

Aspects of a Unification Based Multilingual System for Computer-Aided Translation

Oliver Streiter, Randall Sharp, Johann Haller,
Catherine Pease and Antje Schmidt-Wigger

IAI

Martin-Luther-Straße 14

D-66111 Saarbrücken

Germany

Tel. (0681) 39313

Fax (0681) 397482

email : catler@iai.uni-sb.de

AI'94

Fourteenth International Avignon Conference

Paris, May 30 - June 3, 1994

Natural Language Processing

Proceedings

Aspects of a Unification Based Multilingual System for Computer-Aided Translation

Oliver Streiter, Randall Sharp, Johann Haller,
Catherine Pease and Antje Schmidt-Wigger

IAI

Martin-Luther-Straße 14

D-66111 Saarbrücken

Germany

Tel. (0681) 39313

Fax (0681) 397482

email : catler@iai.uni-sb.de

Abstract

CAT2, a unification-based transfer-based Machine Translation system, has been developed as a sideline of the CEC-sponsored EUROTRA program. Together with a major software company this prototype is being validated within a user environment and extended with several user-oriented tools. In this paper we want to describe the basic extensions the system has undergone due to its industrial application, mainly addressing the demands of robustness and consistency. These modifications, as well as the overall architecture of the system, concern its internal linguistic conception, which has to satisfy two apparently contradictory demands, that of covering most of the linguistic phenomena of real-life text and at the same time remaining simple enough to be extended and maintained by a group of linguists.

Keywords

consistency, constraints, database, derivation, extended head features, lexical functions, machine translation, macros, robustness, unification

0 Introduction

The CAT2 Machine Translation System was first developed in 1987 as a sideline to Eurotra. From 1992 to 1993 the CAT2 system underwent a pilot study of industrial validation in cooperation with a major software company to test its possible application for commercial purposes. At the beginning of 1994 the CEC started the project CAT2-EDS which, scheduled for completion in July 1995, continues this pilot study in order to develop the system into a preindustrial prototype for the three languages English, French and German.

In the following four sections we want first to give a minimal overview of the CAT2 MT system (its formalism and basic architecture). Secondly, we describe the basic extensions the system has undergone due to its industrial application. Thirdly, we describe some aspects of the linguistic base component, which allows for the treatment of the syntactic and phraseological phenomena of real-life text without losing its internal intelligibility and thus the possibility of being easily maintained and ameliorated. In the last section we give a short overview of the future modifications we think are necessary to make the system practical for multilingual MT.

1 The CAT2 System

The CAT2 system has two basic parts, the CAT2 formalism and its implementation (the software) and the CAT2 lexica, grammars and translation modules (the lingware). The software can be subdivided into the core formalism, which is responsible for the interaction of the grammars and lexica, and a number of facilities (a) for developing the grammars, lexica and translation modules and (b) for its application as a professional MT system, enabling an appropriate environment for pre- and post-edition.

In this section we shall describe some aspects of the CAT2 system, in order to facilitate the reading of Section 2 and Section 3.

1.1 The Formalism

The formal properties of the system, described in [Sha91] [SS92] [Ha93] [Sha94] are summarized as follows: similar to the PATR-2 formalism, unification is its only computational mechanism, working on tree structures and on the feature structures annotated to every node of the tree. Unification may be constrained by negative, disjunctive or implicative constraints over simple and complex features.

Feature structures are represented in CAT2 by attribute-value pairs within curly brackets, representing unordered sets of features. Tree structures are represented using square brackets indicating lists, that is, an ordered set of daughters.

A non-atomic (i.e. phrasal) node is represented by a tree structure containing at least one daughter (a), whereas an atomic (i.e. lexical) node has no daughters (b):

- (a) $\{...\}.[\{...\}, \dots, \{...\}]$.
- (b) $\{...\}.[]$.

