary for entirely different reasons than NLP applications. For instance, most of the information in NLP dictionaries is concerned with the grammatical description of

[1]The Lexic project was financed and supported by the three project partners: Philips Research, developing the *Rosetta* machine translation system, the Foundation for Language Technology, participating in the *Eurotra* project, and Van Dale, one of the main dictionary publishers in the Netherlands, as well as by the the European Commission, and the Dutch ministries of Education and Economic Affairs. Details of the project are discussed in [van der Eijk *et al.*, 1991].

The authors want to thank Anne van Bolhuis, Joy Herklotz, Jeroen Fokker and Tim Dumas for contribution to the activities discussed in this paper.

words, which in many dictionaries is only rudimentarily available[2].

Furthermore, given that humans can use their intelligence and knowledge of the language(s), much information is only present in unformalized definitions and examples. As discussed in e.g. [McNaught, 1988], it is often feasible to extract (relatively) formalized information, but the cost-effectiveness of automatic extraction of information from less formalized data is highly questionable.

From this discussion it follows that MRDs *alone* cannot be the source for NLP dictionaries. In section 2.2 we will discuss in more detail the evaluation of the potential sources of data for our specific purposes.

Corpora Automatic extraction of lexical features by applying various pattern recognition techniques to large bodies of text has received some attention recently (cf. e.g. [Zernik and Jacobs, 1990]). However, the information needed for our applications cannot be extracted from corpora yet, although important improvements can be expected in the following years.

Lexicography Given the present inadequacy of MRDs and corpus-related tools, manual labour is indispensable for lexicon development. The tools described in section 3 have been developed as a 'workbench' to support these lexicographical activities. We will show that this tool allows for easy integration of information extracted from MRDs with lexicographic editing.

2.2 Sources

Evaluation Measure It is difficult to assess the "reusability" of existing data without an evaluation measure, i.e. without knowing *for what purpose* the data should be usable. This is especially difficult in the case of grammatical features. We developed a lexicon fragment (implemented as TFS type hierarchy, cf. section 3) defining the classification scheme for the monolingual dictionaries. This fragment is inspired by HPSG and GB, and incorporates many of the (innovative) distinctions developed by the client applications EUROTRA and ROSETTA. It is, however, much more *lexicalist* than these systems.

Eventually, all lexical entries in the two languages should be described using this scheme, so that they can be readily converted to client applications. The data that can be extracted from a potential source has been interpreted with respect to this classification scheme to assess the amount of information contained in it.

Data Analysis The machine-readable sources we considered are the existing Van Dale Dutch monolingual and bilingual Dutch-Spanish machine-readable dictionary and the CELEX lexical database. From our evaluation it followed that existing MRDs for Dutch (as for almost all other languages) contain only a small part of the information needed by NLP applications.

Fortunately, the CELEX lexical database has enriched a selection of 30000 entries of the "Van Dale Dictionary of Contemporary Dutch" with grammatical information, taking into account the requirements of a number of (prototype) NLP applications under development in the Netherlands. A large amount of information needed for our target applications can be converted automatically from this database. The entries, stored in a relational database, can be imported into the Dutch lexicon using the TFS constraint solver similarly to the conversion to client applications (see section 5). The CELEX dictionary has historic links to the Van Dale dictionaries (especially with respect to reading distinction), which greatly simplifies integration of these sources.

With respect to translation information we found that the "raw" translational data could be extracted easily from the Van Dale bilingual dictionaries. The original Van Dale concept is especially interesting for multilingual applications, as the Dutch part is the same (at least in principle) in all bilingual dictionaries with Dutch as source language (cf. [van Sterkenburg *et al.*, 1982]).

Extraction of information about phrasal translation, such as the choice of the support verb of a noun in the target language, is unfortunately hidden in unrestricted text (example sentences etc.), from which it is difficult to extract. Phrasal information also suffers greatly from incompleteness.

3 The TFS Formalism

Before discussing the proposed lexicon architecture we will introduce the computational framework in which it has been formalized and implemented, the formalism of *typed feature structures*.

Currently the family of unification-based formalisms is an emerging standard as the implementation formalism of natural language processing systems. A variant called *typed feature structures*, discussed a.o. in [Carpenter, 1990], [Emele and Zajac, 1990] and [Zajac, 1990], has been adopted in a number of European lexicon projects, including ACQUILEX, EUROTRA 7 and MULTILEX. In the course of our project, a TFS database, user interface and a constraint solver have been implemented.

