ARIES: a ready for use platform for engineering Spanish-processing tools*

José C. González, José M. Goñi
Amalio F. Nieto
E.T.S.I. Telecomunicación
Universidad Politécnica de Madrid
E-28040 Madrid (Spain)
Tel.: +34 1 5495700 ext. 440 Fax: +34 1 5432077
E-mail: {jgonzalez, jgoni, anieto}@gsi.dit.upm.es

Abstract

A lexical platform has been developed for the Spanish language both portable and efficient in terms of lexical coverage. Upon this platform a set of tools have been implemented, including first versions of a spell checker, unification based parser and grammar, and a stochastic morpho-syntactic tagger.

1 Introduction

The real situation of the Spanish language in relation with language technology is far behind most of its European counterparts. This situation is specially worrying if we take into account the spreading of Spanish all around the world and, therefore, the potential market for Spanish processing tools.

Our group has been working during the last four years to develop (almost from the scratch) a lexical platform both, portable and efficient (in terms of linguistic coverage of the morpho-syntactic phenomena, storage and access time). Upon this platform, a set of tools has been developed, as it will be shown below.

The main significance of these tools comes from the fact that they have been developed with a strong practical orientation and, simultaneously, starting from formalized linguistic knowledge, with support from the Laboratory of Computational Linguistics, Universidad Autónoma de Madrid (LLC-UAM), Spain.

2 The lexical platform

The architecture of our lexical platform is shown in figure 1. We distinguish in it two levels of representation:

Source level: this is the human-readable level. It is designed to permit the encoding of linguistic generalizations through default inheritance, rules to compute morpheme allomorphs, and the grouping of related entries in lemmas.

Object level: this is the computer-readable level. It is designed for the efficient retrieval of the entries by the applications. Different object dictionaries suited for different applications can be automatically compiled from the source level.

*This work has been supported in part by the Spanish *Plan Nacional de I+D*, through the project TIC91-0217C02-01, *An Architecture for Natural Language Interfaces with User Modelling.*

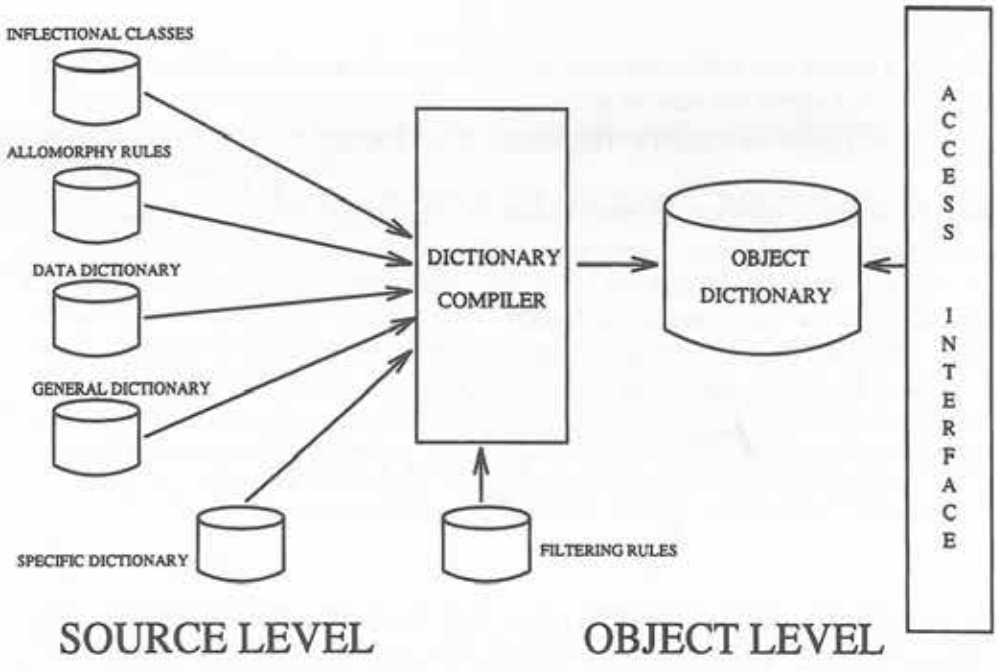

Figure 1: Lexical architecture.

2.1 Lexical source level

We have designed a specific formalism for source level lexical representation. Although it is described elsewhere [Goñi and González, 1995], we summarize here its main features:

Expressiveness: All the information needed in the lexical database can be expressed in a structured way. Linguistic generalizations are captured by grouping related entries in *lemmas*, or by using the mechanism of information inheritance. The information related to a lemma is structured in a tree-shaped feature bundle attached to it.