Feature structures consist of attributes and their related constraints on values, which can be simple (c) or complex (d):

- (c) $\{\text{attribute}=\text{value}, \dots\}$
- (d) $\{\text{attribute}=\{\text{attribute}=\text{value}, \dots\}, \dots\}$

The grammar rules describe valid structures for each level, and these are used to generate objects through unification. Two feature structures are said to 'unify' when they contain all the features in each of the two structures and do not contain any contradictory information. As a result of this unification the object inherits any additional constraints contained in the grammar rule. These constraints may be positive ($\{\text{attribute}=\text{value}\}$), negative ($\{\text{attribute}\neq\text{value}\}$), or disjunctive ($\{\text{attribute}=(\text{value}_1;\text{value}_2)\}$). Positive, negative

and disjunctive constraints can be used in any logically meaningful combination as in (e) and (f):

- (e) {attribute1~={attribute2=(value1;value2)}}
 (f) {attribute1={attribute2~=(value1;value2)}}

In addition, objects can be conditionally compelled to have a particular structure through using the implication operator. If the constraints within the condition part of the implication unify with the object, the constraints in the consequent must also unify with the object if the object is to be considered valid. If the condition does not apply, the consequent is ignored. The implication operator has the form (g) or (h):

- (g) {condition}>>{consequent}
 (h) {condition}>>({consequent1};{consequent2};...)

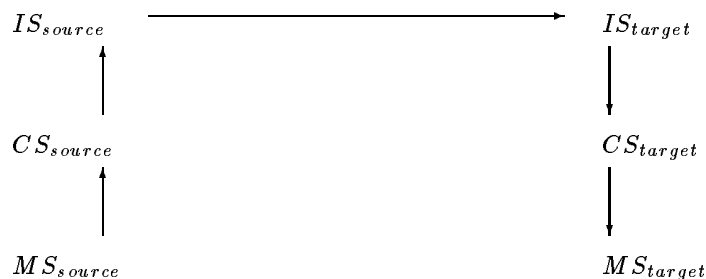
Following Prolog convention, logical variables beginning with uppercase letters can be used to bind two values within the object. Once one variable is instantiated by unification all other variables bound to it become equally instantiated, and share all the same constraints. Variables can refer to a complete node (j) in an tree structure, just as to a feature structure within a node (i).

- (i) {...}.[{attribute=VAR,...}, {attribute=VAR,...}].
 (j) {...}.[{attribute=VAR,...}, VAR].

In (j) the value of the attribute must unify with the adjacent node in the tree. Given such a construct, we can, for example, easily state subcategorization requirements in one element that must be fulfilled by an adjacent element. (We see such an application below.)

1.2 Architecture for MT

The basic architecture of the CAT2 system is that of a classic stratificational, transfer-based one, using tree structures at all levels. In our diagram MS refers to 'morphological structure', CS to 'constituent structure' and IS to 'interface structure'.



Following current sign-based approaches in linguistics (cf. [PS87]), syntactic and semantic analysis is done simultaneously at the constituent level (CS). By the joint cooperation of syntax and semantics, the creation of spurious analyzes due to structural ambiguities can be reduced. The IS level, having no linguistic justification other than that of serving for transfer, can be shaped and restricted according to these demands. In particular, the IS is designed to allow compositional translation of each of the major constituents, relegating the task of verifying the resulting target structure to the target CS generator. This makes the IS structure more efficient for transfer and more intelligible for the linguist/translator. The CS level and the IS level are connected by a number of general mapping rules which in analysis elide functional categories (determiners, prepositions, complementizers, auxiliaries etc...), undo every kind of "movement" (e.g. passivization, extraposition, etc.), and replace the syntactic predicates which are semantically empty (so-called light verbs and copulas) by the corresponding semantic predicate (the predicative noun or adjective). In generation these mapping rules perform the reverse, i.e. generate functional categories, move elements to surface position, and try to introduce dummy predicates wherever possible.

This symmetrical architecture allows for the reversibility of the system. For two reasons reversibility is an essential feature for an MT system: (a) minimal resources are needed to extend the system, a fact of economic importance, (b) translation itself is a process which presupposes competence in both languages. Transfer has to be underspecified as regards surface features (word order, choice of articles and prepositions, choice of morpho-syntactic voice, tense and aspect marking) if the transfer component is not to resemble that of a commercial kick-and-rush system. The abstracting away from surface features and the reliance on semantic and pragmatic aspects in transfer presuppose a fully competent language component which can relate the semantic and pragmatic content to its surface representations.

