INSTITUT UNIVERSITARI DE LINGÜÍSTICA APLICADA
UNIVERSITAT POMPEU FABRA

Ph.D. program: Applied Linguistics
Academic period 1996 - 1998

Ph.D. Dissertation

Combining Machine Learning and Rule-Based Approaches in Spanish Syntactic Generation

Maria Teresa Melero Nogués

Ph.D. Dissertation
To obtain the Ph.D. degree in the Universitat Pompeu Fabra

Directed by: Antoni Badia i Cardús

UNIVERSITAT POMPEU FABRA

Barcelona, 2006

Dipòsit legal: B.34197-2006
ISBN: 84-690-0317-8

Per en Pau, la Mònica i la Mireia i, sobretot, per en Gabriel

Acknowledgements

I am indebted to the whole NLP team at MSR with whom I shared the challenge of successfully getting MSR-MT off the ground between the years 1999-2003. I am particularly grateful to my fellow Generation grammarians: Lee Schwartz, Takako Aikawa and Andy Wu who shared with me interesting discussions on robustness and application-independence of the Generator, and to Michael Gamon whose help with Decision Trees was invaluable.

Table of Contents

Chapter 1
Introduction

1.1 Goals of this Thesis

As the title of this thesis indicates, what we are presenting here is a Generator for Spanish, which combines hand-written rules and Machine Learning techniques. A Generator does not normally have an independent existence; it usually belongs to a specific application. In our case, it was originally designed for –and is still used by- a full-scale commercial quality Machine Translation system developed at Microsoft Research[1] (MSR-MT).

The output of any Generator is –usually- grammatically acceptable text in a given human language, e.g. Spanish. That is why a Generator is better defined, or distinguished from other Generators, by its input. The input to our Generator is a predicate-argument representation, with deep syntactic information, at the sentence level, such as the Logical Form provided by MSR-MT [Heidorn, 2002]. This Generator therefore falls within the category of (*deep) Surface Realizers*.

The contents of this thesis can be distinctively divided in two parts:

1. The first part contains a complete description of the Generator's grammar hand-written rules and Generation specific linguistic strategies. Here we also explain how the design of the grammar takes into account the sequence of the application of the rules, as well as the availability of information at every point in the process. Although we claim that the Generator has been designed to be application-independent, in order to better understand its role and functionalities, as well as its sometimes problematic input, the description of the Generator proper is preceded

[1] All the work described in this thesis was carried out by the author while she was collaborating in the development of MSR-MT at Microsoft (Redmond, USA), between 2001-2003.

by an overview of the MT system of which it is the last step. The need for robustness in real-world situations in the everyday use of the MT system requires from the Generator an extra effort which is resolved by adding a Pre-Generation layer. The Pre-Generation module is then able to *fix* the input to Generation, without *contaminating* the grammar rules. We argue the benefits of this modular architecture for the independence and maintainability of the Generator, compared with incorporating ad-hoc operations in the core grammar. In order to put the system described into perspective, we include an evaluation of MSR-MT, in which it is compared against one of the best English-Spanish translators in the market, as well as an autonomous evaluation of the Generator isolated from the rest of the system's components.

The main ideas present in this part of the thesis have appear in the following publications by the author:

Melero, M. and Font-Llitjos, A. (2001). Construction of a Spanish Generation module in the framework of a General-Purpose, Multilingual Natural Language Processing System. In Proceedings of the VII International Symposium on Social Communication, Santiago de Cuba, pages 283-287.

Aikawa, T., M. Melero, L. Schwartz, and A. Wu. (2001). Multilingual Sentence Generation. In Proceedings of 8th European Workshop on Natural Language Generation (ACL-2001), Toulouse (France).

Aikawa, T., M. Melero, L. Schwartz and A. Wu (2001). Generation for Multilingual MT. In Proceedings of the VIII MT-Summit, Santiago de Compostela (Spain).

2. In the second part we explore the use of Decision Tree classifiers (DT) for automatically learning one of the operations that take place in the Pre-Generation component, namely lexical selection of the Spanish copula (i.e. *ser* and *estar*). In our experiment, we use machine-learning techniques to leverage large amounts of data for discovering the relevant conditioning features for the selection of the copula. As a machine learning technique for the problem at hand, we choose

decision tree learning, a practical approach to inductive inference in widespread use. In this part of the thesis, we evaluate the usefulness of selecting the copula in Generation rather than doing it in Transfer and we show that it is possible to infer from examples, by means of DTs, the contexts for this non-trivial linguistic phenomenon with high accuracy. In our experiments, we evaluate the impact of the linguistic domain of the training data on the quality of the statistical model and discuss the differences between the results obtained using corpora from different domains.

The results of the experiment described in this part have been published in:

Melero, M., T. Aikawa and L. Schwartz (2002). Combining machine learning and rule-based approaches in Spanish and Japanese sentence realization. Second International Natural Language Generation Conference, New York (USA).

1.2 Natural Language Generation and Surface Realization

Natural Language Generation is a subfield of Computational Linguistics that is concerned with producing understandable, grammatical texts in Spanish or any other human language.

It has been suggested that the relative lack of work in NLG, compared to Natural Language Understanding (NLU) or Analysis, is due to the fact that while all Analysis systems take language utterances (i.e. plain text) as input, it is much harder to define what the input to Generation is [Dale, 2000]. This question is known as *the problem of the source*. Without independently motivated input representation it can be hard to say very much that is not idiosyncratic.

Generation must be seen as a problem of construction and planning rather than analysis. Major problems in Natural Language Analysis are caused by ambiguity. Generation has the opposite information flow. Provided that the input is faithful, ambiguity in a Generator is not possible. Rather, a Generator's problem is to choose from an oversupply of possibilities, and what information to omit.

Generation entails realizing goals in the presence of constraints and dealing with the implications of limitations of resources (for example, expressive capacity of the syntactic and lexical devices of a given language). What is needed to produce a fluent text is either trivial (e.g. template method) or else is quite difficult because one has to work out a significant number of new techniques and facts about language that other areas of language research have never considered [Mc Donald, 2000].

The lack of a consistent answer to the *problem of the source* has been at the heart of the problem of how to make research on Generation intelligible and engaging to the rest of the Computational Linguistics community, and it has complicated efforts to evaluate alternative treatments. Differences in what information is assumed to be available has an influence on what architectures are plausible for Generation and what efficiencies they can achieve.

Advances in the field have come precisely from insights into the representation of the source. In [Cardeñosa et al, 2003], the authors propose the use of a Universal Networking Language (UNL), a sort of interlingual representation as a possible standard for the normalization of inputs to generation processes.

The Generation task generally includes two main components [Mc Donald, 2000]:

1. A text planner. Selects or receives the units from the application and organizes them to create a structure for the utterances as a text by employing some knowledge of rhetoric or discourse. Takes care of the information flow: new and old; in focus or not, etc.
2. A linguistic component (also known as Realizer). Realizes the planner's output as an utterance. Its task is to adapt (and possibly select) linguistic forms to fit their grammatical contexts and to orchestrate their composition. This process leads to a surface structure for the utterance that is then read out to produce the grammatically and morphologically appropriate wording for the utterance.

When the application that uses the Generator is a Machine Translation system, usually the first component can be dispensed of. The reason is that the text in the source language that is being translated already provides the planning required by Generation.

On the other hand, other applications such as data mining, text summarization, dialogue systems, etc. may require a text planner.

The linguistic component, or Surface Realizer, is generally considered the most mature and well-defined of all the processes in NLG. It consists in the application of a grammar (aka Generation Grammar) to produce a final text in a particular language from the elements that were decided on by the earlier processing (e.g. syntactic and semantic representations).

All grammars are incomplete when it comes to providing accounts of the actual range of texts that people produce. In this case, Generation may seem in better situation than Analysis because as a constructive discipline, we can choose whether to use a construct or not, leaving out everything that is problematic. Analysis systems, on the other hand, must attempt to read the texts with which they happen to be confronted, and so inevitably will be faced at almost every turn with constructs beyond the competence of its grammar. However, as we will see in the course of this thesis, the input to Generation in real-life applications is not always as clean and predictable as it should be, and therefore an inevitable parallel has to be drawn with the problems that Analysis faces with respect to ill-formed input.

A grammatically correct input sentence is commonly considered a legitimate input to a Parser; in the case of Generation, a well-formed, complete input should be correct in a similar sense [Buseman, 2002]. If we want our Generator to be application-independent and reusable, then a formal specification of the input to Generation is compulsory. On the other hand if we want it to perform –and perform well- on real-world situations, it needs to be able to confront ill-formed input. We address this conundrum in this thesis. Our solution goes in the line of fixing the input before it reaches the Generation grammar. A similar approach has been followed by other systems, such as Storybook, a narrative prose generator that uses FUF/Surge as Surface Realizer [Callaway, 2002]. In this case it is up to the sentence planner to ensure that only Functional Descriptions that will create grammatical sentences can be constructed. FDs are hybrid semantic/syntactic entities that can be used to produce text via unification. Similarly, [Corston-Oliver, 2003] also uses Decision Trees and transformation-based learning to correct transferred linguistic representations.

The Generator that we present in this thesis shares the approach to surface realization with other in-depth, linguistically motivated, realization components, such as Penman [Penman, 1989], KPML [Bateman, 1997], SURGE[2] [Elhadad and Robin, 1996b], RealPRO [Lavoie and Rambow, 1997] and [White and Caldwell, 1998]. All these Realizers are –or claim to be- domain-independent, based on sound linguistic principles and exhibit a broad coverage of English. All are symbolic, hand-written grammar-based systems, often based on syntactic linguistic theories such as Halliday's [Halliday, 1976] systemic functional theory (FUF/SURGE and KPML) or Mel'cuk's [Mel'cuk, 1988] Meaning-Text Theory (REALPRO). RealPro's deep syntactic structures (DSyntSs), which represent semantic roles and dependency structures, are most similar to MSR-MT Logical Forms.

Statistical and machine-learned approaches have been applied to sentence realization as well. The Nitrogen system, for example, uses a word bigram language model to score and rank a large set of alternative sentence realizations [Langkilde and Knight, 1998]. Other approaches use syntactic representations. FERGUS [Bangalore and Rambow, 2000], Halogen [Langkilde-Geary, 2002] and Amalgam [Corston-Oliver et al., 2002] use syntactic trees as an intermediate representation to determine the optimal string output. Amalgam, in particular, has more relevance to the experiment presented in the second part of this thesis about the automatic selection of the copula. Like Amalgam, our input is a logical form graph, i.e., a sentence-level dependency graph with fixed lexical choices for content words. This graph represents the predicate-argument structure of a sentence and includes semantic information concerning relations between nodes of the graph. Like for some of Amalgam operations, we use Decision Tree classifiers. There are differences however. While in our approach we separate input checking operations (i.e. Pre-Generation) from core Generation Grammar, and we advocate statistical modeling for the former and hand-written rules for the latter, in the Amalgam approach, there is no Pre-Generation layer. Machine-learned operations and hand-written rules co-exist in the same Generation layer. In Amalgam, hand-written rules are introduced when the classifier is unable to make the correct decision due to insufficient training data, or when the operations are straightforward. Other recent surface Generators are hybrid as well, for

[2] Systemic Reusable Grammar of English

example, SEGUE [Pan and Shaw, 2004], which employs case-based paradigm but performs rule-based adaptations. It uses an annotated corpus as its knowledge source and employs grammatical rules to construct new sentences.

As for Spanish Generation components or Sentence Realizers, the situation is somewhat leaner. We find a Spanish version of SURGE, implemented by the group developing Storybook [Callaway et al., 1999], although the coverage is admittedly inferior to its English counterpart. There are two Spanish grammars available for Generation with KPML: The Ontogeneration/GUME-Project (UPM, Madrid), which ran from 1996 to 1998, developed a grammar for use in verifying entries into a database concerned with chemistry [Aguado et al, 1998] and there is an undergoing development as part of a project investigating the use of Generation grammars in language learning, teaching, and reference grammars; this grammar is being written by Juan Rafael Zamorano Mansilla as part of his PhD project at the University of Bremen [Zamorano, 2003].

1.3 Statistics vs. Knowledge-Engineered: Hybrid is Beautiful

In the space of two decades, statistical methods have gone from being frowned upon to being "the right way" (almost the *only* way) to do Computational Linguistics. No doubt, corpus-based, statistical methods have revolutionized the field and for a while it seemed that hand-written grammars, based on linguistic theories and sound principles could be disposed of without much ado.

Is that so? Is the "ancient style" rule-based approach less valid than it used to be? After all, most of the real working systems (in the field of MT, for example) are rule-based and some of them, including one of the most used, Systran, have thousands of hours of linguist's patient work behind them.

There is certainly no going back to the time before corpus-based linguistic, but knowledge-engineered resources, although they never ceased to exist, are experiencing a come-back in the form of hybrid systems, like the one we describe in this thesis. The fact is that hybrid systems are becoming mainstream in many NLP applications, particularly MT [Somers, 1999; Carl and Schäler, 2002].

Fortunately, statistics and linguistic knowledge are good companions. Each one is good at different parts of the job and they complement each other well. Rule-based or symbolic systems benefit from linguistic generalizations, and in general can be considered to have a larger scope than corpus-based systems. However, they are bad at dealing with exceptions and sub-domain particularities. Corpus-based systems, on the other hand, exploit the fact that language observes statistical regularities. They have made progress possible on a number of issues that were real hurdles for symbolic approaches, such as disambiguation, error detection and ill-formed input.

Our Generator is first and foremost a knowledge-engineered Grammar that incorporates a machine-learned operation (copula selection), after a successful experiment with interesting results, reported in the second part of this thesis. Let us also note that the corpus used to train the Decision Tree classifiers to learn copula selection has been processed with a rule-based Analysis grammar.

It is worth mentioning that the MT system that uses the Generator is an excellent example of a hybrid system. MSR-MT is a data-driven MT system that combines rule-based Analysis and Generation components with statistical, deep Example-Based transfer.

Example Based Machine Translation (EBMT) emulates human translation practice in recognizing the similarity of a new source language sentence or phrase to a previously translated item and using this previous translation to perform what is known as "translation by analogy”. The underlying hypothesis is that translation often involves the finding or recalling of analogous examples, i.e. how a particular expression or some similar phrase has been translated before. Like Statistical MT, EBMT uses aligned, parallel corpora, from which a source string -> target string database is built. In an EBMT database, the more linguistically processed this string is, the better its generalization capability is. For example, a lemmatized database is more powerful than a literal database; and a database of sentences that have been syntactically or semantically analyzed, and where surface word order phenomena have been neutralized is even more able to generalize. The latter is true for MSR-MT.

EBMT systems, at learning time, extract and select equivalent phrases or word groups from a databank of parallel bilingual texts, which have been previously aligned, generally by statistical methods. At run-time, the database of parallel translations is searched for the source language sentences and phrases closest matching a new source language sentence. The translations of the matched phrases are then modified and combined to form a translation of the new sentence. Just as for Translation Memories, the analogy-based translation builds on approved translations; consequently the quality of the output is expected to be high.

MSR-MT, by combining symbolic techniques and statistical techniques, benefits from the advantages of EBMT systems, namely adaptation to new domains and new language pairs, as well as higher quality translations, together with the advantages of linguistic generalizations.

A major problem of EBMT, which is the re-combination of selected target language examples (generally short phrases in pure EBMT systems) in order to produce fluent and grammatical output, is solved by a full-fledged Generation grammar in a hybrid system such as MSR-MT.

1.4 Organization of this Thesis

The first part of the thesis spans from Chapter 2 to Chapter 5; the second part is entirely contained in Chapter 6.

- **Chapter 2** describes MSR-MT, the Machine Translation (MT) system that uses the Generation grammar described in this thesis;
- In **Chapter 3** we describe a Generation grammar for Spanish that takes as input a predicate-argument structure with deep syntactic information, called Logical Form, and then outputs a linearly ordered surface syntax tree with fully inflected leaves;
- In **Chapter 4**, we discuss the actual problems encountered when dealing with real-world applications and we will propose a modular architecture for the Generator that provides both application independence and robustness;
- **Chapter 5** presents different evaluation exercises that help put the Generator in context: evaluation and user satisfaction survey of the English-Spanish MT,

evaluation of monolingual Generation, and evaluation of a reusability experiment in which a French-Spanish system is quickly assembled;

- In **Chapter 6** we explore the use of decision tree classifiers (DT) for automatically learning the lexical selection of the Spanish copula in the context of our Generator. We also evaluate the impact of the linguistic domain of the training data on the quality of the statistical model and discuss the differences between the results obtained using corpora from different domains;
- In **Chapter 7** we summarize the main contributions of this thesis.

