Hermeto: A NL-UNL Enconverting Environment

Ronaldo Martins, Ricardo Hasegawa and M. Graças V. Nunes

Núcleo Interinstitucional de Lingüística Computacional (NILC)
Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação
Av. do Trabalhador São-Carlense, 400. CEP 13560-970. São Carlos – SP – Brasil
ronaldo@nilc.icmc.usp.br; gracan@icmc.usp.br; rh@nilcicmc.usp.br
http://www.nilc.icmc.usp.br

Abstract. This paper aims at presenting and describing HERMETO, a computational environment for fully-automatic, both syntactic and semantic, natural language analysis. HERMETO converts a list structure into a network structure, and can be used to enconvert from any natural language into the Universal Networking Language (UNL). As a language-independent platform, HERMETO should be parameterized for each language, in a way very close to the one required by the UNL Center's EnConverter. However, HERMETO brings together three special distinctive features: 1) it takes rather high-level syntactic and semantic grammars; 2) its dictionaries support attribute-value pair assignments; and 3) its user-friendly interface comprises debug, compiling and editing facilities. In this sense, HERMETO is said to provide a better environment for the automatic production of UNL expressions.

1 Introduction

In the UNL System [1], natural language (automatic) analysis has been carried out either by the EnConverter (EnCo) [2] or, more recently, by the Universal Parser (UP) [3], both provided by the UNL Center. In the first case, enconverting from natural language (NL) to Universal Networking Language (UNL) is supposed to be conducted in a fully-automatic way, whereas in the second case a full-fledged human tagging of the input text should be carried out before NL analysis is triggered. In both cases, results have not been adequate. EnCo's grammar formalism, as well as UP's tagging needs, are rather low-level, and requires a human expertise seldom available. In what follows, we present an alternative analysis system, HERMETO, developed at the Interinstitutional Center for Computational Linguistics (NILC), in Sao Carlos, Brazil, which has been used for automatic enconverting from English and Brazilian Portuguese into UNL. Due to its interface debugging and editing facilities, along with its high-level syntactic and semantic grammar and its dictionary structure, it is claimed that HERMETO may provide a more user-friendly environment for the production of UNL expressions than EnCo and UP.

The structure of this paper is as follows. The second section, on motivation, addresses the context in which the HERMETO initiative was conceived and the goals ascribed to the system. The third section presents HERMETO's architecture. HERMETO's functioning is briefly detailed in section four (on resources) and five

© J. Cardeñosa, A. Gelbukh, E. Tovar (Eds.)
Universal Network Language: Advances in Theory and Applications.
Research on Computing Science 12, 2005, pp. 254–260.

(on processes). Partial results, though rather preliminary, are reported in section six. Limitations and further work are addressed in section seven.

2 Motivation and Goals

HERMETO is a side product of two ongoing research and development projects carried out by NILC: POLICARPO and PUL∅. The former concerns the development of an English-to-Portuguese web translator, specialized in translating headlines and leads from the electronic edition of *The New York Times on the Web* into Brazilian Portuguese. PUL∅ concerns the development of a bimodal human-aided machine translation system for translating a Brazilian comics into LIST, a linearized version of Libras, the Brazilian Sign Language (for deaf people). Both systems are conceived as exclusively language-based, in the sense they are not supposed to require any extra-linguistic knowledge (as the one required in KBMT systems [4]) neither a corpus of already translated samples (as in the case for EBMT systems [5]). Additionally, both POLICARPO and PUL∅ were originally conceived as interlingua-based multilingual MT systems. Although the transfer approach might seem more suitable for each isolated task, our final goal is to provide a single system able to process, bidirectionally, both the oral-auditive (English and Portuguese) and the sign-gestural (LIST) input and output.

UNL was chosen as the pivot language because of three main reasons: 1) it's an electronic language for representing the semantic structure of utterances rather than its syntactic form; 2) the repertoire of UNL attributes can be extended to comprise semantic visual markers (as '.@round', '.@square', etc) required by sign language processing; and 3) as a multilingual and multilateral project, UNL could be used to assign cross-cultural interpretability to Portuguese and LIST texts. Nevertheless, it should be stressed that the use of UNL as an interlingua does not imply that UNL can only be used in such a way. This was a project strategy rather than a UNL vocation or shortcoming.