The high degree of abstracting away from surface features of language is attained by the inclusion of general cognitive categories relating to time, space and cause in the IS representation. This approach to transfer, located somewhere between a normal word-based transfer and an interlingua, without incurring the drawbacks of an interlingua (cf. [AS92]), allows us to keep transfer simple and at the same time maintain a linguistically interesting model, where the linguistic component is embedded in a more basic cognitive structure (cf. [Cho80]). The asymmetric demands an MT system is submitted to with respect to analysis and generation (e.g. analyze all variants of a sentence vs. generate only one variant) can be resolved by the separation of the CS \Rightarrow IS component from the IS \Rightarrow CS component, the latter by default generating only unmarked structures. In this way, the language-specific CS grammar, the most complex module, may remain reversible, thus applicable for analysis and generation.

2 Robustness and Coherence of the System

When we started to work in 1992 with the major software company, we began the exciting experiment of testing whether the academic approach to MT, characterized by keywords such as *logic programming*, *declarativity*, *constraint based* and *cognitive grammar*, may be validated in an industrial environment. In such an environment, an MT system is submitted to a number of demands which are irrelevant, for a system used in the context of research only. Our task was to isolate such problems and to find appropriate solutions, without losing the advantages of our system, those of simplicity, intelligibility and easy maintenance. It is one of the first attempts to bridge the gap between current academic research and the needs of an industrial user for whom user interfaces and usability of the results are of prime importance.

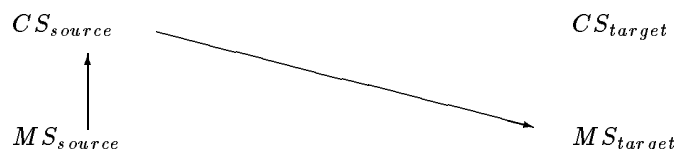
2.1 Robustness: The Fail-Soft Mechanism

One of the main problems we encountered was the lack of robustness in unification-based systems. Such systems traditionally do not have the possibility of gradually weakening their constraints in the case of an unsuccessful linguistic analysis. But given the creative usage that language is submitted to, cases of failing analysis cannot be excluded. Nevertheless such input should not cause the system to fail.

We established two basic maxims relating to the reliability of an MT system: (a) the system should communicate to the user when the input is not covered. (b) it must produce some output. The output may not be a correct translation, but it should be an attempt to present something useful to the user. In other words, the system must not simply fail, producing no output, but neither must it pretend to give perfect translations, thus deceiving the translator, who, once deceived, is charged with the extra burden of detecting those sentences among the output which *prima facie* seem to be successful translations, but which on further analysis reveal no meaning at all, or worse, transmit a false or misleading translation. In the end, such sentences offer little help to the translator, since the only aim of such a cheat-and-hide translation is to create something that looks like a sentence, disregarding its actual content. Our opinion was that the system should rely in such cases on the extensive results of lexical research and a consistent treatment of terminology (cf. [P62]), by suggesting to the users a translation of the content words and providing a

standard translation for function words. A direct use of such raw translations seems often to be quite possible.

Accordingly, we extended the simple architecture of the CAT2 system to include a fail-soft mechanism, which in the case of failing analysis or generation during normal translation reverts to a second strategy which performs a word-for-word translation. Specifically, if the system has constructed an object at the CS level, then the words in this structure, which may now be substantially disambiguated, are translated to words in the target language. If, however, not even the CS structure was created, then the words at the MS level are translated to the target language.



This approach has two major advantages. First, it is computationally simple, not deviating from the basic principle of the normal translation path, using the existing rule types based on unification. Secondly, the word to word translation allows the user to establish the correspondence between the words in the input sentence and the words in the output, which is not a trivial task, especially for sentences some lines long, when the system has moved words according to certain heuristics into unexpected positions.

2.2 Robustness: Treatment of Unknown Words

For the robustness of an NLP system it is absolutely necessary to cope with words unknown to the system. Proper names, numbers, acronyms, anglicisms (gallicisms, germanisms) and new words may appear in any text. No lexicon can be written that contains all these names, numbers and acronyms.

The treatment of unknown German words in the CAT2 system is performed by the morphological component, an independent Prolog state called *mpro* based on a morpheme list of more than 13000 morphemes (and their allomorphs and written variants). They include the major categories as well as prefixes and suffixes, all with extensive descriptions of their behaviour in word formation processes (derivation, compounding, and inflection) as well as their semantic characterization. Experiments have shown that nearly all incoming words are correctly analyzed with respect to inflectional morphology, and 90% are correct with respect to their semantic characterization. Properties concerning subcategorization can however be predicted only very rudimentarily.