1.5 Relevant Links

- For an extensive introduction to NLG and an up-to-date state of the art, the reader is referred to John A. Bateman's web page. The bibliographical references to further work and reading are also extensive.
  http://www.fb10.uni-bremen.de/anglistik/langpro/webspace/jb/info-pages/nlg/ATG01/ATG01.html

- The home page of the Association for Computational Linguistics Special Interest Group on Text Generation contains many useful information and links to Events, Workshops, Conferences and Symposia. It also has a Newsletter and links to downloadable resources.
  http://www.siggen.org/

- A very complete list of implemented Natural Language Generation systems, maintained by John Bateman and Michael Zock can be found here:
  http://www.fb10.uni-bremen.de/anglistik/langpro/NLG-table/NLG-table-root.htm

- Machine Translation is one of the major focuses of the Natural Language Processing group at Microsoft Research. The web for this project can be accessed here:

http://research.microsoft.com/nlp/Projects/MTproj.aspx

- MSR-MT is currently being used to translate Microsoft Product Support Services (PSS) Knowledge Base[3] (KB) into Spanish. It is available online at http://support.microsoft.com/default.aspx?scid=fh;Es-eS;KBHOWTO. (You can enter Spanish queries for the KB and receive back machine-translated hits. The user is warned that the article has been machine-translated).

[3] Over 140,000 articles translated by MSR-MT, together with a few thousand human-translated ones.

Chapter 2

The MSR Machine Translation System

This chapter describes MSR-MT, the Machine Translation (MT) system that uses the Generator described in this thesis. MSR-MT is a large-scale example-based multilingual Machine Translation system developed at Microsoft Research. It is a hybrid system that uses both rule-based and statistical components. Analysis and Generation are performed using linguistic parsers and syntactic realization modules, the rules of both of which are hand-coded. Transfer, on the other hand, is accomplished using Transfer rules or mappings that are automatically extracted from aligned corpora. The main information sources used to write this chapter are published papers by members of the MSR-NLP group: for data on the Spanish components: [Lozano and Melero, 2001], [Jiménez, 2001] and the author's Master thesis [Melero, 2001]; for the Analysis process and parser: George Heidorn's contribution to the Handbook *of Natural Language Processing Techniques* [Heidorn, 2000]; for Logical Form: [Campbell and Suzuki, 2002a, 2002b]; and for the section on Alignment and Transfer: [Richardson et al, 2001a], [Menezes and Richardson, 2001] and [Pinkham and Corston-Oliver, 2001].

2.1 Overview of MSR-MT

MSR-MT is a data-driven hybrid MT system that combines rule-based Analysis and Generation components with statistical, deep example-based Transfer. Figure 1 shows graphically the flow of the translation process from source to target

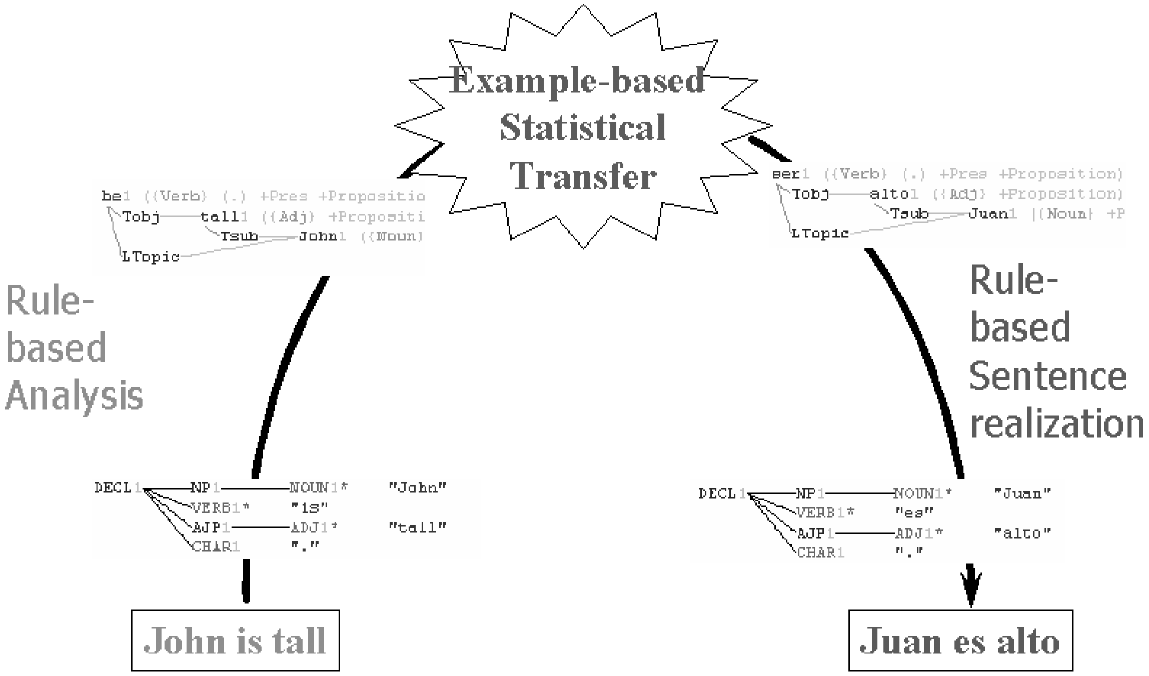


Figure 1: Translation process

The translation process begins with the analysis of a source language sentence by the source language parser. The output, an *annotated syntactic tree*, is the input to the *Logical Form* (LF) module. This module produces an *annotated predicate argument structure*, or LF representation, of the source sentence. These representations use the same basic set of relation types for all languages and thus are compatible across them.

Logical Forms are the final output of the Analysis phase, and the input to the Transfer phase.

Transfer extracts a set of mappings from a bilingual knowledge database known as *translation MindNet*, and applies these mappings to the LF of the source sentence to produce a target LF. Thus, the translation MindNet for a given language pair is a repository of aligned LFs and portions of LFs, produced by analyzing sentence-aligned corpora. An alignment of two LFs is a set of mappings between a node or set of nodes (and the relations between them) in the source LF and a node or set of nodes (and the relations between them) in the target LF.

In the translation process, the Transfer component searches the alignments in the MindNet for those that match portions of the LF of the sentence being translated. That is called *mindmelding*. Mappings with larger context are preferred to mappings with smaller context and higher frequency mappings are preferred to lower frequency mappings.

The lemmas in any portion of the LF of the input sentence that do not participate in a mapping extracted form the Mindnet, are posteriorly mapped to a target lemma using a bilingual dictionary of lemmas. The target LF fragments coming from the transfer mappings and the dictionary mappings are eventually *stitched* together to produce the final target LF or transferred LF.

The transferred LF is then input to the *Sentence Realization* component, aka Generation grammar. The whole translation process relies on the felicity of the alignments and on the result of the mindmelding, which are mostly done automatically, and may yield defective LFs. For this reason, Generation has to be capable of handling inconsistent or incomplete LFs, whenever possible, adding to the total robustness of the system.

Like other MT systems that acquire knowledge automatically from example sentences and their translations in bilingual corpora, MSR-MT is easily customizable to new domains. However, by learning from Logical Forms, rather than directly from sentences, its capacity for generalization and hence its translation quality is potentially much higher.

2.2 The Microsoft NLP System

MSR-MT is an integral part of the Microsoft NLP system. The Microsoft Natural Language Processing system, which has also been used in other applications, such as *grammar checking*, consists of a programming language, called G, and a runtime environment, both specifically developed for Natural Language Processing.

Although G is syntactically similar to C, it provides special support for attribute-value data structures, called *records*, and allows for the use of certain programming constructs, called *rules* (or more appropriately, linguistic rules).

The runtime system is a Microsoft Windows application written in C, often referred to as *NLPWin*. It provides a grammar development environment and a set of functions, for natural language processing. The grammar of a given language, such as the Generation grammar for Spanish, is written in G, then, at compilation time, is translated into C by a program called Gtran, and is then compiled and linked into the NLPWin executable. The NLPWin platform also provides powerful testing and verification tools.

The system is intended to do both Analysis and Generation of Natural Language text.

The Analysis process, which we describe in some detail in the next sections, encompases the following steps:

a. Lexical processing,

b. Syntactic analysis, divided in two stages:

   i. Sketch

   ii. Portrait

b. Logical Form

2.3 Lexical Processing

Lexical processing is the first stage of the Analysis process. The very first step consists of identifying the individual words or tokens (*tokenization*). Then the words are looked up in the dictionary or lexicon[4]. If a word form is not found in the dictionary, it is lemmatized using the morphological analyzer. Then the lemma -or lexical root- is checked again on the lexicon.

Multiword units are also recognized at this stage. There are two types of such units:

- *Multiword entries*, such as *sin embargo* or *en vez de*, which are stored in the lexicon.
- *Factoids* or *named entities*, which are analyzed using simple syntactic rules [Jiménez, 2001]; they comprise expressions such as: names of places (*río Llobregat*, *delta del Ebro*), dates (*11 de septiembre*), proper names (*Felipe González, Winston Churchill*), names of products (*Windows XP*), etc.

[4] The two terms are used indistinctively in this thesis.

The Spanish dictionary contains circa 140,000 entries. It was initially created by processing the *Novell* dictionary and then enriched using information automatically extracted from the *Diccionario VOX de la Lengua Española*[5]. Each lexical entry contains morphological and syntactic[6] information, plus a few basic semantic features, such as 'Human' and 'Count'. During the development of the Spanish Grammar Checker, other features were manually introduced, such as 'TakesSubj', for instance, which is a Boolean feature used to indicate whether a given verb subcategorizes for a subjunctive clause.

The output from the lexical processing is stored in the form of lists of records; each record is a list of attribute-value pairs that contain all the information available to that particular word. This information can be accessed at any point during processing and is used in later stages of the analysis.

An interesting feature of the system is the way it handles ambiguity. When a word has different senses in the dictionary that share the same part-of-speech, all the features, which are called *bits*, are merged into a single record. This procedure is called *smooshing*, and the resulting record is called *smooshed* record. Smooshed records help avoid the initial proliferation of analysis due to homography.

Figure 2 shows the result of lexically processing the word *muestra*, and the resulting smooshed record. In this record, each part-of-speech (noun and verb, in this case) is an attribute, the value of which is itself a record. The verbal record contains all the morphosyntactic features or bits of information corresponding to the different morphological possibilities of the verbal form: in this case, '2nd' and '3rd' person, and 'indicative' and 'imperative' mood. The analysis can also use the information that a certain word has more than one part-of-speech, in order to assign probabilities to the syntactic tree.

[5] This procedure is described in [Jiménez, 2001].

[6] The set of subcategorization features is an expanded version of the one used in *Longman Dictionary Of Contemporary English* (*LDOCE*)

```
{Word       "muestra"
 Noun
      {Segtype    NOUN
       Lex        "muestra"
       Lemma      "muestra"
       Bits       Pers3 Sing
                  Count Fem
       Prob       0.00217
       Infl       Noun-casa}
 Verb
      {Segtype    VERB
       Lex        "muestra"
       Lemma      "mostrar"
       Bits    Pers2 Pers3 Sing Pres
               I0 D1 T1 D5 T5 RegPl
               Indicat Imper
       Prob       0.00373
```

Figure 2: Example of smooshed lexical record for the word *muestra*

To increase the robustness of the analyzer, whenever a lemma is not found in the dictionary, a value for part-of-speech is assigned by default. This value is ‘Noun’. Other morphosyntactic features, such as ‘masculine’ and ‘singular’ are also assigned by default. The unfound word is also marked with the bit ‘Unfound’.

The Spanish morphological component currently contains 80 inflectional rules and approximately 200 derivational rules.

2.4 Syntactic Analysis: Sketch and Portrait

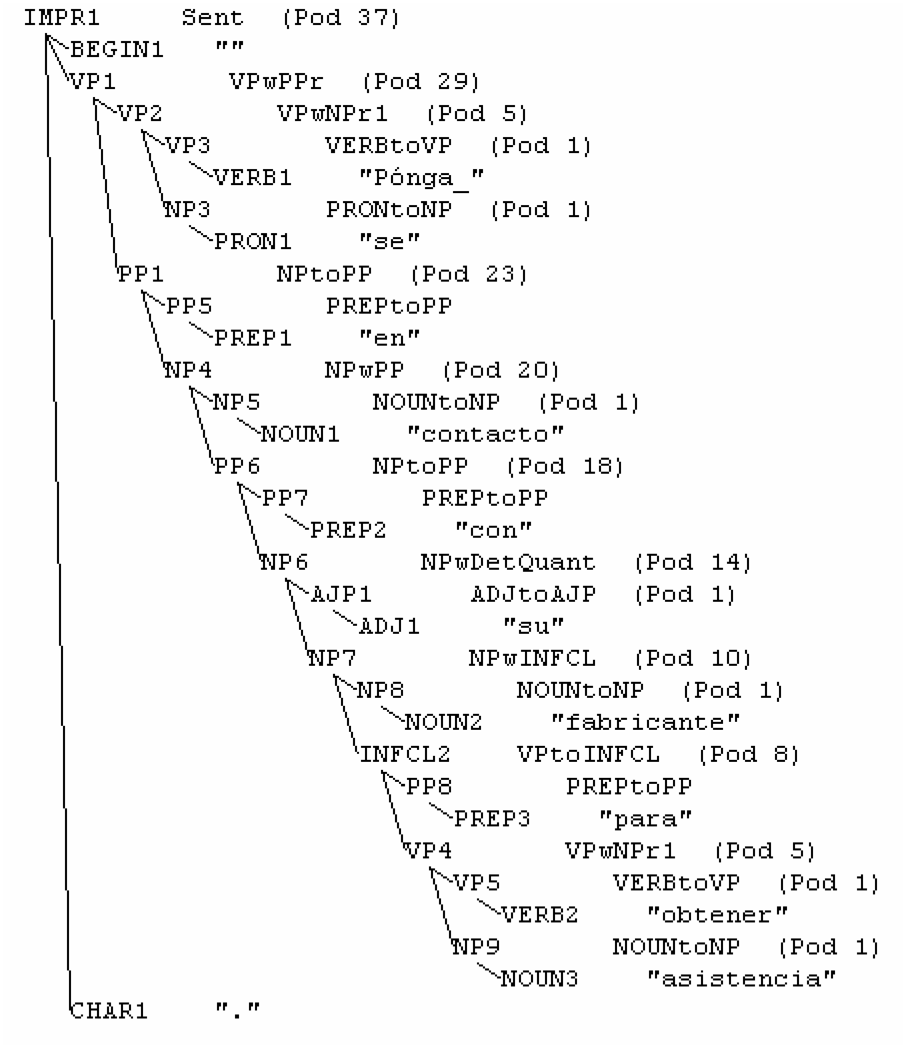
2.4.1 Sketch

The second stage of processing is the parsing component, which is known as *Sketch* because it provides a basic or “sketchy” syntactic parse of the input sentence. Parsing is performed by a set of augmented phrase structure grammar (APSG) rules written in G. The Spanish Sketch grammar consists of 139 rules, most of which are binary, with the remaining few being unary.

The Sketch grammar does a bottom up, parallel parse of the input and produces one or more ranked output analyses. Its input is the list of word records produced by the lexical processing. Grammar rules are used to combine the word records into constituent records, and then to combine these constituent records together to form ever-larger constituent records. The process continues until one or more complete trees are produced.

Figure 3 shows a derivational Sketch tree for the sentence *Póngase en contacto con su fabricante para obtener asistencia*. The derivational tree is an intermediate representation, apt for debugging purposes, that displays the rules that have applied to produce each node.