TFS is an excellent formalism for computational lexicons, as it enables a definition of *types*, or classes, of linguistic objects, arranged in a multiple inheritance hierarchy, where types are associated with an appropriateness specification defining their features and the types of those features and with (possibly disjunctive and complex) constraints. The object-oriented character of the system allows for minimization of *redundancy*, whereas the type system maximizes *integrity* of data.

Three TFS-based tools have been developed:

- a tool for interactive definition[3], entry and modification of data (cf. section 3.1).
- a TFS *database* which can be accessed from the user interface and the constraint solver.

[2] Well-structured dictionaries like [Longman, 1987] are an important exception to this, cf. [Boguraev and Briscoe, 1989].

[3] The TFS-editor can be used to interactively define a type hierarchy, as such a hierarchy can be viewed itself as a typed feature structure, cf. [Fokker, 1992].

- a TFS-compiler for data *manipulation*, e.g. selections and conversion.

The TFS-compiler is similar to the systems described by [Carpenter, 1990], [Emele and Zajac, 1990], and [Franz, 1990], and like these it constitutes a general-purpose constraint-based formalism which can be used for a wide variety of tasks, including parsing, translation and generation. Our prototype is implemented on top of Sicstus Prolog, and is used primarily for selection and conversion of data. It offers a number of tracing and debugging facilities to assist in the design of type-hierarchies and during query-evaluation.

These three tools can import and export data in a special-purpose text format, which is useful for interchange and further processing. The acquisition tools for the Van Dale dictionaries and Celex can also generate their output in this format.

3.1 User Interface

The hierarchical definition of the grammatical types in TFS corresponds closely to a "decision tree" which the lexicographer traverses while editing a lemma. A graphical user interface has been developed by the computer science department of the State University of Utrecht ([Fokker, 1992]) which allows the user to narrow down the main type of the lemma (s)he is editing to a specific subtype and to subsequently edit the associated feature structure. For example, a lemma is refined from **ENTRY** to **VERB** to **DATIVE_VERB**, then *constraints* for this type are retrieved and the features and their substructures can be edited recursively.

Of course, only *appropriate* features are presented and can be edited, e.g. it is impossible to edit a feature *arg3* of an intransitive verb. While editing the value of a feature the editor creates a subwindow already positioned at the minimal type of this feature. E.g. while editing a verb, the feature *semantics* will already be positioned at the type **EVENT**, as this is the minimal type of this feature for verbs.

The editor includes a useful *help* facility which can be viewed as an on-line instruction manual: a help function exists for each choice point which describes a number of criteria and examples to help making the decision.

It will now be clear how lexicographic work using the decision tree model relates to importation of lexical data from existing sources, such as MRDs. These can be converted to partially edited lexical entries, so that the lexicographer doesn't have to start at the 'root' level (e.g. the choice point **ENTRY** in the example), but at an intermediate level (e.g. **VERB**). Further choices lead to more refined descriptions of the word. Like all errors, errors in the source dictionary can be corrected by moving back to a higher-level choice point in the hierarchy.

Completed entries, and also arbitrary substructures, can be named and stored in a database for future use as shared (sub)structures in other entries. Useful applications of this cross-reference mechanism are in morphology and for the implementation of synonymy (see 4.2). Compounds can be assigned a feature *tree* with features *left_daughter* and *right_daughter*, whose values are pointers in the database to their constituting parts.

The editor has been implemented in *C* using the Microsoft Windows 3.0 graphical interface. The program is designed to be easily portable, e.g. to X windows. The underlying database can be shared via a LAN. As the other tools, the database allows for import and export of feature structures in the interchange format.

The editor is designed specifically for the TFS *formalism*. However it can be used for *any* specific type hierarchy, as the definition of the type hierarchy is simply defined in a separate text file which is read by the program during start up. Hence, it is potentially interesting for the development of many other (NLP) dictionaries.

An interesting elaboration of the editor would be to add extra functionality for the lexicographer besides editing and viewing feature structures, such as facilities to consult various on-line dictionaries or text corpora.