If no morphological analysis is possible (for example in the case of acronyms and anglicisms), the lexical unit is preset to $\{lex='@unknown'\}$, and its syntactic category, in the absence of any counterindication, is regarded as a noun ($\{cat=n\}$). If, by the process of parsing, the unknown word can be integrated into the structure of the sentence, normal treatment of the whole sentence will take place. The unknown word will remain in its source language form, which is indeed required for proper nouns and numbers, and to a lesser extent for acronyms (e.g. UNO vs. ONU).

As mentioned above unknown words remain in their source form throughout the translation process. We completed this by a quickcoder with the help of which the user has the possibility of proposing a target language form which will replace the unknown word in the output sentence. The entries the user enters are then available for the entire translation session.

Relating to the treatment of unknown words we created a further tool for the automatic generation of a lexical entry for an unknown word once it has been analyzed, aiming at automatic lexical language acquisition. The so-called INTERCODER guesses the syntactic and semantic characteristics of an unknown word and proposes a coding for this entry. These provisional codings can be incorporated into the ordinary CAT2 lexica after manual verification and elaboration on the coding.

2.3 Coherence: Lexical Consistency Check

More than 50% of the errors in MT systems are caused by faulty and inconsistent lexical codings. Given the huge size of data to be checked regularly for consistency, we wanted to automate this task as much as possible, using the existing computational capacities of the CAT2 system.

We therefore created a procedure which automatically checks every lexical entry of the source language as to whether there can be found a corresponding lexical entry in the target language, which (a) is connected to the source language entry by a unifying transfer rule of the type `{lex=zustaendigkeit}=>{lex=responsibility}` and (b) which fulfills all translational equivalence conditions, that is to say, it checks the compatibility of the semantic characteristics of the lexical entry and its potential translation. If no translation can be found for a source language entry which satisfies both demands, the system returns a warning which specifies the transfer rule and the deviating features, which should be identical but are not. An example of such a warning is given below:

```
lex=zustaendigkeit => lex=responsibility
isde: {head={ehead={sem={temp={ext=_}}}}}
isen: {head={ehead={sem={temp=nil}}}}
```

In our example the German noun *Zuständigkeit* has no equivalent in the English lexicon, since although the word which is targeted by the lexical transfer rule (responsibility) is present, it is coded as having no temporal meaning, whereas the German entry indicates a temporal meaning.

2.4 External Dictionaries

Absolutely essential for an industrial application is the use of large dictionaries, numbering in the thousands of entries, as opposed to most experimental research systems which tend to use at most a few hundred carefully selected entries. In order to expand the lexical coverage in the CAT2 system, we are currently installing external databases for each language. These databases, like CAT2 itself, are coded in Prolog, thereby maintaining the advantages of a single coherent system, especially at this stage when the system, still a prototype, is undergoing industrial validation.

As part of the database, a special macro file is defined, similar to the M4 macros, in which all features and their allowable values are predefined. The user utilizes the macro file when entering new entries into the database, which guides the user in selecting the correct features, and thus prevents the entry of incorrect data.

A special-purpose database management system called CAT2/DBMS [SC94] has been constructed for the purpose of maintaining the CAT2 databases. It makes use of SQL-like commands to INSERT, DELETE, and SELECT entries, as well as commands to CREATE and DROP databases. With this facility, the CAT2 databases can be maintained independently of the machine translation system itself. Furthermore, there is no longer a limit to the sizes of the dictionaries, other than physical disk capacity.

3 Development of the Linguistic Core Component

3.1 Use of Common Modules

Often the same linguistic information must be available in different places of the NLP system. To give an example, the CS and IS levels make use of the same lexicon. General ("universal") grammatical rules (e.g. for testing the completeness of an argument structure or a general X'-rule) and common lexical entries (terminology, punctuation marks, etc...) may be used by different language components. In earlier versions of the CAT2 system all grammatical resources had to be copied into the modules that made use of them. Looking for a more economical solution, we now allow grammatical resources to be declared as a COMMON module. These common modules then can be called by any other module with

the CALL statement, just like a subprocedure or function in most ordinary programming languages.