The name of the rule appears to the right of the node. For example, the rule VPwPPr has applied to VP2 and PP1 to produce VP1. The numbers in parentheses (e.g. Pod 37) indicate the relative weight or probability of the rule (see below on automatically computed probabilities).

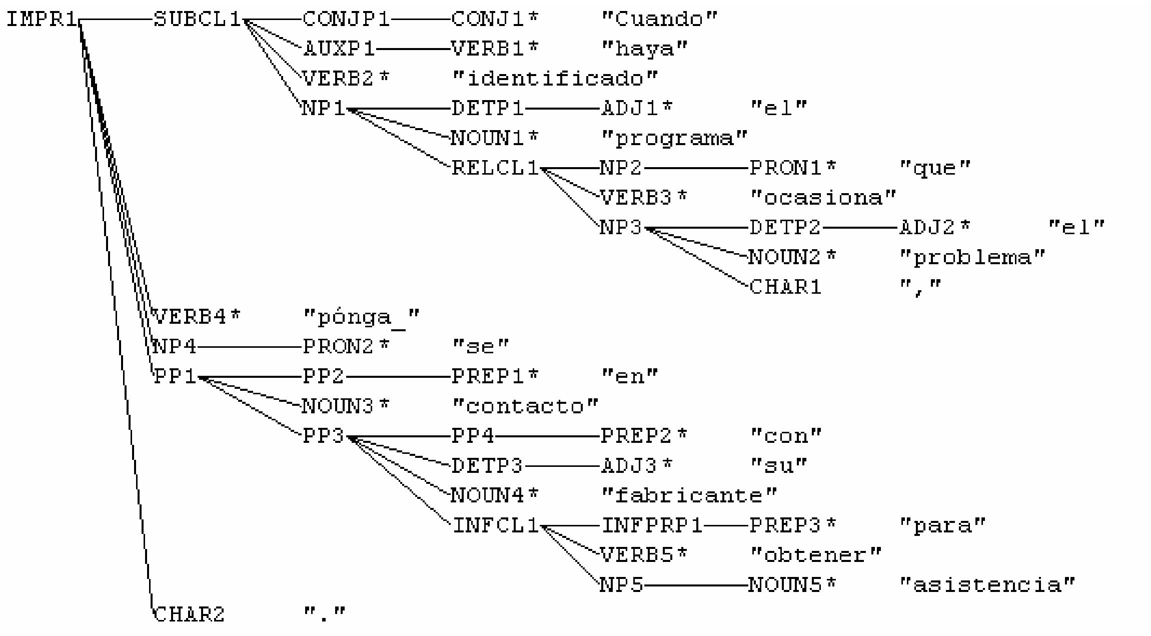


```
IMPR1     Sent   (Pod 37)
  BEGIN1    ""
  VP1       VPwPPr   (Pod 29)
    VP2       VPwNPr1   (Pod 5)
      VP3       VERBtoVP   (Pod 1)
        VERB1     "Pónga_"
      NP3       PRONtoNP   (Pod 1)
        PRON1     "se"
    PP1       NPtoPP   (Pod 23)
      PP5       PREPtoPP
        PREP1     "en"
      NP4       NPwPP   (Pod 20)
        NP5       NOUNtoNP   (Pod 1)
          NOUN1     "contacto"
        PP6       NPtoPP   (Pod 18)
          PP7       PREPtoPP
            PREP2     "con"
          NP6       NPwDetQuant   (Pod 14)
            AJP1      ADJtoAJP   (Pod 1)
              ADJ1      "su"
            NP7       NPwINFCL   (Pod 10)
              NP8       NOUNtoNP   (Pod 1)
                NOUN2     "fabricante"
              INFCL2    VPtoINFCL   (Pod 8)
                PP8       PREPtoPP
                  PREP3     "para"
                VP4       VPwNPr1   (Pod 5)
                  VP5       VERBtoVP   (Pod 1)
                    VERB2     "obtener"
                  NP9       NOUNtoNP   (Pod 1)
                    NOUN3     "asistencia"
  CHAR1     "."
```

Figure 3: Derivational tree (intermediate representation that shows the application of the Sketch gramamr rules)

The derivational tree is useful for debugging purposes, it is generally used to verify which rules have applied when, however it is not the representation that is passed on to the next step. The true output of the Sketch grammar –the so-called Sketch tree- is automatically obtained from the derivational tree by flattening much of its structure.

The Sketch tree is a *syntactic annotated tree* with somewhat of a dependency flavor. In Figure 4 we show the Sketch tree for the sentence: *Cuando haya identificado el programa que ocasiona el problema, póngase en contacto con su fabricante para obtener asistencia*.



```
IMPR1-----SUBCL1----CONJP1-----CONJ1*     "Cuando"
                  \-AUXP1------VERB1*     "haya"
                  \-VERB2*     "identificado"
                  \-NP1--------DETP1------ADJ1*     "el"
                         \-----NOUN1*     "programa"
                         \-----RELCL1-----NP2--------PRON1*     "que"
                                      \---VERB3*     "ocasiona"
                                      \---NP3--------DETP2------ADJ2*     "el"
                                             \-------NOUN2*     "problema"
                                             \-------CHAR1      ","
     \----VERB4*     "pónga_"
     \----NP4--------PRON2*     "se"
     \----PP1--------PP2--------PREP1*     "en"
            \--------NOUN3*     "contacto"
            \--------PP3--------PP4--------PREP2*     "con"
                        \-------DETP3------ADJ3*     "su"
                        \-------NOUN4*     "fabricante"
                        \-------INFCL1-----INFPRP1----PREP3*     "para"
                                      \----VERB5*     "obtener"
                                      \----NP5--------NOUN5*     "asistencia"
     \----CHAR2      "."
```

Figure 4: Sketch tree

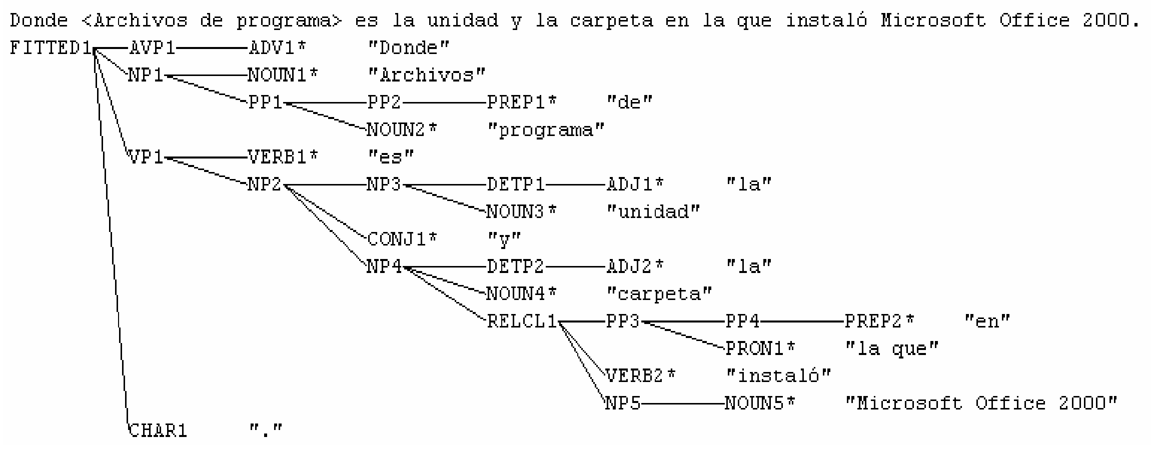
Each syntactic node has a lexical head (e.g. VERB4 is the head of IMPR), and it may have pre-modifiers (Prmods) and/or post-modifiers (Psmods) (e.g SUBCL1 is pre-modifying VERB4, and NP4, PP1 and CHAR2 are postmodifying it).

Since the goal was to develop a broad coverage Analysis system that could handle unrestricted text and that could be used in different NLP applications, such as grammar checking, the Analysis grammar is robust and has been designed to accept ill-formed as well as well-formed input. It makes scarce use of subcategorization information at this point and generally attaches the modifier to its closest constituent, following a strategy of *minimal attachment*.

NLPWin uses probabilities to guide the search algorithm used by the parser [Lozano and Melero, 2001]. This combination of rule-based approach and statistics has many benefits: it greatly improves the system's speed and it allows the grammar to be very forgiving and allow unusual and rare sentences while still favoring the more normal (i.e. probable) analyses. For a sentence that has multiple valid analyses, it is able to produce the most probable one first. The basic idea is to assign a probability to each rule. During parsing, NLPWin tries rules that are more probable first. In addition, probabilities are assigned to each part of speech for the word. Again, NLPWin considers the ones that are more probable first. The system halts as soon as it finds *n* complete parse trees (where *n* is a system parameter that defaults to one).

One interesting aspect of this approach is that the values for the probabilities are computed using NLPWin, i.e. the parser is used to bootstrap itself. Readers can find details of this approach in [Richardson, 1994]. To calculate probabilities, a large Spanish corpus was parsed using NLPWin. Sentences that could not be parsed[7] or produced multiple parses were discarded. Since resulting parses were not manually verified, this strategy follows the assumption that parses with unique results are more likely to be good parses. Probabilities were then computed by counting how often a rule or word sense was used in the final parse tree.

The Sketch grammar always produces at least one parse, even when the input is ungrammatical or the grammar rules are unable to handle it. In that case, the grammar does the best job it can on the pieces of the tree and then produces as a result what is called a *fitted* tree. Fitted trees, such as the one shown in Figure 5, consist of a collection of parsed nodes that no grammar rule can assemble together. The sentence shown in Figure 5 is in fact ungrammatical since it lacks a main verb.



```
Donde <Archivos de programa> es la unidad y la carpeta en la que instaló Microsoft Office 2000.
FITTED1----AVP1--------ADV1*     "Donde"
       \---NP1---------NOUN1*    "Archivos"
       |          \----PP1-------PP2--------PREP1*    "de"
       |                   \-----NOUN2*    "programa"
       \---VP1---------VERB1*    "es"
       |          \----NP2-------NP3--------DETP1------ADJ1*     "la"
       |                 \            \-----NOUN3*    "unidad"
       |                  \------CONJ1*    "y"
       |                   \-----NP4--------DETP2------ADJ2*     "la"
       |                              \-----NOUN4*    "carpeta"
       |                               \----RELCL1-----PP3--------PP4--------PREP2*    "en"
       |                                          \          \----PRON1*    "la que"
       |                                           \---VERB2*    "instaló"
       |                                            \--NP5--------NOUN5*    "Microsoft Office 2000"
       \---CHAR1     "."
```

Figure 5: Fitted parse

Fitted parses although no true grammar structures can still be used as input in applications such as Grammar Checking or Machine Translation.

[7] I.e. resulted in fitted parses .

2.4.2 Portrait

The third stage of processing is known as *Portrait*. The Portrait grammar *refines* the output of the Sketch grammar. At this stage some dependency relations in the Sketch tree may be altered, on the basis of syntactic information encoded in the lexicon, such as bound prepositions and subcategorization frames. The Portrait component uses this information to produce a more accurate tree attachment of constituents such as prepositional phrases, relative clauses, or infinitive clauses. This reattachment is purely syntactic. A semantic reattachment such as described in [Jensen, 1993], which would deal with cases such as *Vi un pájaro con el telescopio*[8] by means of lexical and semantic relations, although foreseen when NLPWin was first devised has never been used in MSR-MT or other NLPWin based applications.

Figure 6 shows the Portrait version of the Sketch tree that we saw in Figure 4. In this version the infinitive clause headed by the preposition *para* (node INFCL1) has been moved from PP3 to the clause level (node IMPR1).

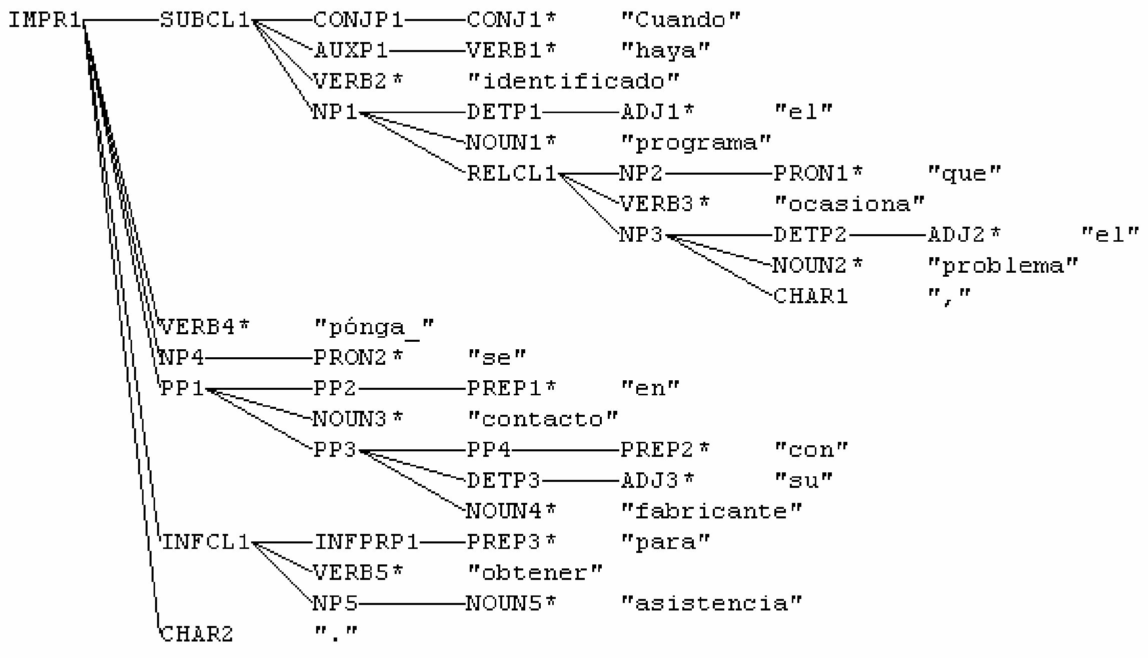


Figure 6: Portrait tree

[8] *Con el telescopio* may be syntactically attached to *pájaro* or the verb *ver*; but semantically it would naturally go with the verb *ver*.

Each node in the tree is actually a record that contains all the information computed so far for that segment of the input string. Some of the bits come directly from the lexical information of the head and some are computed by the Analysis rules.

Records are the basic data structures used by all the components in NLPWin (they are also called *segrec*, or segment record). A record is a set of attribute-value pairs, where the values can be simple ('strings', 'integers', 'atoms'[9]) or arbitrarily complex (i.e. nested records). Attributes can also have as value a list, such as a list of atoms or a list of records. Records are used to represent the tokens produced by the lexical processing, the syntactic nodes in the parse tree, and the nodes in the logical form. The contents of the segment record for the top node (IMPR1) of the tree in Figure 6 are shown in Figure 7 below.

Attributes are listed in a column on the left. Their values are listed on the right. The attributes that are connected to their values through a dotted line are attributes whose value is a record (e.g. Head) or a list of records (e.g. Prmods or Psmods). Values between quotes are of type string (e.g. Lemma). A bracketed list of values (e.g. Vprp) is a list of atoms.

The `Bits` attribute contains a list of Boolean-valued variables (known as bits), which carry the linguistic information relevant to that record. Bits can be turned on and off and can also be tested for equality and inequality. G allows referring to bits in the rules using an implicit notation. For example, +Sing, is equivalent to Sing(Bits)=1 and it amounts to setting the Sing(ular) value on for that record.

All the information present in the *current record*[10] can be accessed by the grammar rules at any given point. Other records can be accessed through the attributes that have another record as value (e.g. Subject). More specifically, through Prmods and Psmods, the whole tree can be accessed or visited at any given time of the process from any record in the tree.

[9] An atom is a basic, indivisible value. The only operations defined on atoms are equality and inequality.