Versatility: Different implemented applications have different lexical interfaces, that are designed in a programming language dependent way. In our approach translation to other representation formats and languages is easily done in a non-ambiguous way.

Economy of expression: The syntactic overhead needed to structure the information has been reduced to a minimum, without endangering neither the expressive ability nor the non-ambiguity of the syntax of the formalism.

Non redundancy: Redundant information is kept to a minimum by exploiting default inheritance devices and notational abbreviations, such as value disjunction.

The design of this lexical representation language was influenced by the strong reliance of the Spanish language on inflectional morphology (e.g. 53 simple word forms for verbs, up to 4 forms for nouns and adjectives). We adopted a computational model for the treatment of Spanish inflectional morphology that is described in [Moreno, 1992] and in [Moreno and Goñi, 1995]. The main features of this model are:

1. Morphological processing is constrained to morpheme concatenation, so its allomorphic variants have to be stored or computed.

2. The model follows a *Graphical Word criterion*, that only considers relevant its written form. This criterion requires that additional allomorphs be necessary in some cases, because of diacritical marks, or different surface realization of the same phoneme in different contexts.

3. Feature unification is the information combining device used to select the relevant allomorphic variants to be concatenated. Allomorphs have an attached feature structure (DAG), and two or more of them are concatenated only if their feature structures are validated by context-free word formation rules[1].

4. Models for verbs and nominals are described in order to capture some interesting and well founded linguistic generalizations. These models capture regularities in the inflectional behaviour of the Spanish verbs and nouns.

5. Some lexicalized forms are included for highly irregular word forms not belonging to any of the proposed models.

Our lexical representation language includes devices to implement this morphological model:

- Inflectional models are encoded into classes, so information about the behaviour of the word forms can be inherited by lemmas.

- The inheritance mechanism implemented is multiple default inheritance, with a priority scheme to avoid conflicts (depth first, left to right searching). The default scheme is used to override information in exceptions.

- The different allomorphs needed by each lemma are not directly stored, but computed by means of regular expression based rules that can be invoqued by the classes or directly by the lemma entries.

- Sections of the lexical source base are designed for different types of entries, like lexicalized words (strong irregularities, invariable entries, etc.), morphemes and lemmas.

- A data dictionary section is included for the declaration of the possible feature names and values that can be attached to entries. This allows some limited type checking and efficient encoding of the compiled dictionary.

Our source lexical base has now a considerable size, accounting for more than 400,000 inflected forms. It currently includes 76 classes (both inflectional and auxiliary ones), 53 allomorphy rules (with a total of 176 productions), 622 morphemes, 697 lexicalized invariable entries, 21,000 nouns, 10,000 adjectives and 7,500 verbs.

2.2 Dictionary compiler

Object dictionary generation is achieved by specialized tools that are guided by filtering rules. These are a set of rules of a tree-manipulating language that allows the filtering, addition or modification of the branches of the feature bundle attached to each entry, as well as the computing of the different allomorphs for each lemma and their assignment of the relevant feature bundle. The tools (all of them implemented in C/C++) deal with:

1. Interpretation of filtering rules.

2. Inheritance searching algorithms, to assign the relevant feature bundle to each entry in the compiled dictionary.

3. Interpretation of regular expression rules to compute the allomorphs. The *GNU regexp* package has been used for such purpose.

4. Building the *letter-tree* index (*trie*) for efficient retrieval. This indexing system will be described in the next section.

[1]Rules currently used are like PATR-II ones, as described in [Shieber, 1986].

2.3 Dictionary interface

A flexible software access interface has been implemented to allow the applications a fast retrieval of lexical entries. The implementation language adopted is C/C++ also. A *trie* index is built while the object dictionary is compiled. The storage inefficiency of the *vanilla* trie index is solved with some simple modifications of the algorithm of the double-array trie described in [Aoe and Morimoto, 1992]. This indexing mechanism has allowed us the integration in this interface of the word segmentation needed for morphological analysis. Thus, word segmentation is no longer a blind process, since it is guided by the entries included in the lexicon.

3 Parsing tools

Applications can be built in a modular way, picking up the modules that are best suited for them. Figure 2 shows the system architecture, in particular inter-module dependencies in the implemented tools. In the following we describe all of them.