The advantages of this solution are obvious: From a computational point of view, the common module is loaded once only into the memory. From a linguistic point of view, rules need be written and modified only once, which augments the consistency of the grammatical information. In addition, it allows for a parametrized approach to grammar, where the language-specific grammar is composed of universal rules, parameters of variation and language-specific rules, following the ideas developed by Chomsky [Cho81]. As a consequence grammars of new language components are written in a very short time by setting the parameters appropriate to the new language.

To give an example of common rules, we describe the three main types of building rules which are used in the English, French and German syntactic components. They are responsible for the detection of arguments, the subcategorization by functional categories and the detection of modifier relations.

3.1.1 Predicate–Argument Structures

Arguments to predicates are specified in the frame feature of each lexical entry. If no argument is specified, the value of ARG1, ARG2 etc... is NIL. Two general-purpose structure building rules (called “b-rules”) are responsible for the detection of argument relations during parsing. Since the b-rules in the CAT2 formalism not only specify dominance relations but also linear order of the constituents, the two b-rules necessary to express the predicate-argument relation include (a) when the argument precedes the predicate (e.g. in German), and (b) when it follows the predicate (e.g. in English). The b-rule responsible for argument detection to the right is reproduced here:

```
{head=HEAD} . [ {head=HEAD,frame=({arg1=ARG};{arg2=ARG};{arg3=ARG};{arg4=ARG})} , ARG ] .
```

The rule is read as follows: A lexical category which is the head of a projection (`{head=HEAD}`) and hereby sharing all head features with the mother node takes a constituent to its right as an argument if the constituent unifies with one of the descriptions in ARG1, ARG2, ARG3 or ARG4.

3.1.2 Head–Modifier Structures

A constituent is analyzed as a modifier of another constituent if the restrictions (`{restr=RESTR}`) imposed by the modifier unify with the description of the constituent being modified. Again, two rules accommodate modifiers to the left of the head and modifiers to the right; an example of the former is shown here:

```
{head=HEAD} . [ {role=mod,head={restr=RESTR}} , {head=HEAD}>>RESTR ] .
```

3.1.3 Functional Categories

In our implementation, we also assume that the functional categories form head projections. Following Grimshaw’s Extended Projections[Gri91], nouns combine with determiners to form extended noun projections, and prepositions combine with extended noun projections to form larger extended noun projections. In the same way, complementizers combine with verbs to form extended verb projections. As functional categories we include determiners and prepositions (for nouns), particles, auxiliaries and complementizers (for verbs), degree words (for adjectives and adverbs) and coordinators. All functional categories except coordinators have one argument position, coordinators have two. Here is the rule responsible for functional categories in initial position:

```
{head=HEAD} . [ {role=funct,head=HEAD,frame=({arg1=ARG};{arg2=ARG})} , ARG ] .
```

The features that relate to the entire extended projection are contained in the ehead (extended head) feature. For example, if a noun phrase contains the *+wh* feature (e.g. “which country”), then a prepositional phrase containing this noun phrase also contains the *+wh* feature (e.g. “in which country”). By definition, ehead features are a subset of head features.

This EXTENDED HEAD FEATURE PRINCIPLE, which applies to all functional categories (except coordinators) is implemented in the form of a lexical feature instantiation rule (“f-rule”), which unifies the ehead features of the functional category with the ehead features of the lexical head subcategorized for.

```
{role=funct,head={ehead=EH},frame={arg1={head={ehead=EH}},arg2=nil}}. [] .
```

Note that the constraint *arg2=nil* prevents this rule from being applied to coordinators. The features obligatorily shared by the coordinated structures and the coordination itself form a subset of the ehead features.

3.2 Simplification of the grammar: The Use of Extended Head features

Although head features and subcategorization features are regarded as standard in linguistic circles, extended head features are not yet part of mainstream linguistics. This is astonishing insofar as the extended head feature principle as formulated by Grimshaw[Gri91] seems to be a fundamental principle of natural languages, the implementation of which simplifies the representation of grammatical relations and allows for a higher degree of lexical control.

First of all, there exists an undeniable difference between head features and extended head features. *VFORM* (verb form, e.g. finite, nonfinite, participle, etc.) is a typical head feature which is available at the maximal VP projection of the verbal head. But this information is not passed up higher than the local VP, since higher VP projections may have different *VFORMS* (e.g. when an auxiliary with one *VFORM* takes a VP projection having a different *VFORM* as an argument, forming an extended VP projection). Information about tense and aspect, on the other hand, must be present at every point in an extended VP projection, passing the morpho-syntactic realization of tense and aspect up to S, where the adjunction of temporal modifiers must be controlled and where the tense-aspect information must agree with possible complementizers.