4 Dictionary organization

Having introduced the computational framework we will proceed with the discussion of the organization of the dictionary[4]. The emphasis has been on two types of modularity:

1. Modularity of dictionaries and thesaurus.

   The general approach is to define clearly a number of abstraction levels (cf. section 4.1) in order to achieve easy connectability of the monolingual dictionaries via bilingual dictionaries. By generalizing bilingual *translation* to bilingual synonymy (or *equivalence*, cf. section 4.2) we can even separate semantic descriptions ("concepts") from the elements in which they are realized in languages. We will show how such conceptual dictionaries can be generated from bilingual dictionaries (4.3).

2. Modularity of grammatical description (cf. section 5).

   With respect to the linguistic content of the monolingual dictionaries (i.e. the grammatical description) we will discuss the use of typed feature structure constraints expressing relations between grammatical descriptions in various linguistic theories. This allows for a very flexible relation between various grammatical descriptions.

4.1 The monolingual dictionary

Word forms in a language, as found in text corpora, are associated with canonical forms according to lexicological conventions. In particular contexts they are associated with exactly one of a fixed finite number of *designations*[5]. In [Zgusta, 1971], two other "components" of meaning are distinguished besides *designation*, viz. *connotation* and *range of application*. Our (somewhat poor) working definition of *synonymy* is a relation between readings sharing designation only, both within a language and across languages (where it is traditionally called *equivalence*).

[4] This is a condensed summary of [van der Eijk, 1992a].

[5] Note that we adopt the approach of *discrete* readings, cf. [ten Hacken, 1990].

The relation between word forms and canonical forms is many-to-many: orthographic variants are mapped onto a single canonical form, and a single word form can be related to several lexical entries via inflectional rules[6]. The monolingual dictionary is a set of lexical entries, which are pairings of canonical word forms of a language and their designations, and in addition describe their grammatical properties.

As a result, a lexical entry should minimally have the two features *canonical_form* and *semantics*. The former feature has the simple type STRING, the latter, the description of the designation, has a complex value, possibly including semantic features, but minimally containing an identifying feature[7], as we want to make sure it will always be possible to interconnect the monolingual dictionaries via bilingual dictionaries. Apart from these two features, there will be other features for the description of the grammatical properties of the word.

The combination of *canonical_form* and grammatical description should allow for the complete and correct generation of all word forms and their associated feature structures. As our intended client applications have front ends for this purpose the database was not designed to be a full form dictionary; this could change, depending on the needs of future client applications.

The set of designations can be viewed as a thesaurus or "knowledge base"; the lexical entries are "pointers" from words into this knowledge base, and can be implemented as such in TFS.

The relation between canonical word forms and designations is also many-to-many, due to synonymy (several word forms related to the same designation) and lexical ambiguity (one word form related to several designations). In addition to this there will be alternations in the description because of alternative grammatical patterns. These alternations are implemented as TFS disjunctions.

4.2 The bilingual dictionary

Bilingual dictionaries can be viewed as a relation between words in two languages. The levels "word form", "lexical entry" and "reading" correspond to various degrees of granularity in bilingual dictionaries. Ideally, the bilingual dictionary relates lexical items between languages at the level of *readings*, though in practice most existing dictionaries refer to canonical forms or even to word forms in the target language. Furthermore, the source language side in bilingual dictionaries usually refers to readings different from the monolingually motivated ones, because they are *tuned* to the target language: two readings are not distinguished if they translate to the same word, or an additional reading is created for an additional translation. An exception is the original concept of the bilingual Van Dale dictionaries, where the source language reading structure of the bilingual dictionaries is based directly on the monolingual reading structure (cf. [van Sterkenburg *et al.*, 1982]).

An interesting approach to the bilingual dictionary would be to view it as describing pairings of bilingual synonyms. The advantage of this would be that

1. the dictionary supports preservation of meaning in translation.
2. formal properties of equivalence relations (e.g. transitive closure) can be exploited to automatically expand the dictionary.
3. coding efforts can be reduced: the definition of the designation can be shared between monolingual and bilingual synonyms.

The main difference between traditional dictionaries and our approach is therefore that the *indirect* translational description of bilingual synonymy is replaced by a *direct* relation between lexical entries in the monolingual dictionaries to an independent "knowledge base" of synonym clusters. This approach is common in e.g. multilingual terminology (cf. [Picht and Draskau, 1985]), but less common in lexicology.