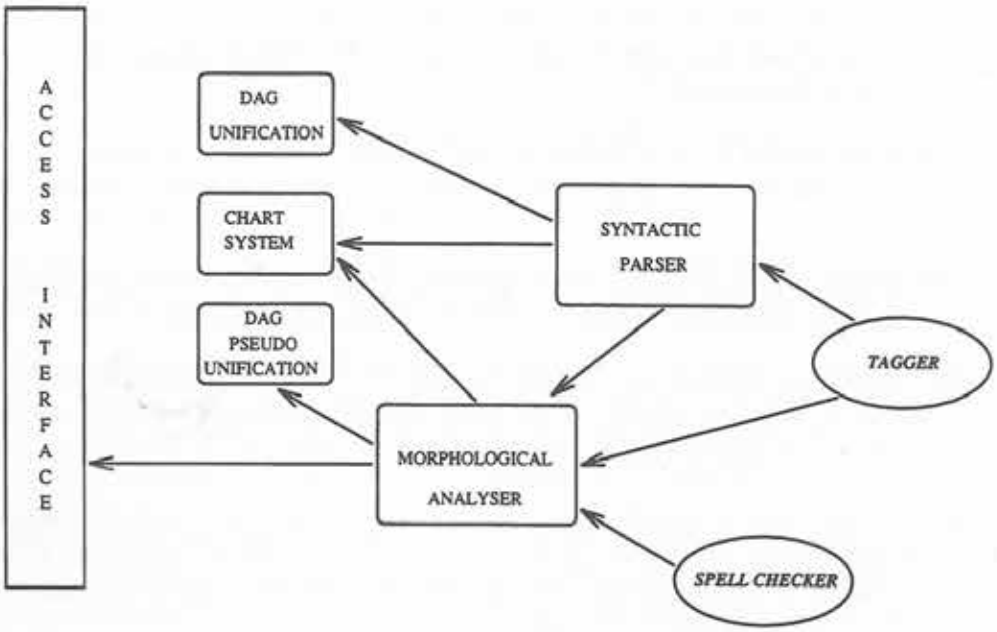

Figure 2: Parsing tools architecture.

3.1 Unification modules

Two modules were built for the processing of feature structures contained in PATR-II rules:

- A full unification module that allows real value sharing among features. The algorithm implemented is an enhancement of the *quasi-destructive graph unification algorithm* described in [Tomabechi, 1991].
- A pseudo-unification module that treats feature bundles as trees, without value sharing. Our implementation is a modification of the pseudo-unification algorithm described in [Tomita, 1988]. A lot of enhancements have been added to maintain consistency in the unified feature structures. These enhancements, that include sorting of the equations of the rule and two evaluations, guarantee that all the values are exactly the same in the (pseudo) unified paths.

3.2 The modular chart parsing system

Our modular chart parsing system, called NUCLEO, is a module that serve as a basis for the development of chart parsers. It is based on the CMCHART architecture proposed in [Thompson, 1983], and includes facilities for implementing parsers with different searching strategies, rule invocation strategies, and top-down, bottom-up, or mixed strategies.

3.3 Tools for morphological analysis/generation

A morphological analyser has been implemented using NUCLEO, the pseudo unification module and the lexical access interface (see figure 2). It is a modified chart parser that interprets word formation rules in PATR-II format. It can build the feature structures that can be associated to a given word.

We also developed a prototype of morphological analyser/generator in PROLOG programming language, called GRAMPAL. It is, basically, a DCG grammar for word formation. The advantages of this prototype are declarativity and bidirectionality. The lexicon needed for its operation is automatically derived from our source lexical base. It is fully described in [Moreno and Goñi, 1995].

3.4 The parser

The parser is built upon the NUCLEO system, and is integrated with the full unification module. The grammars have to be written in a PATR-II like format to feed the parser. A parser with the pseudo-unification module is also available if so desired. Although it is not implemented yet, a robust parser can be built in this parsing system, since charts can be used to recover partial results in incorrect analysis.

4 Application tools

4.1 The tokenizer

A tokenizer for the Spanish language has been developed using *flex*, the GNU generator of lexical analyzers. This tokenizer is just a prototype, used for evaluating the possibilities of a tool of this kind. It is planned to implement a more efficient tokenizer in C in a near future.

The main goal of the tokenizer is to split a text into tokens, which most of the time will be simple words, although multi-word tokens are allowed. The tokenizer returns a token per line (which makes easier the post processing of the output) and each of these tokens may be marked with a tag in order to differentiate among types of tokens: proper nouns, text between quotes or parenthesis, beginning and end of sentences, punctuation marks, paragraph delimiters or strange words (those which contain both numeric and alphabetic characters, or non Spanish characters); when the tokenizer returns a *normal* word, it is not tagged.