Further motivation for the use of extended head features comes from the predictions that lexical elements can make about their possible functional governors. Similar to Mel'čuk's lexical functions (cf. [Mel74]) we decided to allow nouns to predict their determiners and prepositions in the case of proper names (French: *Les Etats-Unis, Le Canada*), in support verb constructions, collocations (*to kick the/*a bucket*) and in spatial and temporal expressions (Russian: *na zavode, v teatre* "in the factory, in the theatre", French: *en hiver, au printemps, au ciel, en enfer* "in winter, in spring, in heaven, in hell"). In the following we show two lexical entries, which by the extended head feature principle can account for these facts:

```
{lex='Canada',head={cat=n,ehead={type={def=art}}}}. [] .
```

```
{lex=zavod,head={cat=n,ehead={({pvalue=loc,pform=(na;s)};{pvalue~=loc})}}. [] .
```

For a more detailed description of extended head features and their function within an MT system see [SS93].

3.3 Lexical Consistency: The Treatment of Derivation

Following the principle of argument-inheritance we have chosen to represent verbs and their nominal derivations in one lexical entry. Since the semantic and morphological types of nominalizations are idiosyncratic, they must be specified in the lexicon. The German verb *adoptieren* (to adopt) may form action nominalizations by using the infinitive (*das Adoptieren*) or the suffix *-tion* (*die Adoption*), but not with the suffix *-ung* (**die Adoptierung*).

Common to both the verb (*adoptieren*) and its nominalizations (*Adoption, Adoptieren*) are

the lexical unit ($_{\text{LEX=ADOPTIEREN}}$) and the subcategorization frame, where the arguments to be inherited are described.

Differences in the surface realization of arguments with respect to case marking or prepositional form depend on general principles (e.g. structural case assignment, *of*-insertion, adjacency restrictions). Thus case and preposition, if changing, are not entered into the lexicon, but are determined by lexical default rules.

The following lexical entry of *adoptieren* encompasses the lemmata *adoptieren*, *die Adoption* and *das Adoptieren*

```
{lex=adoptieren,head={pref=nil},
  head=({cat=v,deriv=nil,ehhead={sem={anim=nil,temp={ext=_}}}}
        ;{cat=n,deriv=ion,ehhead={sem={anim=nil,temp={ext=_}},gen=fem}}
        ;{cat=n,deriv=inf,ehhead={sem={anim=nil,temp={ext=_}},gen=neut,type={def=_},}),
  frame={arg1={role=agent,head={ehhead={cat=n,sem={anim=hum}}}},
        arg2={role=theme,head={ehhead={cat=n,pform=nil,sem={anim=hum}}}}}. [] .
```

By coding the relevant derivations together with the underlying base form in one lexical entry, the coherence of the subcategorization properties of the verb and its derived nouns is guaranteed.

In the future we intend to extend this principle to state derivations of nouns from adjectives (e.g. *high*⇒*height*) and to derivations of adjectives from verbs (e.g. *regret*⇒*regrettable*).

3.4 Lexical Performance: The Treatment of Compounds

Another problem of inefficiency the CAT2 system was faced with was the treatment of compounds. German compounds were originally treated in the same way as single words and simply entered in the lexicon as such, and unknown compounds (i.e. those not found in the lexicon) were then coded using the quick-coding facility. The frequent occurrence of compounds meant (a) a duplication of dictionary entries and (b) a great deal of time spent in quick-coding. We therefore decided to analyze compounds into their various components and translate them independently.

The morphological program used in CAT2 analyzes nominal compounds into their parts and gives as output the head of the compound (the rightmost component), which contains in its feature bundle a complete description of the non-head, as shown for the German compound *Regenschirm* (umbrella):

```
{lex=schirm,head={cat=n,...},clex={lex=regen,head={cat=n,...}}}. [] .
```

Between the CS and IS module, the compound is restructured into a tree structure containing the head (e.g. *Schirm*) and the non-head (e.g. *Regen*). The non-head is analyzed according to its role (i.e. whether it is an argument or a modifier of the head of the compound). This step of the analysis is essential, since the relation between the head and the non-head gives an important indication of how the compound should be translated. The non-head can only enter into an argument position if this position has not yet been occupied by another constituent. In *the government debate about finance*, for example, the second argument slot of *debate* is already filled by *about finance*, which allows *government* only to enter into the first argument slot.