We will show that the two representations can be translated into each other. Section 4.3 describes how a knowledge base is generated from monolingual and bilingual dictionaries. A bilingual dictionary can be generated automatically from a set of monolingual dictionaries and a knowledge base by enumerating the pairs of lexical entries in two monolingual dictionaries pointing to the same synonym cluster.

4.3 Generating Synonym Clusters

Existing machine-readable bilingual dictionaries[8] can be converted to a representation based on bilingual synonymy, by "extracting" the underlying concepts. The process consists of the following steps:

First, the dictionaries are parsed and transformed to a table **synonym** of the relation between a reading R_1 in a language L_1 and a reading R_2 in L_2. Two versions of this program have been developed and tested: one for the Van Dale Dutch-Spanish dictionary and one for bilingual entries in the EUROTRA transfer rule format. A version for dictionaries in a standard interchange format would be a possible future extension.

Second, reflexive, symmetric, and transitive closure is applied to the **synonym/4** relation[9]. For each reading the generated synonym cluster can be viewed. E.g. according to the Van Dale Dutch-Spanish dictionary, reading `0.1` of Dutch **eerbetoon** (English *(mark of) honour*) has one synonymous reading in Dutch and three synonyms in Spanish.

```
eerbetoon_0.1 :
```

[6] E.g. the Dutch word form *bekend* is associated with the adjective *bekend* (meaning *well-known*) and (by participle formation) to the verb *bekennen* (*to confess*).

[7] The *name* of stored semantic substructures in the TFS database serves this purpose.

[8] Actually, there is no restriction to a *bi*lingual dictionary: several bi- or multilingual dictionaries, and even monolingual dictionaries of synonyms, can be processed similarly, resulting in a *multi*lingual dictionary. This has been checked using several Eurotra transfer dictionaries.

[9] This program was first implemented in Prolog for the Ndict system ([Bloksma *et al.*, 1990]) and modified for a Eurotra research group on "Reversible Transfer".

```
es: { homenaje honores tributo }.
nl: { eerbetoon_0.1 eerbewijs_0.1 }.
```

The current implementation is not yet fully satisfying. Because there is no reading distinction on the Spanish side in the Van Dale N-S (only the Dutch words in the example are marked with a reading number, e.g. `0.1`), some clusters will get mixed up[10]. E.g. Spanish *fresco* as adjective means *fresh* and as noun *fresco*, though the program will currently not make this distinction.

```
fresco_0.1 :
        es: { fresco limpio refresco }.
        nl: { fresco_0.1 fris_0.1 }.
```

The program could of course be modified to use the grammatical information about the target word in the dictionary as reading distinguisher; the noun *fresco* would then never be confused with the adjective. This is undesirable in principle, however, as we do not want *syntactic* criteria to guide reading distinction. For instance, many adjectives in Romance languages have homophonous nominal counterparts, with identical morphology and semantics. We don't want to be forced a priori to distinguish separate readings for these two cases. Furthermore, well-known examples of category shift in translation (e.g. adverbs translating to verbs etc.) show it is impossible to attach a unique syntactic category to an equivalence class.

These presentations of synonym clusters can be very helpful to interactively improve transfer dictionaries: errors of this type can easily be detected by native speakers of the languages (who need not know the other language) and corrected by creating appropriate reading distinction in Spanish.

We checked the quality of the synonym clusters generated from from both Van Dale and a EUROTRA Spanish-Dutch dictionary. The Eurotra dictionary, where both source and target language items are referred to at the *reading* level, was converted to over 2187 clusters, 315 of which contained more than one Spanish reading. Native speakers agreed with more than 95% of these synonym sets generated via the bilingual closure step. The interpretation of bilingual translation as *synonymy* is therefore correct in the vast majority of cases.

However, exceptions exist, such as the translation of the Spanish *reloj*, which, even though a true (and infrequent) Dutch synonym exists (viz. *uurwerk* (cf. English *timepiece*)), more commonly translates to one of its hyponyms *horloge* (Eng *watch*) or *klok* (Eng *clock*).

An interesting elaboration of our approach would be to extend the knowledge base by ordering the synonym clusters themselves via *hyponomy*[11] (cf. [Cruse, 1986], [Lyons, 1977]). Client applications could then extract translational data based not only on synonymy but also on hyp(er)onymy. However, this is a difficult area, where no obvious solutions exist. It is not clear at all which translation solution automatic translators should select in cases like this anyway.