4.2 The stochastic tagger

In the context of the CRATER project[2], the Xerox Tagger [Cutting et. al., 1992] has been adapted to Spanish. The Xerox Tagger uses a statistical method for text tagging. In this kind of systems, ambiguity of assignment of a tag to a word is solved on the basis of the most likely

[2]This project (*C*orpus *R*esources *A*nd *T*erminology *Ext*R*action*) is funded by the Commission of the European Communities under contract MLAP-93/20. Our group contributes to this project as subcontractor of *Laboratorio de Lingüística Informàtica, Universidad Autónoma de Madrid*, Spain, being Fernando Sánchez the project leader of the Spanish team.

interpretation. A form of Markov model is used that assumes that a word depends probabilistically on just its part-of-speech, which in turn depends on the category of the preceding two words.

One of the main advantages of the Xerox Tagger is that it does not need any tagged corpus; it is only necessary a lexicon with the more common words and their corresponding tags, and a suffix lexicon which stores the set of tags which may correspond to a particular suffix[3]. With these elements it is possible to train the Hidden Markov Model (HMM) in which the tagger is based.

Once the tagger has been trained, it can be fed with a string or a text to be tagged. First of all, the input is converted into a sequence of tokens which are passed to the lexicon; if the token is in the lexicon, it is annotated with the tag(s) provided by the lexicon. If it is not, but the token can be split into a verb form plus some clitics pronouns, this decomposition is done, and each of the elements resulting from the splitting is tagged as if it would be a token originally obtained from the input text. If neither the token is in the lexicon nor can be split, the suffix table is looked up for a matching; if there is any, the set of tags corresponding to the matched entry is returned. Finally, if all of this fails, a default set of tags is returned.

The output of the tagger could be used as input for a syntactic parser, or could be used for compiling statistics about order and relations between different categories of words.

The modifications made to the Xerox Tagger in order to adapt it to Spanish have to do mainly with suffix handling and processing of clitics pronouns.

4.3 The verbal subcategorization tool

Verbal subcategorization frames are the different syntactic structures that a particular verb can admit: types of complements (and required prepositions), non-finite forms, reflexive forms, auxiliaries, etc. A tool (called SOAMAS [Monedero et al., 1995])has been developed to automatically obtain the frames corresponding to Spanish verbs from tagged texts. The system includes three grammars. The first one, for the analysis of the verb phrase, identifies main and auxiliary verbs (as well as the possible intermediate prepositions or conjunctions and clitic pronouns). The second one recognizes nominal, adjective and prepositional phrases inside the verb phrase. The last grammar describes the structure of the verbal complements in Spanish. Grammars have been implemented from the work in [Hallebeek, 1992].

25 different subcategorization frames are considered, including simple, composed and auxiliary frames. The system has not been integrated yet with the stochastic tagger and the current version of the morpho-syntactic analyzer. The system has been tested by using a 10,000 words text manually tagged by Martín [Martín, 1994], but no statistical analysis has been made yet.

5 Demonstrators

An e-mail interface has been built to show the capabilities of the tools developed until now. The address to which petitions can be sent is `aries@mat.upm.es`. The subject must be:

- `ayuda`: to get instructions (in Spanish) on the use of the demonstrator.
- `comprueba`: takes at most 10 lines of the message body or the first 100 words and returns to the petitioner the lists of unknown and unanalyzed words.
- `analiza`: to get the morpho-syntactic analysis of one word (the first word in the message body. For instance, the answer obtained for the word `volviais` is:

[3]This suffix lexicon has been generated manually by Fernando Sánchez; the original tagger has the possibility of generating suffixes automatically from a corpus, but this method produces poor results in the case of Spanish.

```
lex = volver
cat = v
agr pers = 2
agr num = plu
vinfo tense = impf
vinfo mood = ind
```

what means that the lemma for this word is the verb `volver`, second person, plural of the imperfect tense, mood indicative.

6 Evaluation

A first evaluation of the ARIES platform has been carried out regarding its ability for spell checking purposes. Lexical coverage has been the only aspect under consideration. What we are using as a spell checker is in fact a morpho-syntactic analyzer that generates, for recognized words, all their possible analysis. No alternative spelling is provided for unrecognized words.

The current version of the platform is relatively efficient in terms of space needed for the storage of object dictionaries. No effort has been allocated however until now to improve time efficiency (around 5 words/second in a HP-715/50 workstation) . Anyway, an improvement of at least one order of magnitude is considered as a feasible goal to achieve in the very short term.