In such a multilingual MT environment, HERMETO was conceived as an embedded NL analysis system, which should allow for developer's customization and language parameterization. In its current state, it takes any plain text and enconverts it into UNL by means of a bilingual NL-UNL dictionary and a syntactic-semantic context-free grammar, both defined and provided by the user. The system was developed in C++, but it is still bound to the Windows environment. HERMETO's architecture is presented in the next section.

3 Architecture

HERMETO's architecture is presented in Figure 1 below. The input text - a plain text (.txt) written in ASCII characters - is split into sentences, each of which is tokenized and tagged according to the dictionary entries. Next, each sentence is traversed by a top-down left-to-right recursive parser, which searches for the best candidate match-

ing as defined in the context-free grammar provided by the user. After parsing, the resulting syntactic structure is interpreted into UNL according to the projection rules written in the user's semantic grammar. The output is a UNL document, in its table form, i.e., as a list of binary relations embedded between UNL tags.

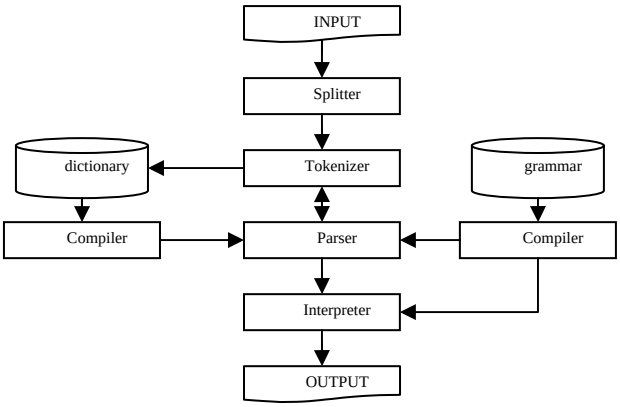

Fig. 1 HERMETO's architecture

4 Resources

HERMETO's lingware consists of a bilingual NL-UNL dictionary and a NL-UNL transfer grammar. No other language resource (as the UNL KB, for instance) is required for the time being. Both dictionary and grammars are plain text files, which are automatically compiled by the very machine. In order to improve grammar-writing tasks, HERMETO also comprises a grammar editor.

4.1 Dictionary

As EnCo, HERMETO takes a NL-UNL dictionary, whose entries, one per line, must be presented in the following format:

[NLE] {id} NLL "UW" (FEATURE LIST) <LG,F,P>;

NLE stands for "NL entry", which can be a word, a subword or a multiword expression, depending on the user's choice. NLL stands for "NL lemma". It is an optional field that can be used to clarify the string intended as NLE. The feature list consists of a list of attribute-value pairs, separated by comma. LG stands for a two-character language flag, according to the ISO 639. F and P indicate frequency and priority and are used for analysis and generation, respectively. Finally, any entry can be glossed and exemplified after the semi-colon.

The structure of HERMETO's dictionary is very much the same as EnCo's one: both dictionaries do not state any predefined structure, except for the syntax of each entry, and they can be customized by the user, who is supposed to decide the form of the entry, the need for lemmas and the set of attributes and the values they can take. However, there are three differences that should be stressed: 1) HERMETO compiles the plain text file itself, i.e., there is no need for a tool as DicBuild; 2) in HERMETO, the feature list is not a mere list of features but a list of attribute-value pairs, which allow for introducing variables in the grammar rules; and 3) HERMETO not only indexes but also compresses the dictionary (at the average rate of 65%).

Examples of dictionary entries are presented below:

[mesa] {} mesa “table(icl>furniture)” (pos:nou, gen:fem) <PT,1,1>;
[table] {} table “table(icl>furniture)” (pos:nou) <EN,1,1>;
[mesa] {} mesa “table(icl>furniture)” (pos:nou, ref:phy, fmt:squ) <LI[1],1,1>;

Except for the structure of the feature list and the language flag, HERMETO’s dictionary formalism is the same as the one proposed in the EnCo’s environment.