[10] By *current record*, we mean the record on which a rule is being applied at a given point of the process.

```
{Segtype    SENT
 Nodetype   IMPR
 Nodename   IMPR1
 Ft-Lt      1-21
 String     "Cuando haya identificado usted el programa que
             ocasiona el problema , pónga_ usted se en contacto
             con su fabricante para obtener asistencia ."
 CopyOf     VP1
 Rules      (PrLF_VerbInfo_Pron_Subj Sent VPwSUBCLl VPwPPr VPwNPr1
             VERBtoVP)
 Constits   (IMPR1 BEGIN1 VP1 CHAR2)
 Lex        "pónga_"
 Lemma      "poner"
 Bits       Reflex Pers2 Sing
            Closed X9 I0 D1 T1 I3
            T5 Infld Mov UnderSubj
            Imper Polite EstarA
 Prmods-----SUBCL1 "Cuando haya identificado usted el programa que
                   ocasiona el problema ,"
 Head-------VERB4 "pónga_"
 Psmods-----NP10 "usted"
            NP4 "se"
            PP1 "en contacto con su fabricante"
            INFCL1 "para obtener asistencia"
            CHAR2 "."
 Subject----NP10 "usted"
 Pod        85
 Prob       0.99770
 Infin------INFCL1 "para obtener asistencia"
 Inverts----SUBCL1 "Cuando haya identificado usted el programa que
                   ocasiona el problema ,"
 Nargs      1
 FrstV------VERB4 "pónga_"
 Vprp       (en de)
 Bitrecs
       {Bits       T1 Mov EstarA
        Infl       Verb-poner
        Vprp       (en de)
        LexemeID   1 }
 SemNode    poner1
 AmbNP-------NP4 "se"
 Infprp     (a)
 Mrphrecs
       {Bits       Pers2 Sing Imper
                   Polite }
 Cltsright  NP4 "se"
 Clitics----CLITIC1 "se"
 Clform     póngase }
```

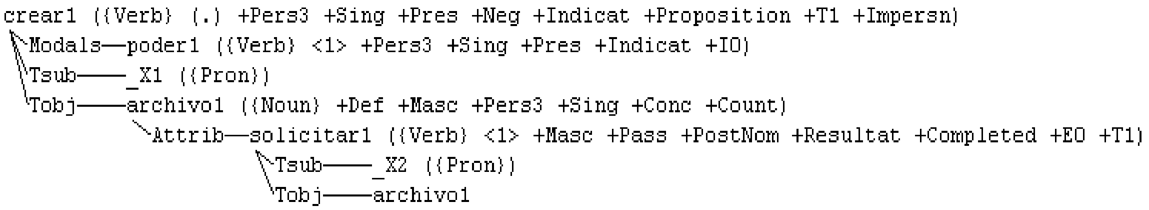
Figure 7: Segment record for the IMPR node

2.5 Logical Form

The fourth stage of the Analysis process produces an annotated predicate-argument structure in the form of a labeled directed graph, called *Logical Form* or LF.

Logical Form represents the logical arrangement of the parts of a sentence, independent of arbitrary, language-particular aspects of structure such as word order, inflectional morphology, function words, etc. Surface variations such as passive/active forms are neutralized at this level.

Figure 8 shows the Logical Form for the sentence *No se puede crear el archivo solicitado*.



```
crear1 ({Verb} (.) +Pers3 +Sing +Pres +Neg +Indicat +Proposition +T1 +Impersn)
 \Modals—poder1 ({Verb} <1> +Pers3 +Sing +Pres +Indicat +IO)
 \Tsub———_X1 ({Pron})
 \Tobj———archivo1 ({Noun} +Def +Masc +Pers3 +Sing +Conc +Count)
          \Attrib—solicitar1 ({Verb} <1> +Masc +Pass +PostNom +Resultat +Completed +EO +T1)
                   \Tsub———_X2 ({Pron})
                   \Tobj———archivo1
```

Figure 8: Logical Form for *No se puede crear el archivo solicitado*

The nodes in the graph are labeled with the *lemmas*, or root forms of the content words of the sentence (*crear, poder, archivo, solicitar*, in the example in Figure 8). Arcs connect parents with children nodes. The labels on the arcs tell the relation of the children with respect to the parent. Some of them represent *deep* grammatical functions such as logical subject or Tsub, or logical object or Tobj.

A node can also have a label that does not correspond to any string or lemma in the surface input. In this example, _X1 and _X2 are the unspecified logical subjects of `crear1` and `solicitar1`, respectively.

The number to the right of the lemmas is an index that indicates the actual node. Whenever two distinct words in a given sentence have the same lemma, each word gets a different index. In the LF shown in Figure 8, the word *archivo* appears twice, with the same index in both cases. It is, in fact, the graphical way to express that the node is one and the same, and that it has two different parents (namely, `crear1` and `solicitar1`)[11].

Function words, such as the negation adverb *no*, the definite article *el*, or the reflexive pronoun *se*, are omitted altogether, often replaced by features (+Neg, +Def and +Impersn, respectively, in the example).

In the LF in our example, *crear* has been identified as being the head of the predication, and *poder* is stored in the attribute Modals.

As an abstract syntactic representation, Logical Form is similar to other deep syntactic representations, such as DSyntS [Lavoie and Rambow, 1997] based on Dependency Syntax [Mel'čuk, 1988]. Similarly to LF, DSyntS neutralizes surface word order and certain functional information, and tries to encode syntactic structure using a language-neutral formal vocabulary. On the one hand, both make use of labeled arcs, their nodes are always terminal nodes and they are labeled with lemmas. Also, both are based on words and not on word senses, unlike other representations, such as F-structure in Lexical Functional Grammar [Bresnan, 2002]. On the other hand, their representations differ in that DSyntS is an unordered tree while LF is a graph (i.e. nodes can have more than one parent).

As was the case in previous processing stages, each node in the graph is a record that contains the information collected during the Analysis process. Figure 9 shows the record for the root node of the LF in Figure 8.

```
{Nodename   crear1
 Rules      (SrLF_dobj_to_tobj SrLF_dsub_to_tsub SynToSem1)
 Constits   (crear1 crear1 DECL2)
 Lemma      "crear"
 Bits       Pers3 Sing Pres Neg T1
            Indicat Proposition
            Impersn
 SynNode----DECL2 "No se puede crear el archivo solicitado ."
 Cat        Verb
 Pred       crear
 Tobj-------archivo1
 Tsub-------_X1
 Modals-----poder1
 SentPunc   (.) }
```

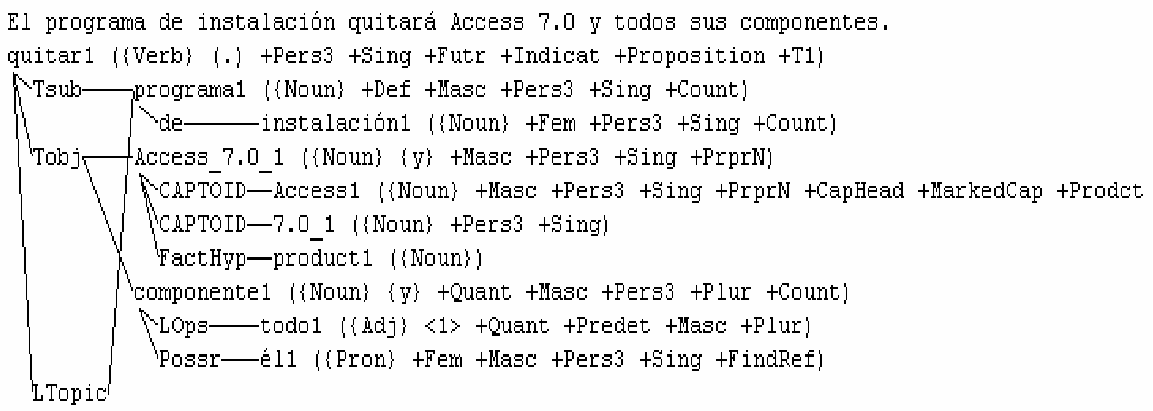
Figure 9: Record of the root LF node *crear1*

[11] As mentioned above, an LF is a graph and not a tree; therefore, nodes can have more than one parent.

LF records look simpler and more compact than those of Sketch. In spite of that, all the information collected during Sketch is expressed in Logical Form in one way or another. LF normalizes syntactic variation but does not lose information about the syntactic variations that are significatively distinct. For instance the bit 'Pass' carries the information that a surface sentence is in *passive* form[12].

Here, too, attributes connected to their values with a dotted line are attributes whose values are themselves a record.

Logical relations, such as Tsub or Tobj, are attributes whose value is a *list of records*[13]. The motivation for this is to give the same representation to coordinated and non-coordinated arguments[14]. Coordination is expressed as a list of nodes. Non-coordinated arguments, on the other hand, are lists of one element. This allows rules to be more general. In the example shown in Figure 10, the Tobj of the sentence: *El programa de instalación quitará Access 7.0 y todos sus componentes*, is a list containing two elements: `Access_7.0_1` and `componente1`.



```
El programa de instalación quitará Access 7.0 y todos sus componentes.
quitar1 ({Verb} (.) +Pers3 +Sing +Futr +Indicat +Proposition +T1)
 Tsub———programa1 ({Noun} +Def +Masc +Pers3 +Sing +Count)
           de————instalación1 ({Noun} +Fem +Pers3 +Sing +Count)
 Tobj———Access_7.0_1 ({Noun} {y} +Masc +Pers3 +Sing +PrprN)
           CAPTOID—Access1 ({Noun} +Masc +Pers3 +Sing +PrprN +CapHead +MarkedCap +Prodct
           CAPTOID—7.0_1 ({Noun} +Pers3 +Sing)
           FactHyp—product1 ({Noun})
        componente1 ({Noun} {y} +Quant +Masc +Pers3 +Plur +Count)
           LOps———todo1 ({Adj} <1> +Quant +Predet +Masc +Plur)
           Possr———él1 ({Pron} +Fem +Masc +Pers3 +Sing +FindRef)
 LTopic
```

Figure 10: Coordination in LF

[12] While certain word order variations are recorded in LF, such as topicalization and dislocation, other less relevant order variations are lost, such as clitic raising, as in: *Quiero decirle que venga* vs *Le quiero decir que venga*, or position of modifying adverbials, for example.

[13] By contrast, syntactic relations in Sketch, such as Subject or Object, are of type record.

[14] Since all nodes in LF are terminal, there cannot be an intermediate coordinated node.

Table 1 contains the list of the main logical relations in Logical Form. All are of type list of records.

Relations	**Meaning**	**Examples**[15]
Tsub	logical subject	El **archivo** está disponible; El equipo ha sido afectado por el **virus.**
Tobj	logical object	El **equipo** ha sido afectado por el virus; Se describen dos **métodos.**
Tind	logical indirect object	A todos los **archivos** se les asigna la fecha actual.
Lcmp	object complement	Mantenga **presionada** la tecla**;** Tiene **instalada** la opción; Es un archivo llamado **Test.txt;** Estos se denominan **programas** residentes;
Modals	modal or aspectual operators	El fichero **sigue** estando disponible; **Puede** tener acceso; **Deben** instalarse las funciones; **Vuelva** a enviar el fax; Se **va** a instalar Internet Explorer
Locn	location	Este archivo está ubicado en la **carpeta** \Windows.
Time	time	Siga los pasos citados **anteriormente**; Puede tardar varios **minutos**; Espere mientras se **cierra** el sistema
Purp	purpose clause	Haga clic en Aceptar para **continuar**; Se ofrecen consejos para que las distintas versiones no **entren** en conflicto.
Conditn	condition clause	Si el Asistente no se **inicia**, cierre Outlook;
Duration	duration	Se bloquea durante varios **minutos**
Cause	cause	Esto ocurre porque **hizo** clic en No.
Manner	manner	Explica **cómo** instalar Symantec; Se abren **utilizando** un formulario; No puede acceder **sin** autorización
PrepRel	semantically unspecifed prepositional complement	La configuración varía según las **características**; Haga clic en **Aceptar**
Possr	possessor	**su** sistema; el archivo del **equipo**
Attrib	attributive modifier (adjective, relative clause, or similar function)	la ubicación **original**; los criterios que se **describen;** los archivos **utilizados**
LOps	quantifier/determiner	los **dos** casos; el **otro** equipo; **todas** las partes; **grupos** de usuarios; **muchos** de los archivos; **parte** de sus funciones; **más** facilidad; **cualquier** software
Appostn	appositive (mostly Proper nouns)	el dígito **+1;** el menú **Herramientas**; el panel **Quitar** versión anterior; VBScript **versión** 5.0
Intnsifs	intensifier	**muy** rápida.
Classifier	classifier; often this is the grammatical head but not the	**familias** de procesadores; 5 **megabytes** de espacio; **instancia** del programa

[15] The head of the Attribute is marked in bold.

	logical head	
Mod	otherwise unresolved modifier, esp. adverbs, also nouns	Esta información **también** se encuentra en el Kit; programa del equipo **servidor**
Props	semantically unspecified clause, generally preceded by a preposition	Tras **instalar** Fax Starter Edition, elimine Outcmd.dat; Hay dos maneras de **resolver** este mensaje; Una vez **finalizada** la reinstalación, haga clic en el cuadro de diálogo; La aplicación continúa sin **aparecer** en la lista
SMods	unresolved sentence-level modifiers (mostly prepositional, also adverbs)	En otro **caso**, uno de los programas está ocasionando el problema.

Table 1: Relation labels in Logical Form

All nodes in a Logical Form, except for the root -such as crear1 in Figure 9 -, have the attribute Parents, which contains the list of nodes that are parents to that node.

Figure 11 shows the record `archivo1` of the example in Figure 8 (*No se puede crear el archivo solicitado*). In it, the attribute Parents is a list containing two elements: the main verb -`crear1`- and the past-participle `solicitar1`.

```
{Nodename   archivo1
 Rules      (SynToSem1)
 Constits   (NP2)
 Lemma      "archivo"
 Bits       Def Masc Pers3 Sing
            Conc Count
 SynNode----NP2 "el archivo solicitado"
 Cat        Noun
 Pred       archivo
 Parents----crear1
            solicitar1
 Attrib-----solicitar1 }
```

Figure 11: Contents of LF record `archivo1`

Logical Form is the last step of the Analysis process and the most abstract representation of the linguistic content in NLPWin. It is also the representation of the aligned data from which the bilingual Mindnet is created in the Alignment stage, which we describe below.

2.6 Alignment and Transfer

During the training phase or **Alignment**, source and target sentences from the aligned bilingual corpus are parsed to produce LFs. The normalized word forms resulting from parsing are also fed to a statistical word association learner which outputs learned single word translation pairs as well as multi-word pairs.

LFs are then aligned with the aid of translations from a bilingual dictionary and the learned single word pairs. The LF alignment algorithm first establishes tentative lexical correspondences between nodes in the source and target LFs using translation pairs from a bilingual lexicon[16]. After establishing possible correspondences, the algorithm uses a small set of alignment grammar rules to align LF nodes according to both lexical and structural considerations and to create LF transfer mappings. The final step is to filter the mappings based on the frequency of their source and target sides. [Menezes and Richardson, 2001] provide further details and an evaluation of the LF alignment algorithm.

The English-Spanish bilingual training corpus consists largely of Microsoft manuals and help text, and contains around 340K sentences.

Transfer mappings resulting from LF alignment, in the form of linked source and target LF segments, are stored in a special repository known as MindNet. [Richardson et al., 1998] describes how MindNet began as a lexical knowledge base containing LF-like structures that were produced automatically from the definitions and example sentences in machine-readable dictionaries.

Later, MindNet was generalized, becoming an architecture for a class of repositories that can store and access LFs produced for a variety of expository texts, including but not limited to dictionaries, encyclopedias, and technical manuals.

Figure 12 schematically shows the training process, where the bilingual mappings in MindNet are created (Alignment phase) and the runtime process where MindNet is consulted to produce translations (Transfer phase).

[16] The English/Spanish lexicon contains 88,500 translation pairs.