Evaluation has been carried out by using a collection of texts from the Culture Supplement of the Spanish newspaper *ABC*. The text was previously tokenized to filter out proper nouns, acronyms, quoted and bracketed expressions, etc. Of course, the filtered output was still containing foreign words and typographic errors. A specialized dictionary of culture terms including around 400 words (less than 1200 inflected forms) was created to supplement the core dictionaries of the three spell checkers used for evaluation: ARIES on one side and the Spanish versions of the WP 5.1 and ISPELL 3.1.00 spell checkers on the other[4]. The size of the corpus used, after tokenization, is 714,124 words long. The number of distinct forms is 42,364.

The results of the evaluation are shown in table 1.

Spell checker	Unrecognized distinct words	% error (42,364 distinct words)	Unrecognized words	% error (over 714,124 words)
ARIES	3,167	7.5%	4,264	0.6%
ISPELL	5,630	13.3%	9,919	1.4%
WP 5.1	9,245	21,8%	18,622	2.6%

Table 1: Results of the evaluation

7 Acknowledgments

Special mention deserve the students (around 40) that have participated in the development of lexical resources and tools in our group for the last five years, although their names can not be included here for obvious reasons.

[4]WP is a trademark of WordPerfect Corporation. The Spanish ISPELL dictionaries used for this evaluation (version 1.4, December 1994) have been developed by J. Carretero and S. Rodríguez at Facultad de Informática, Universidad Politécnica de Madrid, Spain.

References

[Aoe and Morimoto, 1992] Aoe, J. and Morimoto, K. An efficient implementation of trie structures. *Software-Practice and Experience, vol. 22, n. 9, pp. 695–721, September, 1992.*

[Cutting et. al., 1992] Cutting, D.; Kupiec, J.; Pedersen, J. and Sibun, P. A Practical Part-of-Speech Tagger. *Xerox PARC, 1992.*

[Goñi and González, 1995] Goñi, J.M. and González, J.C. A framework for lexical representation. *In Proceedings of AI'95: Fifteenth International Conference. Language Engineering '95, pp. 243–252. Montpellier, June 27–30, 1995.*

[Hallebeek, 1992] Hallebeek, J. A Formal Approach to Spanish Grammar. *Language and Computers: Studies in Practical Linguistics, Volume 7. Rodopi, 1992.*

[Martín, 1994] Martín de Santa Olalla, A. Una Propuesta de Codificación Morfosintáctica para Corpus de Referencia en Lengua Española. *Tesis doctoral, Universidad Autónoma de Madrid, 1994.*

[Monedero et al., 1995] Monedero, J.; González, J.C.; Goñi, J.M.; Iglesias, C.A. and Nieto, A. Obtención automática de marcos de Subcategorización verbal a partir de texto etiquetado: el sistema SOAMAS. *Accepted for presentation at the XI Conferencia de la Sociedad Española para el Procesamiento de Lenguaje Natural (SEPLN-95), Bilbao. Septiembre, 1995.*

[Moreno, 1992] Moreno, A. Un modelo computacional basado en la unificación para el análisis y la generación de la morfología del español. *Tesis Doctoral. Universidad Autónoma de Madrid, 1992.*

[Moreno and Goñi, 1995] Moreno, A. and Goñi, J.M. GRAMPAL: A morphological model and processor for Spanish implemented in Prolog. *Accepted for presentation at the 1995 Joint Conference on Declarative Programming (GULP-PRODE'95), Marina di Vietri (Salerno, Italy). September, 1995.*

[Sánchez and Nieto, 1995] Sánchez, F. and Nieto, A.F. Development of a Spanish Version of the Xerox Tagger. *Universidad Autónoma de Madrid, May, 1995.* (Recoverable as http://xxx.lanl.gov:80/list/cmp-lg/9505035).

[Shieber, 1986] Shieber, S.M. An Introduction to Unification-Based Approaches to Grammar. *CSLI Lecture Notes. Center for the Study of Language and Information. Stanford University, 1986.*

[Thompson, 1983] Thompson, H.S. MCHART: A Flexible Modular Chart Parsing System. *In Proceedings of the National Conference on Artificial Intelligence (AAAI"83), pp. 408–410. August, 1983.*

[Tomabechi, 1991] Tomabechi, H. Quasi-Destructive Graph Unification. *In Proceedings of the 29th Annual Meeting of the ACL, pp. 315–322. June, 1991.*

[Tomita, 1988] Tomita, M. The Generalized LR Parser/Compiler. User"s Guide. *Center for Machine Translation. Carnegie Mellon University, 1988.*