4.2 Grammar

HERMETO's grammar is a phrase-structure grammar defined by the 6-uple <N,T,P,I,W,S>, where N stands for the set of non-terminal symbols; T is the set of terminal symbols; P is the set of production rules; I is the set of interpretation rules; W is the weight (priority) of rules; and S stands for the start symbol. It is a context-free grammar, written in a plain text file, to be automatically compiled by the machine. The set of terminal symbols to be used as variables should be defined in the top of the grammar file, and the mapping between this set and the dictionary attribute values should be stated at the end of the document.

The rules should follow the formalism: p –> i, where $p \in P$, and $i \in I$. P, which is the syntactic component, can be expanded as a[w] := b, where $a \in N$, $b \in N \cup T$, and $w \in W$. I, the semantic component, is expanded as a list of attributes and relations in the following format: $\mathbf{att_1, att_2, ..., att_n, rel_1, rel_2, ..., rel_n}$ where att stands for attributive rules, and rel stands for relational rules, both comprised in the UNL Specification.

Attributive and relational rules hold between positions (in the rule string) or indexes rather than words. The grammar also takes a given set of primitive operators (such as '[ ]', for optional; ‘{ }’, for exclusive; '< >' for lemma; '+' for blank space; '#' for word delimiter, etc.) in order to extend the expressive power of the formalism and reduce the necessary number of rules. The ‘@entry’ marker should be stated in every level, and the entry word is to be considered the head of each phrase. As in X-bar theory [6], entry word features are projected to and can be referred by the immediate higher level.

Examples of HERMETO's rules are presented below:

[1] Due to the lack of an ISO 639 code for it, we have been using LI for LIST.

```
; 2.1.2. COMPLEX NOUN PHRASE (CNOP)
CNOP[2] := SNOP + 'and' + SNOP.@entry -> and(:03, :01)
CNOP[3] := SNOP + 'or' + SNOP.@entry -> or(:03, :01)
; 3.3. VERB
VERW[1] := ver.@entry - 'ied' -> :01.@past
VERW[1] := ver.@entry - 'ed' -> :01.@past
VERW[1] := ver.@entry - 'd' -> :01.@past
```

In such a grammar, context-sensitiveness can be stated as internal (dis)agreement between attribute values, such as in:

```
SNOP[1] := DET(GEN:x, NBR:y) + NOU(GEN:x, NBR:y).@entry -> :02.@def
```

The grammar is automatically compiled by HERMETO, which brings it to be an object-oriented scheme, where each non-terminal symbol is defined as an object, to be evoked by the others, during the syntactic and semantic processing. In order to optimize the compilation process, the length of each rule is limited to six symbols, and no nesting is admitted.

Although the expressive power of HERMETO's formalism may be the same as the one stated by EnCo, we claim that it is more intuitive, in the sense grammar writers are no longer supposed to be worried about the position of left and right analysis windows. They can work with (and even import) rules written according to more classic, high-level formalisms in NL understanding tradition.

5 Processes

HERMETO's resources are parameters for more general, language-independent processes, as splitting, tokenizing, tagging, parsing and semantic processing. These constitute the NL analysis and UNL generation modules. In this sense, HERMETO can be seen as a unidirectional transfer-based MT system itself, where NL is the source and the UNL is the target language.

5.1 Splitting, Tokenizing and Tagging

The process of sentence splitting, in HERMETO, is customized by the user, who is supposed to define, in the grammar, the intended set of sentence boundaries, such as punctuation marks and formatting markers, for instance. Each string of alphabetic characters or digits is considered a token, and blank spaces, as well as punctuation marks and non-alphabetic characters, are understood as word boundaries. Tagging is carried out through the dictionary, and no disambiguation decision is taken at this level. The word retrieval strategy seeks for the longest entries first, in the same way EnCo does. The word choice can be withdrawn, if HERMETO's parser comes to a dead-end situation.

5.2 Parsing

The tagged string of words is traversed by a chart parser, which applies the left (p) part of the grammar rules according to the priority defined by the user. Backtracking is supported, but cannot be induced. The parsing is rather deterministic, in the sense it provides only one parse tree for each sentence, the one best suited to the rules weight. Part-of-speech disambiguation is carried out during parsing, as the parser gets to the first possible parse tree. Parsing results can be exhibited by the interface and serve as the basis for semantic processing.