After this correction process the synonym clusters can be converted to TFS format and stored in the database. The associated monolingual dictionaries are then modified automatically by adding cross-reference information (via the feature *semantics*) from the lexical entries to the synonym clusters they are associated with.

4.4 Creating a knowledge base

Synonym clusters really become descriptions of designations once semantic information is added to the synonym clusters, which is then, in a truly *interlingual* way, shared between synonyms. Much semantic information from the CELEX Dutch dictionary can be moved to the synonym clusters, as well as Van Dale definitions of concepts in natural language. The latter are useful for semi-automatic interactive applications[12].

The current approach can be said to implement the approach of *possible* bilingual lexical *translation*. This approach should be developed in a number of ways. Apart from the problem of translation to non-synonyms we mentioned, it is desirable to include information in the dictionary to guide the choice among possible translations, in cases where there are several synonyms in the target language. Stylistic, collocational and frequency information can be of use for this purpose. This information is partly available from existing sources (such as CELEX and Van Dale), and large text corpora are also obviously relevant sources of this information.

5 A model for conversion

Conversion or exchange of lexical data presupposes a detailed comparison of the various dictionaries, which in turn requires a careful description of the various dictionaries. Given the purpose of comparison, the descriptions should be cast in a uniform, preferably high-level data description language. Several such languages exist, such as the Entity-Relationship model, a tool in database design. We will use the TFS formalism introduced in section 3 for this purpose.

A first step in this comparison is to convert various dictionaries to the uniform TFS format. In most NLP formalisms lexical entries are records or feature structures, so this *syntactic* transformation is generally unproblematic. In passing, implicit *semantic* structure in the various dictionaries (e.g. feature cooccurrence restrictions) can be rendered explicit by constructing a type hierarchy for these systems.

On the basis of these descriptions, constraints on the relation between lexical entries in the different dictionaries can be defined. These constraints can be called

[10] The problem of connecting word forms to their readings has been called the *mapping* problem. Cf. [Byrd *et al.*, 1987] for discussion of a method to map word forms to readings by comparing a.o. semantic features like *human* of the source reading and potential target readings.

[11] This idea is similar to Wordnet, a collection of synonym sets linked via a variety of lexical relations ([Beckwirth *et al.*, 1989]). Our approach extends this idea by adding a multilingual dimension. Wordnet's synonym sets are also related by relations with less clear translational consequences. Also see [Calzolari, 1990] for a proposal similar to ours to integrate the dictionary and the thesaurus.

[12] For example, Rosetta incorporates an interactive reading selection facility.

semantic, as they relate the *content* of the various dictionaries, and *neutral* as they merely pinpoint correspondences between dictionaries; they define the way dictionaries (which may be unrelated in other respects) are similar.

Constraints can be viewed as implicational and biconditional constraints (as in [van der Eijk, 1992b]), and it is possible to implement them as a complex TFS type. This type serves both as documentation of the dictionary and as conversion specification.

A conversion specification is a TFS type CONVERT having features for each of the dictionaries (e.g. *lexic, eurotra* and *rosetta*), and establishes the basic conversion relation between entries in the LEXIC dictionary (as derived from the sources and augmented by lexicographers) and entries in the EUROTRA and ROSETTA dictionaries. This conversion type is structured hierarchically as well: the high-level type CONVERT has many subtypes specifying how specific subtypes (and hence subsets of the respective lexicons) of the various dictionaries are related. Disjuncts in the constraints of these types enumerate corresponding patterns described as feature structures.

An advantage is that these conversion constraints can be defined at the appropriate level of abstraction. It is in principle possible to establish relations holding for *all* entries as well as for an individual entry. As the conversion types are also ordered in an inheritance hierarchy, subtypes will inherit the constraints of their supertype(s).

Note the inherent *declarative* character of the conversion constraints: there is no notion of 'input' and 'output'. One advantage of this is that a single formalism can be used for importation, generation as well as integration of lexicons. A second advantage is that the conversion constraints can also be used to test whether two existing dictionaries are related as postulated in the conversion constraints.

Full *derivability* of a particular dictionary can be viewed as a special case of the general (in principle relational) scheme, where the substructure of a feature like *rosetta* is fully (and functionally) derivable from the substructure of another (*lexic*). Informally, all primitive distinctions in the target dictionary can be computed given the information in the source dictionary, i.e. the constraints define a homomorphism from the serving dictionary to the client application.