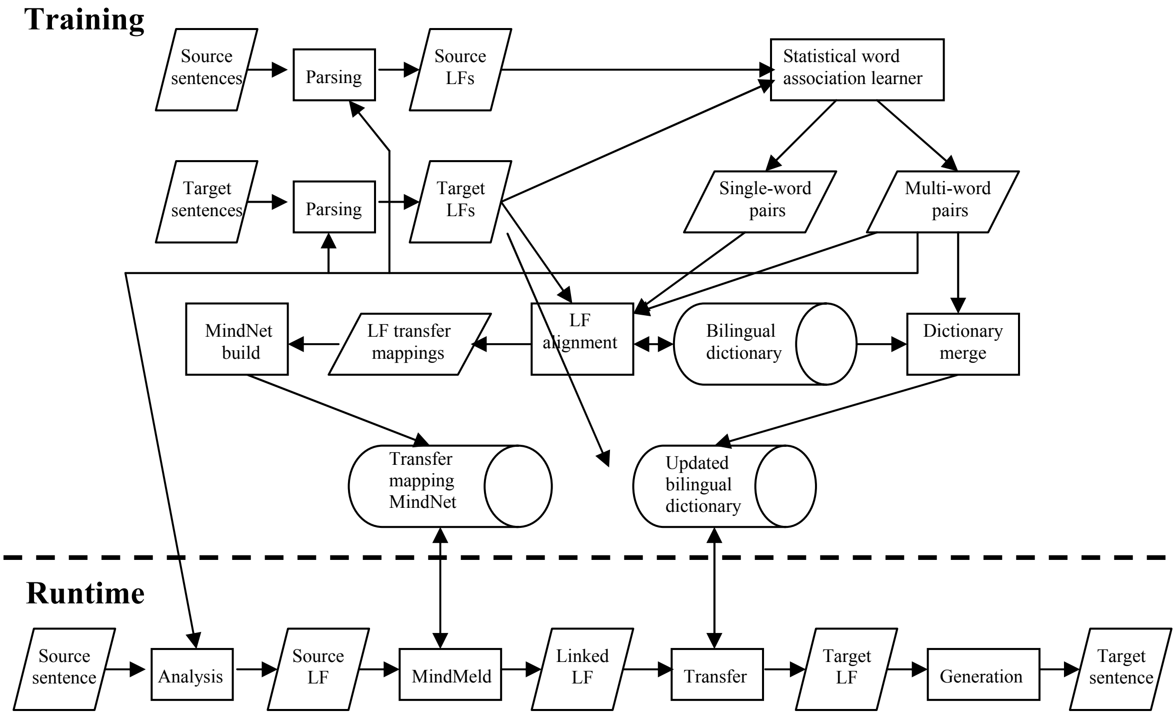


Figure 12: Training and run time process[17]

At runtime, during the phase known as **Transfer**, source sentences are parsed by the Analysis grammar up to Logical Form. These LFs then undergo a process (known as *MindMeld*), which matches them against the LF transfer mappings stored in MindNet. Larger (more specific) mappings are preferred to smaller (more general) mappings. In other words, transfers with context will be matched preferentially, but the system will fall back to the smaller transfers when no matching context is found. Among mappings of equal size, MindMeld prefers higher-frequency mappings.

This strategy allows Transfer to produce higher quality translations that are more sensitive to context, specific to the type of data on which it has been trained, than conventional hand-written Transfer rules[18].

[17] Figure reproduced from [Richardson et al, 2001a]

[18] By way of example, the following are real MSR-MT translations of the same verb (*activar*) in different contexts:

Active el contador.	Activate the counter.
Active la seguridad de IP.	Enable the IP security.

After matching source LFs, MindMeld also *links* them with corresponding target LF segments stored in MindNet. These target LF segments are stitched together into a single target LF during Transfer. In cases where no applicable transfer mapping was found during MindMeld, the nodes in the source LF and their relations are simply copied into the target LF. Bilingual dictionaries, containing only word pairs and their parts of speech, provide translation candidates for the alignment procedure and are also used as a backup source of translations during Transfer[19].

Generation is the last stage of the MSR-MT system. It receives the *stitched* target LF as input, from which it produces a gramamtical target sentence. The Generation process is described in detail in Chapter 3.

Active la captura.	Raise the trap.
Active el desencadenador.	Fire the trigger.
Active el indicador.	Set the flag.
Active la casilla de verficación.	Select the check box.

[19] Some experiments have shown that the MT system does equally well without using a full bilingual dictionary. [Pinkham and Smets, 2002] shows that by using automatically derived translation word pairs combined with a function word only lexicon, the results either matched or nearly matched the translation quality of the system that used a full traditional bilingual lexicon in addition. The language pairs studied were French-English and Spanish-English.

Chapter 3
The Spanish Generation Grammar

In this chapter, we describe a large-scale, wide-coverage Generation grammar for Spanish. This grammar is the core of the Generator that is currently being used by MSR-MT.

The Spanish Generation grammar is a rule-based Syntactic Realizer, at sentence level, that takes as input a predicate-argument structure with deep syntactic information, called Logical Form (LF) -described in Section 2.5-, and then outputs a linearly ordered surface syntax tree with fully inflected leaves. The target sentence is then automatically read off this tree.

In the design of the Generation module, we take into account the sequence of the application of the rules as well as the availability of the information at every point in the process.

3.1 Generation Component

3.1.1 Input, Output and Data

Although the Generation grammar described here is currently being used by a Machine Translation application such as MSR-MT, it has not been specifically written for MT, but is rather intended to be application independent.

It has been designed to apply to a range of tasks, including question answering, dialog systems, database querying, grammar checking, machine translation, etc.

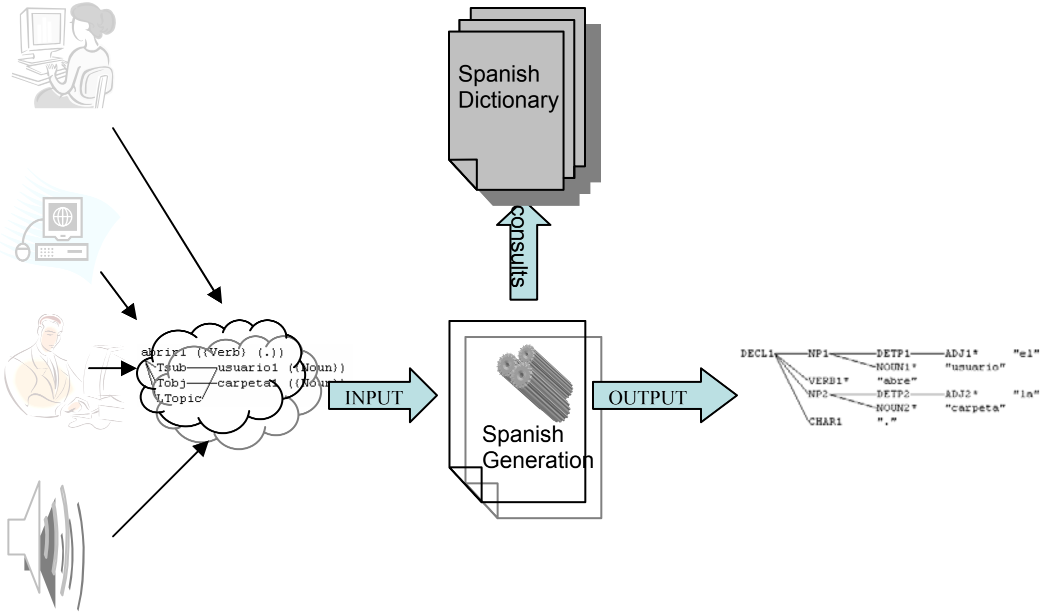


Figure 13: Input/Output of the Generation process

The Generation grammar takes as input a well-formed Spanish LF that may have been produced by different types of applications or, in the case of MT, be the translation of any language.

The information available to the Generation component comes exclusively from two sources: (i) the input Logical Form; (ii) and the (monolingual) Spanish dictionary, which is the same repository of lexical information used by the Spanish Analysis grammar. From these two sources, Generation rules produce a well-formed, linearly ordered syntactic tree enriched with inflectional information.

3.1.2 Coverage of the Generation Grammar

3.1.2.1 Text type coverage

The NLP system that uses this Generation component has been conceived as a large-scale, wide-coverage, general-purpose system; consequently, it covers the full range of linguistic phenomena in Spanish and is able to deal with unrestricted text.

Similarly, this Spanish Generation grammar aims at covering as many Spanish structures as the Spanish Analysis grammar is able to produce. This is achieved by means of a comprehensive set of reference sentences that guides the development of the grammar. The testbed used for testing and developing the grammar comprises thousands of sentences, more precisely a corpus of 14,000 is used for monolingual testing and another of 9,000 for bilingual testing, i.e. a total of 23,000 sentences. This test corpus has been collected from the following sources:

- Grammar textbooks
- Technical documents (Microsoft)
- Newspapers
- Encyclopedia (Encarta)

For evaluation purposes, we have mainly used technical documents not included in the reference set.

3.1.2.2 Grammatical coverage

The size of the test corpus (23,000 sentences) already is a guarantee that the grammar that we are presenting has a wide coverage. However, we have also used a test suite extracted from grammar textbooks which aims at exhaustively covering a complete list of grammatical phenomena, and which has served as touchstone for completing coverage. The list that we present here[20] is a list of Spanish phenomena; contrarily to Analysis, Spanish is our *output*, not our *input*. This may render complicate measuring coverage in terms of a list of phenomena. However, as we will see in the next section, our procedure for developing the grammar (starting from *monolingual* Generation) makes full use of the grammatical list approach.

1. Main clause: 0 arguments [weather verbs]

[20] This is the list of phenomena, where each phenomenon has been illustrated with examples, it is not the actual grammatical test suite.

E.g. Llovía.   [It rained.]

2. Main clause: 1 argument

2.1. Intransitive verbs, NP subject

E.g. María corre.   [Mary runs.]

2.2. Intransitive verbs, clausal subject

E.g. Suceden cosas extrañas. [Strange things happen.]

2.3. Impersonal verbs

E.g. Hay problemas. [There are problems.]

3. Main clause: 2 arguments

3.1. Transitive verb, NP Direct Object

E.g. Juan entregó el libro. [He handed over the book.]

3.2. Transitive verb, clausal Direct Object

E.g. Exijo que Juan arregle el coche. [I demand that John repairs the car.]

3.3. Transitive verb, indirect interrogative

E.g. Nosotros sabemos qué compró Juan. [We know what John bought.]

3.4. Copulative verb, nominal attribute

E.g. El doctor es ese hombre alto. [The doctor is that tall man.]

3.5. Copulative verb, adjective attribute

E.g. El doctor es alto. [The doctor is tall.]

3.6. Bound prepositional complement

E.g. Felipe habló con el encargado. [Philip talked to the manager.]

4. Main clause: 3 arguments

4.1. Ditransitive verb, NP DO, PP-a Indirect Object

E.g. Juan entregó el libro a María. [John handed the book to Mary.]

4.2. Ditransitive verb, clausal DO, PP-a IO

E.g. María exigió a Pedro que cambiara de estrategia. [Mary demanded Peter to change his tactics.]

4.3. Transitive verb, NP DO, Object attribute

E.g. El consejo nombró a Juan presidente. [The council appointed John as president.]

4.4. Movement verbs, origin and goal complements

E.g. María iba de casa a la estación. [Mary went from the house to the station.]

4.5. Ditransitive verb, NP DO, pronominal Indirect Object (i.e. clitic)

E.g. Juan le entregó el libro. [John handed him the book.]

4.6. Ditransitive verb, clitic DO, clitic IO

E.g. Juan se lo entregó. [John handed it to him.]

4.7. Ditransitive verb, NP DO, PP-a IO, duplicated clitic IO

E.g. Juan le entregó el libro a la niña. [John handed (her) the book to the girl.]

4.8. Ditransitive verb, clitic DO, PP-a IO, duplicated clitic IO

E.g. Juan se lo entregó a la niña. [John handed it (her) to the girl.]

4.9. Ditransitive verb, dislocated NP DO, PP-a IO, duplicated clitic IO, duplicated clitic DO

E.g. El libro se lo entregó Juan a la niña. [The book John handed it (her) to the girl.]

4.10. Ditransitive verb, dislocated PP-a IO, dislocated NP DO, duplicated clitic IO, duplicated clitic DO

E.g. A la niña, el libro se lo entregó Juan. [To the girl, the book John handed it (her).]

5. Time, place and other modifiers

5.1. Time modifiers

E.g. Juan entregará el libro esta noche. [John will hand over the book tonight.]

E.g. El profesor llegó ayer por la mañana. [The professor arrived yesterday morning.]

E.g. Nosotros tomamos el tren por la mañana. [We took the train on the morning.]

E.g. El profesor habló ayer por dos horas. [The professor talked for two hours.]

E.g. A menudo llueve en España en octubre. [It often rains in Spain in October.]

5.2. Place modifiers

E.g. Hay vino en la cocina. [There is wine in the kitchen.]

E.g. Había muchos libros sobre el escritorio. [There were a lot of books on the desktop.]

5.3. Manner modifiers

E.g. María iba de casa a la estación en tren. [Mary went from the house to the station by train.]

5.4. Other modifiers

E.g. Ella compró el libro por tres dólares. [She bought the book for three dollars.]

6. Prodrop

E.g. Veremos a Juan. [WE will see John.]

E.g. Habló con el encargado. [HE / SHE talked to the manager.]

E.g. Es ese hombre alto. [HE is that tall man.]

E.g. Le entregaste el libro a aquella niña. [YOU (sing.) handed the book to that girl.]

E.g. Se lo entregásteis. [YOU (plural) handed it to her/him.]

7. Negation

E.g. No sabemos qué compró Juan. [WE do not know what John bought.]

E.g. No está lloviendo. [It is not raining.]

E.g. Me entristece que Juan no venga. [It upsets me that John is not coming.]

E.g. Yo no sabía si él había salido. [I did not know whether he had left.]

E.g. Yo no sabía que él había salido. [I did not know that he had left.]

8. Passive sentences (past-participle)

E.g. La carta fue enviada por Juan. [The letter was sent by John.]

E.g. La carta fue enviada. [The letter was sent.]

E.g. Fue enviada por Juan. [IT was sent by John.]

E.g. Ha sido premiado por su labor académica. [HE /SHE has been rewarded for his / her academic work.]

9. Auxiliar structures (perfect, progressive,...)

9.1. Perfective constructions

E.g. Hoy hemos llegado tarde. [Today we have arrived late.]

E.g. Juan habrá tenido un percance. [John must have had a mishap.]

9.2. Progressive constructions

E.g. El hombre está cantando un tango. [The man is singing a tango.]

10. Modals

E.g. Juan puede venir a mi casa. [John can / may come to my house.]

E.g. Juan debe venir a mi casa. [John must come to my house.]

E.g. Juan tiene que venir a mi casa. [John has to come to my house.]

E.g. Juan ha de venir a mi casa. [John has to come to my house.]

E.g. Juan ha empezado a estudiar música. [John has started studying music.]

E.g. Juan solía estudiar música. [John used to study music.]

11. NP structure

11.1. NP modifiers (adjectives, mod PPs, possessive)

E.g. El libro rojo de Mao está en la biblioteca. [Mao's red book is in the library.]

E.g. El doctor es ese hombre alto y delgado. [The doctor is that tall and thin man.]

E.g. La pelota de colores de Juanito está en el jardín. [Johnnie's coloured ball is in the garden.]

11.2. Relative clauses

11.2.1. Subject relative pronoun

E.g. El libro que llegó ayer está aquí. [The book that arrived yesterday is here.]

E.g. Los libros de tu hermana que llegaron ayer están aquí. [Your sister's books, which arrived yesterday, are here.]

11.2.2. Object relative pronoun

E.g. El libro que Juan compró está aquí. [The book that John bought is here.]

E.g. No tenemos nada que decir. [WE have nothing to say.]

11.2.3. Bound prepositional relative pronoun

E.g. El libro del que habló Juan está allí. [The book John was talking about is there.]

11.2.4. Possessive relative pronoun

E.g. El niño cuyos padres vinieron ayer está enfermo.[The child whose parents came yesterday is sick.]

11.3. Possessive

E.g. El niño jugaba con sus primos. [The child played with his cousins.]

11.4. NP coordination

E.g. México ha fortalecido su cooperación e intercambio comercial. [Mexico has strengthened his commercial exchange and cooperation.]

E.g. Tengo un hermano y una hermana en España. [I have a brother and a sister in Spain.]

E.g. Juan y María quieren salir. [John and Mary want to go out.]

12. DETP structure

E.g. Ahora vienen unos pocos niños. [Now, only a few children will come.]