In the transfer from source IS to target IS the decision is taken as to whether the original compound should become a compound in the target language, or whether it should be translated as a noun plus PP. This decision is made on the basis of, firstly, semantic information about the head, and, secondly, the prepositional form the non-head would receive in its modifier or argument position in the target language. If the source language non-head has to become a PP in the target language, all PPs attached to the head are examined as potential non-heads.

4 Conclusion and Further Developments

In what precedes we have shown some steps which have helped a research MT prototype become a pre-industrial prototype, without losing the advantages of a simple and intelligible system. The problem of robustness must be regarded as solved within the framework of unification, requiring no additional rule format. The problem of consistency of the growing lexica has been approached by automatic consistency tests, the use of macros and the linguistic treatment of derivation. The apparently contradictory demands, that of covering most linguistic phenomena of a real-life text and at the same time remaining simple in order to be extended and maintained, has been approached by the extensive use of common grammar components.

Further developments have to concentrate on the integration of larger dictionaries into the system, according to the needs of future development partners. Several such dictionaries, especially for the language pair German-English, are already available from former MT projects at the University of the Saarland.

Another area is the further improvement of the user interface where some results of research in ergonomics for translators will play an important role; a two-screen post-editing facility for parallel texts which enables input and output with multilingual characters will be ported to X under OSF/Motif and be upgraded to a demonstrator toolbox where the whole translation process as well as the intermediate stages can be optimally shown in a window environment.

Such developments however cannot be undertaken without the cooperation of large industrial users as in the well-known cooperation between CMU and Caterpillar; there are currently discussions with several such users concerning the possibility of making a "business case" out of the growing CAT2 system.

References

- [AS92] Doug Arnold and Louisa Sadler. Unification and machine translation. *Meta*, 37(4):657–680, décembre 1992.
- [B92] Claude Bédard. La prétraduction automatique. Outil de productivité et d'évolution professionnelle. *Meta*, 37(4):738–760, décembre 1992.
- [Cho80] Noam Chomsky. *Rules and Representations*. Columbia University Press, New York, 1980.
- [Cho81] Noam Chomsky. *Lectures on Government and Binding. The Pisa Lectures*. Studies in Generative Grammar 9. Foris Publication, Dordrecht Holland & Cinnaminson U.S.A., 1981.
- [Gri91] Jane Grimshaw. Extended projection. Brandeis University, Waltham MA 02254, ms, July 1991.
- [Hal93] Johann Haller. CAT2, Vom Forschungssystem zum präindustriellen Prototyp. In Horst P. Pütz and Johann Haller, editors, *Sprachtechnologie: Methoden, Werkzeuge, Perspektiven*, pages 282–303, Hildesheim, 1993. GLDV, Georg Olms AG.
- [Mel74] Igor Aleksandrovič Mel'čuk. *Opyt teorii lingvisticeskix modelej Smysl ⇔ Text. Semantika, sintaksis*. Izdatel'stvo " Nauka", Moskva, 1974.
- [PS87] Carl Pollard and Ivan Sag. *Information-Based Syntax and Semantics*. CSLI Lecture Notes 13, Stanford, 1987.
- [SC94] Randall Sharp and Michael Carl. A lexical database for the CAT2 machine translation system. Working paper (in preparation), IAI, Martin Luther Straße 14, 66111 Saarbrücken, BRD, 1994.
- [Sha91] Randall Sharp. CAT2: An experimental Eurotra alternative. *Machine Translation*, 6(3):215–228, 1991.
- [Sha94] Randall Sharp. *CAT2 Reference Manual Version 3.5*. IAI, Martin Luther Straße 14, 66111 Saarbrücken, BRD, January 1994.
- [SS92] Randall Sharp and Oliver Streiter. Simplifying the complexity of machine translation. *Meta*, 37(4):681–692, décembre 1992.
- [SS93] Oliver Streiter and Randall Sharp. *Linguistic Reference Manual of the CAT2 Machine Translation System*. Martin Luther Straße 14, 66111 Saarbrücken, BRD, December 1993.