It is an empirical issue whether this derivability relation can actually be defined between two dictionaries. For newly to be created "generic" lexicons, this derivability is a design requirement. For the client dictionaries we have had to look at, creation of a generic source appeared to be a complex, but feasible, task.

Operationally, conversion proceeds as *query*-evaluation. Given an appropriate definition of the CONVERT type, the solutions to the following query will find all lexical entries whose canonical form is *fiets* in the LEXIC database and return all corresponding further instantiations of the ROSETTA type.

These instantiations correspond to the ROSETTA descriptions for this lexical entry.

$$\left[\begin{array}{l} \texttt{CONVERT} \\ \textit{lexic} : \left[\begin{array}{l} \texttt{ENTRY} \\ \textit{canonical\_form} : \textit{fiets} \end{array}\right] \\ \textit{rosetta} : \texttt{ROSETTA} \end{array}\right]$$

6 Illustration

We will illustrate conversion using the example in [van der Eijk, 1992a] relating two familiar linguistic theories, GPSG and a unification variant of Categorial Grammar, rather than the LEXIC fragment and ROSETTA, which we actually implemented.

The categorial lexical entries have a feature *subcat* whose value is either a CATEGORY or a FUNCTION. The type FUNCTION has appropriate features *argument*, (with two features *direction* and *category*), and *result*, where the result can be either a function again or a CATEGORY. Individual lexical entries are simply instances of this highly general recursive scheme. E.g. the *subcat* feature of a transitive verb (i.e. (NP\S)/NP) has type FUNCTION, with an NP argument to the right and, recursively, a FUNCTION from a subject NP to an S as *result*.

In GPSG individual lexical entries also have a feature *subcat*, but its value, an integer, is used to select the corresponding context-free grammar rule for this complementation pattern.

One of the disjuncts of the constraints for the CONVERT type will then be the following. Unifying specific categorial entries into the *cg* substructure will cause the corresponding *gpsg* feature to become instantiated.

$$\left[\begin{array}{l} \texttt{CONVERT} \\ \textit{cg} : \left[\begin{array}{l} \textit{subcat} : \left[\begin{array}{l} \texttt{FUNCTION} \\ \textit{arg} : \left[\begin{array}{l} \textit{dir} : \texttt{RIGHT} \\ \textit{cat} : \texttt{NP} \end{array}\right] \\ \textit{res} : \left[\begin{array}{l} \texttt{FUNCTION} \\ \textit{arg} : \left[\begin{array}{l} \textit{dir} : \texttt{LEFT} \\ \textit{cat} : \texttt{NP} \end{array}\right] \\ \textit{res} : \texttt{S} \end{array}\right] \end{array}\right] \end{array}\right] \\ \textit{gpsg} : \left[\begin{array}{l} n : - \\ v : + \\ \textit{bar} : 0 \\ \textit{subcat} : 2 \end{array}\right] \end{array}\right]$$

Due to the declarative character of TFS constraint evaluation, the above constraint will yield the same result whether the *cg*, the *gpsg* or both features are instantiated.

Evidently, the example is very simplistic. The prototype conversion module we developed in our project to translate LEXIC feature structures to ROSETTA feature structures contained over 500 disjuncts[13], and this module only covered conversion of a subset of the verbs. This number is caused by the fact that conversion rules

[13]This number results from expansion to disjunctive normal form. The actual notation for conversion rules allows for embedded disjunctions and is, hence, much more concise.

tend to become very idiosyncratic once the underlying theories of two dictionaries diverge.

7 Conclusion

We discussed how a multilingual lexical database can be constructed using a number of existing lexical resources and lexicography. The TFS formalism is very appropriate for the design and implementation of NLP lexicons. We showed that its hierarchical structure can be used profitably in a data entry tool which allows the lexicographer to manipulate feature structures graphically. Lexical acquisition from existing lexical resources can be combined seamlessly with lexicographic work.

The lexicon architecture we designed is an important improvement over earlier approaches: various abstraction levels and the mappings between them are defined more precisely, and the modularity is increased significantly by the separation of the knowledge base from language-specific dictionaries.

With respect to the issue of *reusability*, we outlined a framework for the specification of comparative description of linguistic encoding schemes. This specification can be used operationally as translation rules to convert lexical data.