E.g. Luego vendrán todos los demás niños. [Later all the other children will come.]

13. Control structures

13.1. Object control verbs

E.g. Quiero cantar. [I want to sing.]

E.g. Los niños quieren ver la película. [The children want to see the movie.]

13.2. Indirect Object control verbs

E.g. Juan mandó disparar. [John ordered to shoot.]

E.g. Juan mandó disparar a los soldados. [John ordered the soldiers to shoot.]

E.g. Juan nos obligó a estudiar. [John made us study.]

E.g. Juan obligó a María a estudiar. [John made Mary study.]

E.g. Juan le ordena a María que estudie. [John orders (her) Mary that she studies.]

E.g. Juan le prometió a la niña llevarla al parque esta tarde. [John promised (her) the girl to take her to the park this afternoon.]

13.3. Raising adjectives

E.g. Es fácil entender la película. [It is easy to understand this movie.]

E.g. La película es fácil de entender. [This movie is easy to understand.]

E.g. Juan es fácil de convencer. [John is easy to convince.]

E.g. Es fácil convencer a Juan. [It is easy to convince John.]

E.g. Convencer a Juan es fácil. [Convincing John is easy.]

E.g. Juan es incapaz de complacer. [John is unable to please.]

E.g. Juan es difícil de complacer. [It is difficult to please John.]

E.g. Juan es incapaz de convencer a nadie. [John is unable to please anybody.]

13.4. Raising verbs

E.g. Parece que Juan llegó ayer. [It seems that John arrived yesterday.]

E.g. Juan parece haber llegado ayer. [John seems to have arrived yesterday.]

E.g. Oímos a Juan cantar un canción. [We heard John singing a song.]

14. Coordination

E.g. Entré en la tienda y te compré una agenda. [I entered the store and bought you a diary.]

3.1.3 First Stage in Grammar Development: Monolingual Generation

In the first stages of development of the Generation grammar, we have essentially followed the same procedure used to build an Analysis grammar:

- Definition of the linguistic coverage (as described in Section 3.1.2)
- Creation of a *monolingual regression set* (described below in Section 3.1.4)

The first goal of a Generation grammar of a given language is to be able to regenerate the sentences that have been analyzed by the Analysis grammar of that same language. In practice, this means that in the first stages of development, the grammar is tested against a comprehensive test suite of Spanish sentences.

As we will see later, this goal is necessary but not sufficient to be able to use the grammar in real applications, such as Machine Translation. Actually, the Logical Form representations produced by the Analysis grammar of the language in question –Spanish, in our case- contains more information than we can realistically expect to find in Logical Forms coming from other sources, for example, Transfer.

Obviously, the process of analyzing a Spanish sentence in order to regenerate it again does not seem to have any practical application; however, in order to guarantee completeness and coherence of coverage, it is adequate to start building the Spanish Generation grammar based on well-formed Logical Forms produced by the Spanish Analysis grammar.

3.1.4 Assessing Progress in Grammar Development: Regression Testing

Following the same procedure used in the development of other components of the system (see for example [Melero, 2001]), the grammar writer uses part of the available corpus to build a regression system that allows immediate evaluation of every change in the code. The regression system is built in the following way:

1. A set of text files is prepared, containing one sentence per line (of any length)[21]. Typically the sentences are of one of two types:
    a. Manually selected sentences, coming from grammar books, including all grammatical phenomena, used to verify the coverage of the grammar.
    b. Unrestricted text coming from a variety of sources (technical, newspaper, encyclopedia…) not manually verified or selected by the linguist.
2. All files are processed by the latest stable version of the system. In our case, the Spanish sentences are analyzed by the Spanish Analysis grammar up to Logical Form. The resulting files contain the original sentences plus the corresponding Logical Forms: one for each sentence. These are the *master files* to be used to verify subsequent changes to the code.
3. Every time the code is changed, a regression test is run, i.e., the master files are processed with a version of the system that includes the change.
4. When the master files are processed, a new set of files is generated, that contain the differences caused by the change. The differences that are *good* are accepted, i.e., they are incorporated to the master files. The differences that are *bad* –i.e. *regressions*- show undesired side effects of our change. These regressions are then used by the developer as evidence for refining her implementation.

The regression testing provides immediate feedback to the grammar writer about her changes.

3.2 The Generation Process

The Generation process consists of the following steps:

[21] The first stage of processing includes automatically breaking the sentences in a text.

1. Automatically map the input LF onto a *basic syntactic tree*.
2. Apply syntactic Generation rules to the basic tree to produce a *linearly ordered surface tree* whose leaves contain sufficient morphological and formatting information.
3. Apply morphological Generation rules to get the correct *inflection* of the leaf nodes of the surface tree, and formatting rules to get the proper *formatting*.
4. Read off the output string from the inflected leaves.

The Generation grammar described here takes care of step 2 of this process. We introduce it with a brief overview of step 1 (Section 3.2.1) that will be useful to understand how does the input to Generation look like. We will not describe steps 3 or 4.

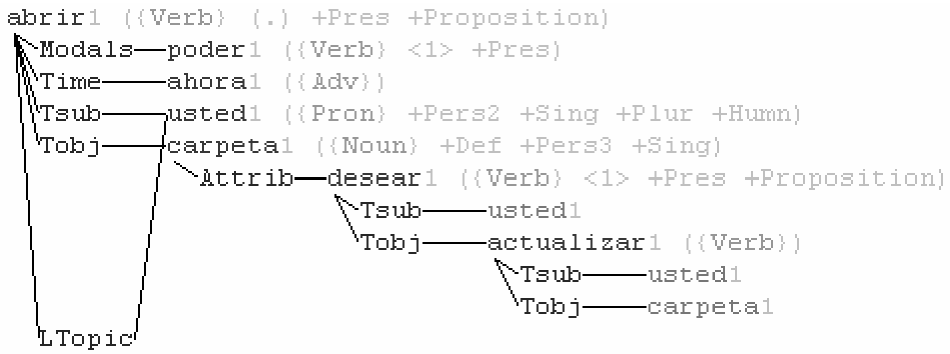
3.2.1 Basic Syntactic Tree: Degraphing of the LF

At the beginning of the Generation process, the input LF is straightforwardly projected onto a basic syntactic tree that conforms to the tree geometry of the NLPWin Analysis system, i.e. a Sketch/Portrait type of tree (see Section 2.4).

The nodes in the LF become subtrees of this tree and the LF relations become complement/adjunct relationships between the subtrees. For Spanish (as well as for English and Chinese) this tree is set up as strictly head-initial with all the complements/adjuncts following the head, resembling the tree of a VSO language[22]. Figure 14 shows the LF for the sentence in example (1), and Figure 15 gives the basic Generation tree produced from that LF.

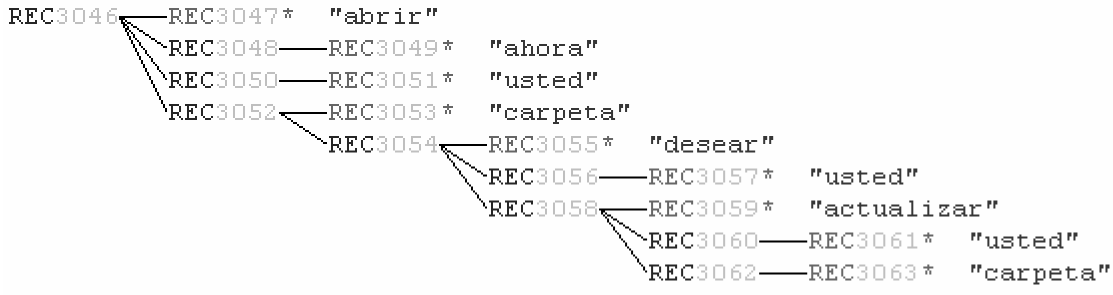
```
(1) Ahora puede abrir la carpeta que desea actualizar.
```

[22] For Japanese and German, it is set up as strictly head-final, with all the complements/adjuncts preceding the head.



```
abrir1 ({Verb} (.) +Pres +Proposition)
  Modals——poder1 ({Verb} <1> +Pres)
  Time————ahora1 ({Adv})
  Tsub————usted1 ({Pron} +Pers2 +Sing +Plur +Humn)
  Tobj————carpeta1 ({Noun} +Def +Pers3 +Sing)
            Attrib——desear1 ({Verb} <1> +Pres +Proposition)
                      Tsub————usted1
                      Tobj————actualizar1 ({Verb})
                                Tsub————usted1
                                Tobj————carpeta1
  LTopic
```

Figure 14: LF for example (1)



```
REC3046——REC3047*  "abrir"
  REC3048——REC3049*  "ahora"
  REC3050——REC3051*  "usted"
  REC3052——REC3053*  "carpeta"
            REC3054——REC3055*  "desear"
                      REC3056——REC3057*  "usted"
                      REC3058——REC3059*  "actualizar"
                                REC3060——REC3061*  "usted"
                                REC3062——REC3063*  "carpeta"
```

Figure 15: Basic tree for the LF in Figure 14

As illustrated by this example, the semantic head is the first child of the parent node at each level of the tree and the rest of the arguments follow as post-modifiers of the head. An intermediate level of empty nodes is created as parent of each terminal node (e.g. REC3048, REC3050,...), except for the head, which hangs directly from the parent node. At this early stage of the Generation process, lemmas are the only information present in this preliminary syntactic tree. The nodes have no syntactic label[23] and no features have yet been recorded.

The LF is a graph; this means that nodes can have more than one parent. In the basic tree, nodes can have *only* one parent. In the process of *degraphing*, nodes that have more than one parent in the LF (i.e. *gap fillers* such as `usted1` and `carpeta1` in Figure 14) are *duplicated* in the basic tree.

It will be up to the Generation rules to decide which nodes should remain and which ones should be removed or turned into coreferential pronouns.

[23] The nodes are conventionally labeled REC plus an internally assigned index. Nodes marked with an asterisk are terminal heads.

Connections between the basic tree and the original Logical Form are maintained. All the information present in the LF, such as *agreement* bits or *modality* attributes may be accessed through an attribute in the basic syntactic tree called *SemNode*, which links each new syntactic node with the corresponding logical node.

At this stage, lexical information based on the value of the lemma is retrieved from the Spanish dictionary and stored in an attribute called –conventionally, but not altogether appropriately- *Temp*.

To sum up, here is the list of the attributes that might or must be present in the basic syntactic tree before the Generation rules apply:

- 'Lemma': every node in the basic tree (both terminal and non-terminal) has a lemma.
- 'SemNode'[24]: every non-terminal node is linked to a node in the original LF via the 'SemNode'.
- 'Temp': every non-terminal node, except those with unknown lemmas, may access the information stored in the monolingual dictionary through 'Temp'; 'Temp' contains a *copy* of the lexical entry for the lemma of the node.
- 'Parent': every node, except for the root, has a 'Parent'.
- 'Head': every non-terminal node has a 'Head', which is always a terminal node.
- 'Psmods'[25]: non-terminal nodes may have complements hanging from a 'Psmods' attribute; 'Psmods' is a list of records, which are always non-terminal nodes.

As an example, Figure 16 shows the contents of a node from the basic syntactic tree in Figure 15; the node in question is the record for the non-terminal projection of the verb "actualizar" or REC3058.

```
{Lemma       "actualizar"
 Head
      {Lemma       "actualizar"
       Parent      REC3058}
```

[24] Stands for "semantic node".

[25] Stands for "post-modifiers".

```
Psmods
      {Lemma       "usted"
       Head         REC3061
       Parent       REC3058
       Temp         REC2885
       SemNode     usted1}

      {Lemma       "carpeta"
       Head         REC3063
       Parent       REC3058
       Temp         REC2892
       SemNode     carpeta1}
Parent
      {Lemma       "desear"
       Head        REC3055
       Psmods      REC3056
                   REC3058
       Parent      REC3052
       Temp        REC2764
       SemNode     desear1}
Temp
      {Segtype     VERB
       Lex         "actualizar"
       Lemma       "actualizar"
       Bits        Inf T1
       Infl        Verb-cruzar}
SemNode    actualizar4}
```

Figure 16: Content of REC3058 in the basic tree from Figure 15

All the information needed by the Generation rules will eventually be retrieved from one of two sources: the monolingual dictionary (via ‘Temp’) and the input Logical Form (via ‘SemNode’). The rest of the attributes provide the geometry of the syntactic tree and allow access to every node in the tree at any given time, from any given node.

All Generation rules will then apply to this basic structure on a top-down (or left-to-right)[26], depth-first manner, removing, creating or moving nodes as needed, as well as incorporating new features which will be used by Generation rules applying later, or, at the end of the process, by the morphological and formatting modules to get the proper inflection and formatting.

3.2.2 Operations Performed by the Generation Rules

[26] The numbering of the records in the tree shows the sequence in which the rules apply to the basic tree.

Each rule, when applied to a node, can perform one or more of the following operations:

1. Assign a syntactic label to the node. For example, the “DECL” label will be assigned to the root node of a declarative sentence and NP will be assigned to the parent of a Noun terminal.

```
NP10 ——————— REC3063* "carpeta"
```

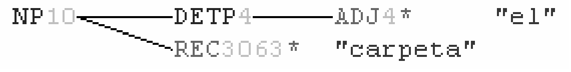
Figure 17: Assignation of NP label to record

2. Add morphosyntactic information to a node. This information can either come directly from the Logical Form, or from the dictionary, or else be computed by previous Generation rules. For example, finite verbs get number and person from the subject. Nominal nodes (such as `carpeta`, in Figure 18) get gender from the lexicon and number (generally) from the LF.

```
Lemma=“carpeta” +Fem +Sing
```

Figure 18: Add gender and number to the node “carpeta”

3. Expand a node by introducing one or more new nodes into the tree, based on the information present in the LF. In our example, the modal operator is turned into the full verb *poder*, and two functional nodes are inserted: the article *el* and the relative pronoun *que*.



```
NP10——————DETP4——————ADJ4*     "el"
    \—————REC3063*  "carpeta"
```

Figure 19: Insertion of determiner in NP node

4. Remove a node. For example, in the case of a pro-drop language, such as Spanish, the pronominal subject may be removed from the tree. Also, nodes that are copies of other nodes (cf. degraphing process in 3.2.1) may need to be removed as well.

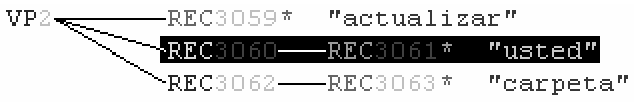


Figure 20: Removal of Subject "usted" from VP

5. Move a node by deleting it from position A and inserting it in position B. For example, the NP subject may be moved from the original post-verbal position to a preverbal position; or an adverb, such as *ahora,* in the example in Figure 15, may need to be moved to the front.

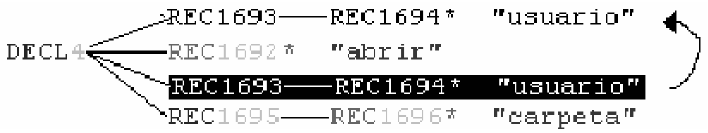

REC1693——REC1694* "usuario"
DECL4
REC1692* "abrir"
REC1693——REC1694* "usuario"
REC1695——REC1696* "carpeta"

Figure 21: Movement of subject to preverbal position

3.2.3 How Generation Rules Operate

The nodes in the generated tree are linked to each other by relations such as “head”, “parent” and “sibling” (i.e. Psmods). The entire tree (and the entire LF) is thus visible from any given node, at any given point, via these relations. When a rule applies to a node, the decisions made in that rule, in principle, can be based not just on features present at that node, but also on features present at any other node in the tree. This flexibility allows for different design approaches.

Having access to the whole structure would effectively eliminate the need for backtracking, which is necessary only if there are local ambiguities resulting from the absence of global information.

However, in the Spanish Generation grammar we have chosen to take consistently a local approach and the rules never look farther than two levels of depth at any given point. This implies that some of the decisions may need to be revisited in rules applying later, following a non-monotonic approach. In our opinion, this design option is intrinsically more modular, makes inspection and maintenance of the code easier, and suits better to the construction building nature of Generation .