5.3 Semantic processing

Semantic processing is carried out together with parsing, in an interleaved way. Although semantic interpretation depends on the result of syntactic analysis, semantic projection rules are applied for any available partial tree, i.e., during the parsing itself. This does not cause, however, any parallelism between the syntactic and semantic modules, as the latter, although triggered by the former, cannot affect it. In this sense, HERMETO cannot deal with any generative semantics approach and is bound to the centrality of the syntactic component. Yet this can bring many difficulties in the UNL generation process, especially concerning the UW choice, i.e., word sense disambiguation, we have not advanced this issue more than EnCo does. The KB solution, which seems to be the most feasible one in EnCo environment, has not been adopted yet, for the trade-off still seems not to be positive, at least so far. As we have been mainly involved with an English sublanguage (the canned structure of English newspaper headlines and leads) and a regularized Portuguese (extracted from the comics), disambiguation can still be solved at the syntactic level.

6 Partial Results

For the POLICARPO and the PUL∅ projects we have been working on the English-UNL and the Portuguese-UNL enconverting respectively. In the former case, we have compiled almost 1,500 web pages, downloaded in September 2002 from the *The NY Times* web site, to constitute our training and assessment *corpora*. Both English-UNL and UNL-Portuguese dictionaries have been already provided for every English word, except proper nouns, appearing in the corpus. The grammar has been split into a core grammar, common to every sentence, and five satellite grammars, specialized in 1) menu items, 2) headlines, 3) leads, 4) advertisements and 5) others. Actually, we have observed that each of these sentence types convey quite different syntactic structures, which can be automatically filtered out of the general corpus. So far, we have already finished the core grammar and the one coping with menu items, and the precision and recall rates, for the assessment corpus, were 77% and 95% respectively, for complete UNL enconverting (i.e., UWs, relations and attributes). Although menu items generally consists on quite simple single word labels, it should be stressed that many of them involved complex morphological structures that had to be addressed by

the menu grammar. Anyway, HERMETO, together with the English-UNL dictionary and the core and menu grammars, has proved to be an interesting alternative for fully automatic English-UNL enconverting, at least in this case. For the time being, headlines have been already addressed, but no assessment has been carried out yet.

In PUL∅ project the coverage is rather small. Actually, the project is in its very beginning, and partial results concern a single story, for which HERMETO proved again, not only to be feasible for Portuguese-UNL enconverting, but to be easily integrated in a more complex system as well.

7 Shortcomings and Further Work

At the moment, we have been facing two main shortcomings: HERMETO accepts only ASCII codes and works only in Windows platform. Although we have planned to extend the current version to deal with Unicode and to run under other operational systems, we did not have the time to implement these changes. Furthermore, as we have been working rather on an English sublanguage (the NYT's one) and a sort of controlled (normalized) Portuguese, we have not really faced unrestricted NL analysis problems, which certainly will drive us to reconsider the UNL KB commitments. Therefore, in spite of the results achieved so far, HERMETO has still a long run before it can be considered a really feasible and suitable general NL-UNL enconverting environment. However, as former users of EnCo, we do believe it really represents a user-friendlier environment for fully automatic generation of UNL expressions out of NL sentences.

References

1. Uchida, H., Zhu, M., Della Senta, T. *A gift for a millennium. IAS/UNU, Tokyo, 1999.*
2. UNL Centre. *Enconverter specifications*. Version 3.3. UNL Centre/UNDL Foundation, 2002.
3. Uchida, H,. Zhu, M. *UNL annotation*. Version 1.0. UNL Centre/UNDL Foundation, 2003.
4. Nirenburg, S, Raskin, V et al. 'On knowledge-based machine translation', *Proceedings of the 11th International Conference on Computational Linguistics*, Bonn, 1986.
5. Furuse, O, Iida, H. 'Cooperation between transfer and analysis in example-based framework', *Proceedings of the 14th International Conference on Computational Linguistics*, Nantes, 1992.
6. Chomsky, N. 'Remarks on Nominalization', in Jacobs, Roderick A. & Rosenbaum, P. (eds.), *Readings in English Transformational Grammar*, Waltham (Mass.): Ginn, 1970, p. 184-221.