References

[Beckwirth *et al.*, 1989] Richard Beckwirth, Christiane Fellbaum, Derek Gross, and George Miller. Wordnet: A lexical database organized on psycholinguistic principles. Paper presented at the First Lexical Acquisition Workshop, IJCAI89, 1989.

[Bloksma *et al.*, 1990] Laura Bloksma, Anne van Bolhuis, Pim van der Eijk, Pius ten Hacken, Joy Herklots, Dirk Heylen, Hans Pijnenburg, Frank Sesink, Anne-Marie Teeuw, Louis des Tombe, and Ton van der Wouden. Ndict: Final report. Technical report, Eurotra-NL, University of Utrecht, 1990.

[Boguraev and Briscoe, 1989] Bran Boguraev and Ted Briscoe, editors. *Computational Lexicography for Natural Language Processing*, London and New York, 1989. Longman.

[Byrd *et al.*, 1987] Roy Byrd, Nicoletta Calzolari, Martin Chodorow, Judith Klavans, Mary Neff, and Ommeya Rizk. Tools and methods for computational lexicology. *Computational Linguistics*, 13(3-4), 1987.

[Calzolari, 1990] Nicoletta Calzolari. The dictionary and the thesaurus can be combined. In *Relational Models of the Lexicon*. Martha Evens, 1990.

[Carpenter, 1990] Bob Carpenter. The logic of typed feature structures. Draft, 1990.

[Cruse, 1986] D.A. Cruse. *Lexical Semantics*. Cambridge University Press, 1986.

[Emele and Zajac, 1990] Martin Emele and Rémi Zajac. Typed unification grammars. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING)*, 1990.

[Fokker, 1992] Jeroen Fokker. Lemming user manual. Technical Report INF/DOC-92-04, Department of Computer Science, State University of Utrecht, 1992.

[Franz, 1990] Alex Franz. A parser for HPSG. Technical report, Laboratory for Computational Linguistics, Carnegie Mellon University, 1990. No. CMU-LCL-90-3.

[Longman, 1987] Longman. *Longman Dictionary of Contemporary English*. Longman House, Burnt Mill, Harlow, Essex, England, 1987. Second Edition.

[Lyons, 1977] John Lyons. *Semantics*. Cambridge University Press, 1977.

[McNaught, 1988] John McNaught. Computational lexicography and computational linguistics. *Lexicographica*, (4), 1988.

[Picht and Draskau, 1985] Heribert Picht and Jennifer Draskau. *Terminology: An Introduction*. University of Surrey, 1985.

[ten Hacken, 1990] Pius ten Hacken. Reading dictinction in MT. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING)*, 1990.

[van der Eijk *et al.*, 1991] Pim van der Eijk, Laura Bloksma, Anne van Bolhuis, Joy Herklots, Elly van Munster, Jeroen Fokker, Mark van der Kraan, and Angelique Geilen. Final report of the Lexic Project Phase I. Technical report, Foundation for Language Technology, 1991.

[van der Eijk, 1992a] Pim van der Eijk. Multilingual lexicon architecture. *Working Papers in Natural Language Processing, Katholieke Universiteit Leuven, Stichting Taaltechnologie Utrecht*, 1992. forthcoming.

[van der Eijk, 1992b] Pim van der Eijk. Neutral dictionaries. In Cheng-Ming Guo, editor, *Machine Tractable Dictionaries: Design and Construction*, chapter 6. Ablex, 1992. forthcoming.

[van Sterkenburg *et al.*, 1982] Piet van Sterkenburg, Willy Martin, and Bernard Al. A new Van Dale project: Bilingual dictionaries on one and the same monolingual basis. In J. Goetschalckx and L. Rolling, editors, *Lexicography in the electronic age*, pages 221–237. North-Holland, Amsterdam, 1982.

[Zajac, 1990] Rémi Zajac. A relational approach to translation. In *Proc. 3rd Int. Conf. on Theoretical and Methodological Issues in Machine Translation of Natural Language*, 1990.

[Zernik and Jacobs, 1990] Uri Zernik and Paul Jacobs. Tagging for learning: Collecting thematic relations from corpus. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING)*, Helsinki, 1990.

[Zgusta, 1971] Ladislav Zgusta. *Manual of Lexicography*. Mouton, 1971.