Generation rules operate on a single tree. Rule application is deterministic and thus very efficient. If necessary, the tree can be traversed more than once, as is the case in the Generation grammar described here. There is a *feeding* relationship among the rules; that is, Generation rules may use information that has been computed by other Generation rules that have applied before.

To improve efficiency and to prevent a rule from applying at the wrong time or to the wrong structure, rules are classified into different groups according to the operation they perform. Each traversal of the tree activates a given group of rules. The order in which the different groups of rules apply depends on the *feeding* relations among them. Within a given group, rules apply sequentially following the order in which they are listed in the file.

Since the generated syntactic tree is built sequentially and acquires information in an *incremental* way, it is crucial to take into account

- the order in which the rules apply
- the order in which the nodes are visited

As noted above, the rules apply to the tree top-down, depth-first so that they explore a branch completely before starting with a new one. In the example, the numbering of the records (REC3046 to REC3063) shows the order of the nodes on which the rules will apply.

This will affect the design of the grammar. For example, the subject has not yet been computed when we are trying to figure out which values for person and number should be given to the verb. A subsequent pass that specifically checks agreement between verb and subject is thus needed. As way of illustration, in the simple example in Figure 15, which we reproduce here for convenience (Figure 22), the Spanish Generation component initially assigns syntactic types and functional roles and creates new syntactic nodes, using the information present in the LF.

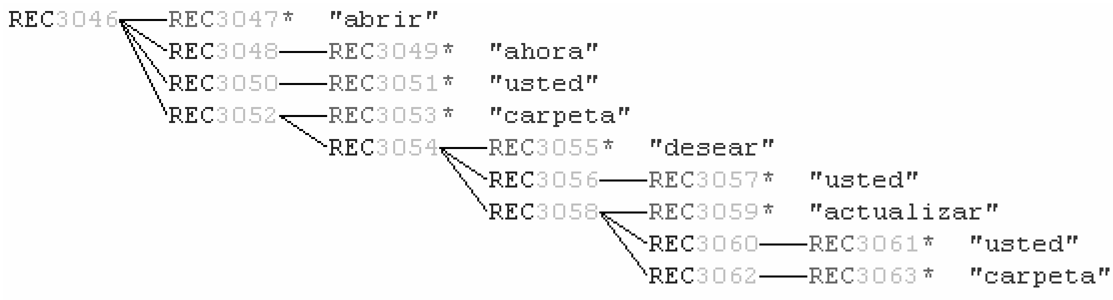


Figure 22: Basic tree for “Ahora puede abrir la carpeta que desea actualizar.”

Also in the first pass of the rules, the modal operator (a feature present in the verbal LF node), which does not appear as a full node in the tree, is turned into the full verb *poder*. The article *el* and the relative pronoun *que* are also inserted as new nodes. In a second pass, agreement is checked, both inside the NP and between subject and verb, assigning the appropriate person, number, and gender information to the terminal nodes. Finally, the temporal modifier is reordered. Figure 23 shows the resulting tree, whose leaves contain lemmas (e.g. *poder*) and inflectional information (e.g. *Pres, Sing, P3, Ind*).

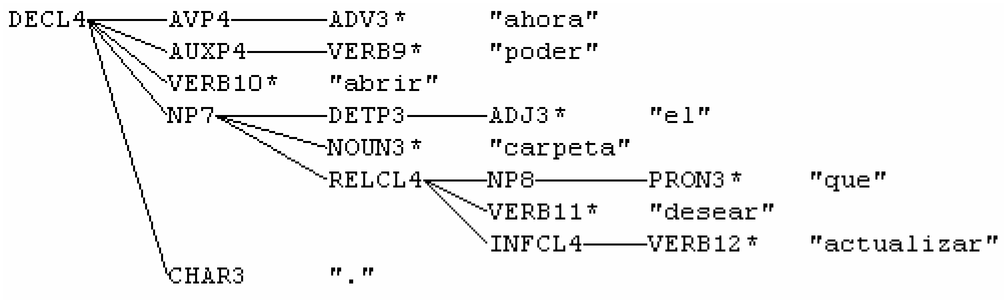


Figure 23: Generated syntactic tree

After all the syntactic Generation rules have applied, the morphological component comes into play. Morphological rules apply only on the leaf nodes of the tree (which carry all the necessary information) and build the proper inflected forms. Each node in the tree is a matrix of features where agreement information has already been assigned by the Generation rules. Morphological processing simply turns the feature matrices into inflected forms. For instance, in our example, the verb *poder* together with the features: Pres “present”, Sing “singular”, P3 “3rd person” and Ind “indicative” is spelled out as *puede*. The inflected form of each leaf node is then displayed to produce the surface string. This completes the Generation process.

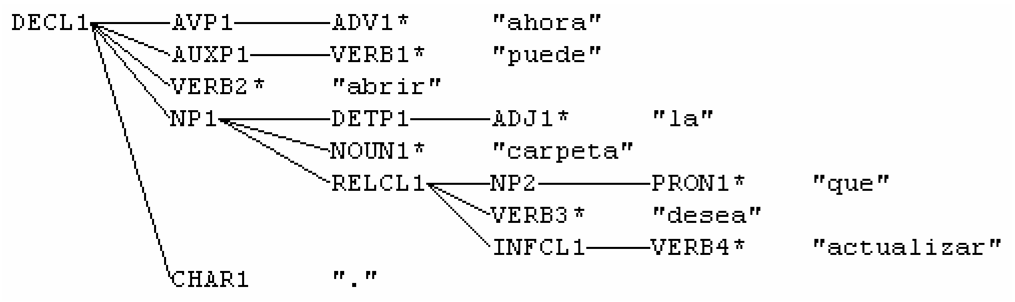


Figure 24: Generated tree with fully inflected leaves

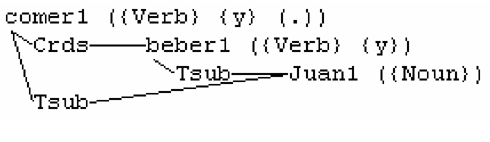
Each language Generation component operates following a similar pattern, with individual variations in the type of linguistic operations that are performed. [Aikawa et al., 2001] gives some examples extracted from the Chinese and Japanese components. In the rest of this chapter we will describe the strategies deployed by the Spanish Generation grammar.

3.2.4 Groups of Rules in the Spanish Generation Grammar

The Spanish Generation grammar has 6 different groups of rules. The rules in groups from 3 to 6 are sometimes called **Post-Generation rules** (cf. [Melero and Font-Llitjos, 2001]).

1. The first group contains only one rule that builds the **top coordinated node** (i.e. the node that is the root of the LF)[27]

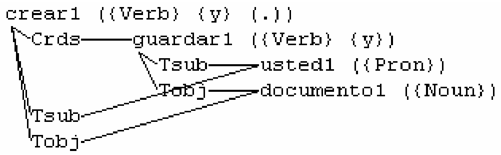
[27] The motivation to have a different coordination rule for the top node lies on an asymmetry in the LF treatment of coordinates; while one of them is the root of the LF, the rest hang from the Coordinates (Crds) list of this node, as illustrated by the example (*Juan come y bebe*)



This asymmetry does not happen in non-top coordination, where all the coordinates are elements of just one list, which may be any LF relation: Tsub, Tobj… (see section 2.5).

2. The second group contains the **main set of Generation rules**, which perform the following functions:
    a. Assignment of syntactic types, such as DECL, NP, RELCL…
    b. Computation of syntactic functions, such as Subject, Object....
    c. Initial calculation of morphosyntactic features, such as number, gender, tense, mood…
    d. Creation of coordinated nodes that are not top LF nodes.
    e. Creation of new syntactic nodes, such as determiners, clitics, modals, auxiliaries, prepositions and conjunctions.
    f. Initial reordering of complements: Subject and clitics in the verbal clause; quantifiers and adjectives within the NP; degree adverbs within the AJP.
    g. Deletion of subject if pro-drop conditions are met.
    h. Marking for deletion of gap-fillers in relative constructions.
    i. Insertion of final punctuation and certain commas.
3. The third group contains four **deletion rules**. These rules:
    a. Remove nodes marked for deletion by the rule from the second group that builds the relative clause. Delaying the removal of gap fillers to the next pass of rules, simplifies the treatment of phenomena such as agreement and obviation.
    b. Identify elements that are duplicated as the result of degraphing the LF (see Section 3.2.1); e.g. complements of a coordinated node[28]. Some are removed and some are marked for "reconstruction", either because they are an incomplete copy or because they have to be replaced by an anaphoric pronoun.
4. The fourth group contains two **reconstruction** rules. These rules:

[28] For example, the DO "documento" is duplicated in: "Cree y guarde el documento"



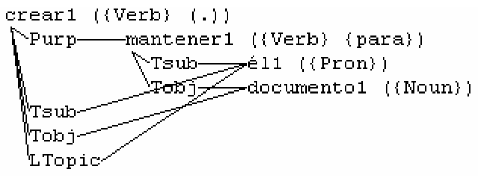
```
crear1 ({Verb} {y} (.))
 Crds———guardar1 ({Verb} {y})
             Tsub———usted1 ({Pron})
             Tobj———documento1 ({Noun})
 Tsub
 Tobj
```

    a. Reconstruct incomplete copies[29].
    b. Replace certain copies with an anaphoric pronoun[30].

5. The fifth group contains five **reordering** or **agreement** rules. These rules:
    a. Ensure agreement, at clausal level, between:
        i. finite verb and subject;
        ii. subject or object and predicative adjective;
        iii. doubled clitic with full complement.
    b. Ensure agreement within the NP between the noun and its modifiers.
    c. Check consistency of tense and mood values.
    d. Reorder noun modifiers.
    e. Reorder sentence constituents
    f. Assign default values for number, tense, gender, and aspect, if necessary.
6. The last group contains six rules that deal with **euphonic** issues and **punctuation**. These rules:
    a. Change the article to masculine when it precedes immediately feminine nouns beginning with an unstressed "*a*".
    b. Check apocopation of determiners or adjectives, such as *ninguno / ningún, grande / gran, etc.*
    c. Generates contracted forms of preposition + article (*del, al*).
    d. Ensure that the coordination has the right form of the conjunction (*y/e* or *o/u*)
    e. Generate parenthesis or quotes.
    f. Generate left-most interrogation sign.

[29] In the process of degraphing the LF and building the basic tree, to avoid getting into an infinite loop in certain situations (esp. antecedent of a relative clause) the modifiers of a copied node are added to just one of the copies, and not to the others, which, as a result, are incomplete.

[30] Anaphoric pronouns are created in certain cases of duplication (e.g. Direct Object in subordinated or relative clauses). For example, when generating a sentence from the following LF, the instance of documento in the Purpose clause will be replaced by a pronoun: *Él crea un documento para mantener**lo***.



```
crear1 ({Verb} {.})
 Purp——mantener1 ({Verb} {para})
          Tsub——él1 ({Pron})
          Tobj——documento1 ({Noun})
 Tsub
 Tobj
 LTopic
```

g. Contract clitics and preposition + article.
h. Check for double punctuation (e.g. final word is an abbreviation).

3.3 Linguistic Strategies in Generation

In Section 3.2.4 we have presented the sequence of the different operations performed by the core Generation and Post-Generation rules, in the same order that they take place. In this section we will look with more detail into some of the linguistic decisions that lie behind the main operations performed by the Generation grammar.

3.3.1 Assignment of Syntactic Types

Initially, non-terminal nodes are assigned a syntactic label based on the syntactic category of the lemma: (non-root) nodes with a verbal lemma are labeled as VP[31]; (non-definite) adjectival nodes are labeled as AJP; adverbial nodes are labeled AVP; and the rest are labeled NP (NP being the default part-of-speech).

In Spanish, the identification of the syntactic category is much less ambiguous than in other languages, such as English. In English most nouns can also be verbs, as, for instance, the words *file* and *record* in the following examples:

```
(2) File the record.
(3) Record the file.
```

Actually, in Spanish, we also find a lot of syntactic category ambiguity between inflected forms of the verb and nouns, such as, for example, feminine nouns and verbs in second and third person singular (*tapa, viola, mesas, cocinas*) or masculine nouns and first person singular forms of verbs (*vino, canto, cojo*). However, this ambiguity, while posing well-known problems to parsing, has little effect on Generation, which takes as input lemmatized, and not inflected, forms of words.

[31] Root verbal nodes may be labeled: DECL (declarative sentence), IMPR (imperative) or QUES (question).

In Spanish, most syntactic ambiguities between lemmas, are often cases of one word belonging to a specific morphological category functioning as a different syntactic category, and frequently involve adjectives, a hybrid part of speech that shares properties with both nouns and verbs. There are many instances of adjectives that can function as nouns: *pequeña, adelantado, privado, rápidos, etc.*; some can also function as adverbs: *lento, suave, etc.*; and many of them, such as *elevado* or *educado* are morphological past participles (see Section 6.2.1.)[32].

In reason of this relative simplicity, the assignment of syntactic category in Spanish relies mostly on the lexicon look-up of the lemma, a much more straightforward strategy than the one required by the English grammar, for example.

In the first pass, all Noun headed phrases are assigned NP as all Verb headed phrases are assigned VP. In subsequent passes, after all the core rules have applied, some labels need to be renamed. In this way, certain NPs become PPs (e.g. human headed Direct Objects), and most VPs become something else, such as RELCL, SUBCL, INFCL, etc.

3.3.2 Computation of Syntactic Functions

We need to compute syntactic functions (SF) in order to generate a correct string. Important surface phenomena, such as agreement and reordering, depend on the value for syntactic function: for example, the subject needs to agree with the verb; indirect object may need to be duplicated by a clitic; direct objects impose certain reordering conditions; etc.

Clitics are a special kind of NPs that use SF information to compute Case. SF is also used to generate preposition *a* marking of Indirect Objects and human Direct Objects.

Syntactic function is assigned based on the semantic role as well as on other types of information, such as passivity or type of verb. The table below shows the set of syntactic functions used by the Spanish Generation grammar and their Logical Form correspondences:

[32] The adjective/past participle ambiguity is more a problem of how the syntactic types are described in the system than a genuine linguistic ambiguity.

SF	LF	Examples
Subject	Tsub of an active clause Tobj of a passive or reflexive passive clause[33]	El **usuario** cierra el programa. El **equipo** ha sido afectado por el virus. Deben instalarse las **funciones**.
Direct Object	Tobj of an active, non-copulative clause	Vuelva a enviar el **fax**.
Indirect Object	Tind	A todos los **archivos** se les asigna la fecha actual.
Predicative Adjective	Adjective Tobj[34] of a copulative clause Lcmp adjective	La tecla está **presionada**. Mantenga **presionada** la tecla.
Predicative Nominal	Tobj noun or pronoun of a copulative clause Lcmp noun	El sistema es **Windows XP**. Se denominan **programas** residentes.
Predicative Complement	Clausal Tobj	La instalación le obliga a **reiniciar**.

The rule that computes syntactic functions is among the first ones to apply. Once the functions have been assigned to the nodes, other rules performing specific actions for the different functions are triggered.

3.3.3 Computation of Morphosyntactic Features

Morphological information, which is required to generate the actual words, comes from one of these three sources:

1) Semantic information coming from the Logical Form structure.

2) Lexical information encoded in the Spanish dictionary

3) Syntactic information already computed by the Generation rules

Morphological bits are computed in the main set of rules but may be reviewed by the Post-Generation rules, as for instance when agreement with the subject is checked. In case of necessity, defaults are applied for the different features.

[33] The rule that assigns syntactic functions also creates a se-pronoun in the case of reflexive passive constructions.

[34] It can also be a verb in participial form.

In Spanish, the part of speech that requires the richest morphological information is the verb. In Table 2, we show the main sources of information for the different types of verbal information plus the default value, if any.

Type of information	Sources of information			Default value
	LF	Computed	Lexicon	
Tense	X	X[35]		Present
Mood		X		Indicative
Aspect[36]	X			Perfect
Voice	X			Active
Person		X		3rd
Number		X[37]		Sing
Subcat[38]			X	--
Type[39]			X	--
Reflexivity[40]	X		X	--

Table 2: Verbal features: sources of information and default value

Determining a default value is necessary in every calculation performed by the Generation grammar, as a measure of robustness. As we will discuss later, particularly in Chapter 4, we cannot rely on the integrity of the input. For example, when translating from English, mood information may be absent or be irrelevant.

3.3.4 Generation of Noun Determiners

[35] The main source of the information about tense is the incoming LF, but this value might be changed, especially in the subordinated clauses, due to phenomena such as *consecutio temporum*.

[36] Perfect/imperfect

[37] The number and person of the head verb are taken from the subject.

[38] The subcategorization features include the ones from LDOCE, adapted to Spanish: I0, I1, I2, I3, I4, I5, D1, D5, D6, L1, L4, L5, L6, L7, L9, T1, T1io, T2, T3, T4, T5, T6, V4, X1, X7, X9 plus information about Control and mood of the subordinate clause (e.g. TakeSubjunctive)

[39] Verbs are partially classified into: Aspectual (*empezar*, *continuar*…), Weather (*llover*, *nevar*…), Movement (*ir*, *traer*…), Speech act (*decir*, *pensar*…) and Psychological (*gustar*, *aburrir*…).

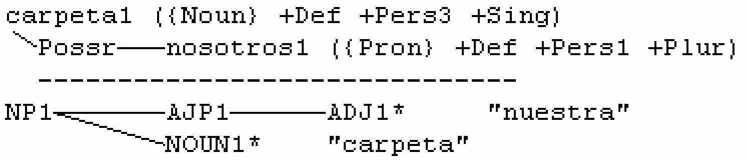
[40] Constructions with "se" are marked as Impersonal, Reflexive passive or Reflexive sense (=pronominal verb) in the LF. Pronominal verbs (*arrepentir*) or verbs with a pronominal reading (c*ansar*) are marked as Reflexive in the lexicon.

Noun determiners include articles (definite and indefinite), demonstratives, possessives and quantifiers. These four types are represented in different ways in LF. While possessive and quantifiers constitute a semantic relation in the LF (thus, are full-fledged nodes), articles and demonstratives are expressed as linguistic features (bits) and do not appear as actual nodes in Logical Form (see Section 2.5). Both types of representation aim at neutralizing diverging surface realizations in the different languages.

Neutralization of syntactic phenomena in Logical Form allows for a more homogeneous representation across languages, since superficially distinct constructions in two languages frequently collapse onto similar or identical LF representations. This shared representation greatly simplifies the task of aligning source and target LF segments in Transfer. As opposed to other Example-Based Machine Translation systems that extract and use examples represented as linear patterns of varying complexity, thanks to neutralization of surface constructions, MSR-MT is able to leverage the linguistic generality to enable broad coverage and to overcome some of the limitations on locality of context characteristic of data-driven approaches [Richardson et al., 2001b].

On the other side of the court, the Generation grammar has to compensate for the neutralization performed in LF. It needs to recreate all the missing nodes by using the information in form of features or relations provided by the input LF.

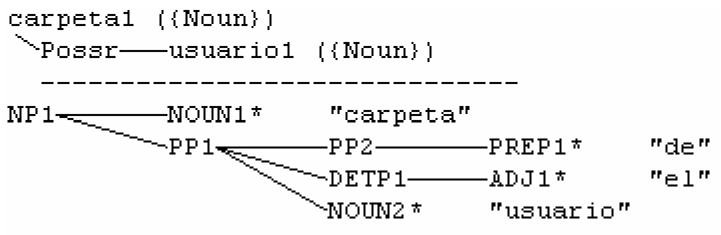
- Possessives are expressed as the relation *Possr*. Whenever the content of the relation is a personal pronoun, as in the example in Figure 25, a possessive adjective (in this case, *nuestro*) is generated. Agreement between noun and possessive will also be taken care of.



```
carpeta1 ({Noun} +Def +Pers3 +Sing)
 `-Possr---nosotros1 ({Pron} +Def +Pers1 +Plur)
  ------------------------------
NP1-------AJP1------ADJ1*     "nuestra"
   `------NOUN1*    "carpeta"
```

Figure 25: LF and tree corresponding to *nuestra carpeta*

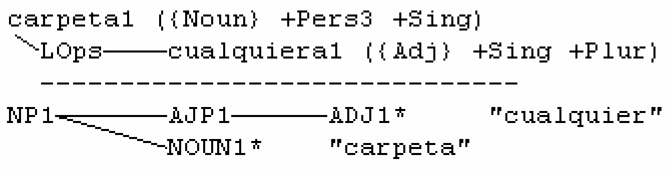
Contrarily, if the possessor is a noun, as *usuario* in Figure 26, then a modifying PP with preposition *de,* with the possessor as head, is created



```
carpeta1 ({Noun})
 `-Possr——usuario1 ({Noun})
  ------------------------------
NP1————————NOUN1*    "carpeta"
    `——————PP1———————PP2—————————PREP1*    "de"
               `—————DETP1———————ADJ1*     "el"
               `—————NOUN2*      "usuario"
```

Figure 26: LF and tree corresponding to *carpeta del usuario*

- Quantifiers (such as *todos*, *muchos*, *cuatro*...) and other indefinite determiners (such as *otro*, *ninguno*, *cualquier*…) are subsumed by the relation LOps[41].



```
carpeta1 ({Noun} +Pers3 +Sing)
 `-LOps————cualquiera1 ({Adj} +Sing +Plur)
  ------------------------------
NP1————————AJP1——————ADJ1*     "cualquier"
    `——————NOUN1*    "carpeta"
```

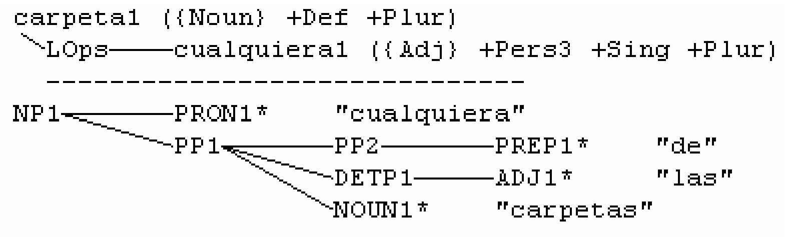
Figure 27: LF corresponding to *cualquier carpeta*

Based on the definiteness and number of the head noun, two kinds of structure are built in Generation. The simplest case is illustrated by Figure 27: if the head noun (*carpeta*) is **singular** and **not definite**, Generation simply needs to reorder the operator after the noun and make them both agree in gender and number.

If the head noun is **definite**, i.e. marked +Def[42], then Generation needs to build an NP where the syntactic head is a pronoun with the lexical value of the quantifier, as shown in Figure 28.

[41] For "Logical Operators"

[42] When generating quantifiers, definiteness of the noun takes precedence over number. If the noun is not definite, then the noun takes singular by default and vice versa.

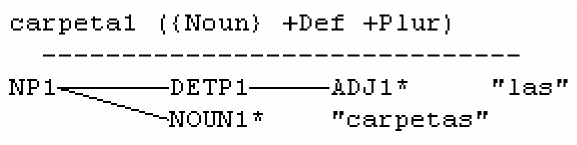


```
carpeta1 ({Noun} +Def +Plur)
 \LOps───cualquiera1 ({Adj} +Pers3 +Sing +Plur)
  ------------------------------
NP1──────PRON1*    "cualquiera"
   \─────PP1───────PP2───────PREP1*    "de"
            \\─────DETP1─────ADJ1*     "las"
              \────NOUN1*    "carpetas"
```

Figure 28: LF and tree for *cualquiera de las carpetas*

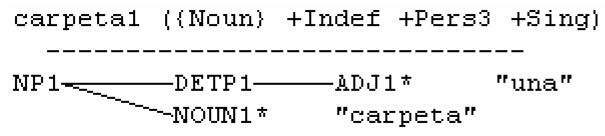
- Definite article, indefinite article, demonstratives are all represented by features in the Logical Form.

In the figures that follow, we illustrate the mapping from the featurized LF to the structural representation created by the Generation rules.



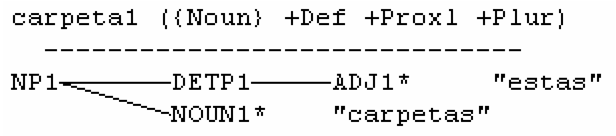
```
carpeta1 ({Noun} +Def +Plur)
  ------------------------------
NP1──────DETP1─────ADJ1*     "las"
   \─────NOUN1*    "carpetas"
```

Figure 29: LF and tree for *las carpetas*



```
carpeta1 ({Noun} +Indef +Pers3 +Sing)
  ------------------------------
NP1──────DETP1─────ADJ1*     "una"
   \─────NOUN1*    "carpeta"
```

Figure 30: LF and tree for *una carpeta*



```
carpeta1 ({Noun} +Def +Prox1 +Plur)
  ------------------------------
NP1──────DETP1─────ADJ1*     "estas"
   \─────NOUN1*    "carpetas"
```

Figure 31: LF and tree for *estas carpetas*

Three bits are used to classify demonstratives in Logical Form. Most of the languages use only two: Proxl implies proximity to the speaker and Distl implies distance. Spanish uses a third bit: Medl to encode *ese*.

3.3.5 Generation of Modals and Verbal Auxiliaries

Logical Form is a predicate-argument structure, therefore, verbs that function as auxiliaries, modals or aspectual operators are eliminated from the main structure, while the information they carry is kept some other way.

Modals and aspectual operators are stored in a relation precisely called Modals. Sentence in example (4) contains a modal (*deber*) and an aspectual operator (*ir a*).

(4) El sistema debe avisarle de que va a reiniciarse.

Figure 32 shows the LF for this sentence. Note that values in the Modals relation are the exact lexical values of the original modal and aspectual verbs (literally *deber* and *ir*). Contrary to other MT systems (Eurotra [Durand et al, 1991], for instance), no semantic classification of modal or aspectual values is attempted[43].

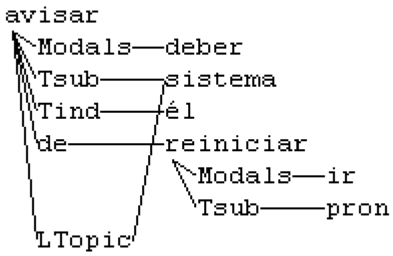


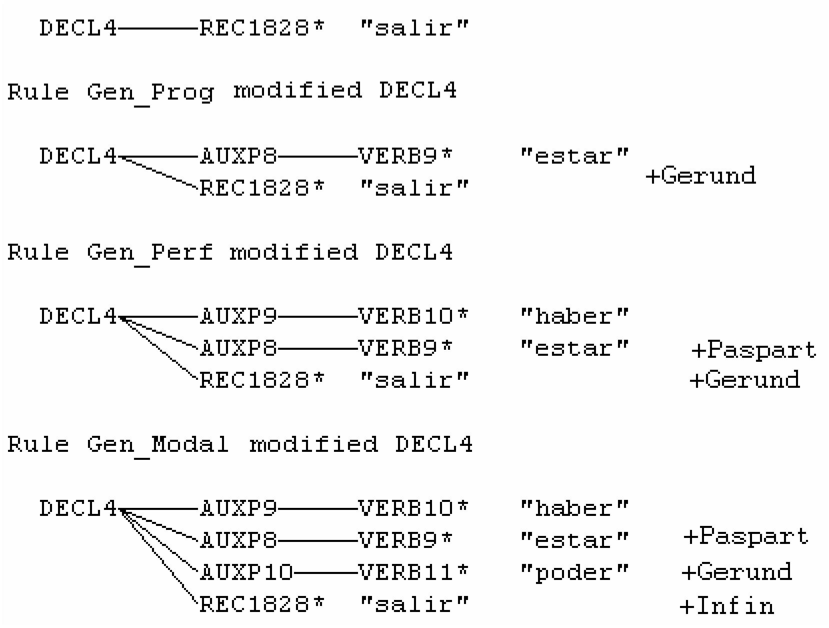
Figure 32: LF with Modals relation

On the other hand, temporal auxiliaries, such as *haber* and *estar* disappear from the structure and are encoded in form of features: *Perf*(*ect*) and *Progr*(*essive*). Generation needs to turn them back into actual nodes, which carry morphosyntactic information and which act as true syntactic heads of the verbal group.

There is a rule that generates the progressive auxiliary *estar*, another rule generates the perfective auxiliary *haber* and a third rule, that applies just after the other two have applied, generates the modal auxiliary. Figure 33 shows the sequence of application of

[43] This solution helps get rid of language-particular aspects of the modal structure, while leaving the translation of the specific modal and aspectual values to machine learned transfer.

the three rules involved in the Generation of the complex VP: *ha estado pudiendo salir*; it also shows the intermediate states of the structure.



```
  DECL4———————REC1828*  "salir"

Rule Gen_Prog modified DECL4

  DECL4———————AUXP8———————VERB9*     "estar"
        \—————REC1828*  "salir"              +Gerund

Rule Gen_Perf modified DECL4

  DECL4———————AUXP9———————VERB10*    "haber"
       |\—————AUXP8———————VERB9*     "estar"    +Paspart
        \—————REC1828*  "salir"                 +Gerund

Rule Gen_Modal modified DECL4

  DECL4———————AUXP9———————VERB10*    "haber"
       |\—————AUXP8———————VERB9*     "estar"   +Paspart
       |\—————AUXP10——————VERB11*    "poder"   +Gerund
        \—————REC1828*  "salir"                +Infin
```

Figure 33: Sequence of application of the rules to generate: "ha estado pudiendo salir"[44]

Each type of auxiliary dictates the inflection of the verb (auxiliary or not) following immediately after it. Thus, in the example in Figure 33, *haber* requires the value *Pastpart* on the next verb, *estar* requires to be followed by a Gerund and finally the modal *poder* needs an Infinitive. Each time that this happens, the auxiliary inherits the previous inflection of the verb whose inflection it is modifying. Note that the final state of each rule is a consistent grammatical structure:

1) sale (+Pres) =>

2) está (+Pres) saliendo (+Gerund) =>

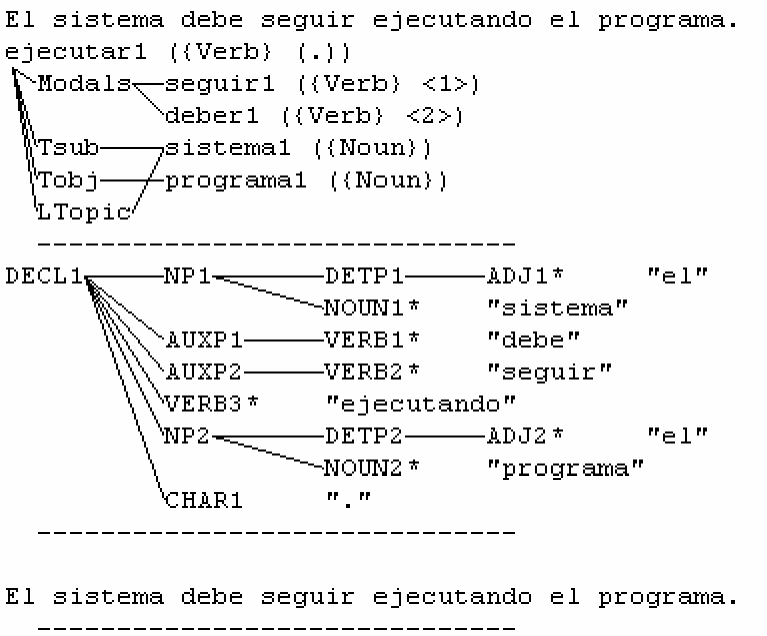
3) ha (+Pres) estado (+Pastpart) saliendo (+Gerund) =>

4) ha (+Pres) estado (+Pastpart) pudiendo (+Gerund) salir (+Infin).[45]

[44] VPs, in NLPwin, are flat trees, with all the auxiliary nodes gathered at the same level as the head.

[45] The distinction between “ha estado pudiendo salir” and “ha podido estar saliendo” is not translatable to other languages, such as English and French. The only way to differentiate them in Spanish LF is through the bits Pastpart and Prespart in the modal. Generation bases the placement of the syntactic modal node on this information.

Generation of more than one modal (or aspectual operator), such as in the example in Figure 34 is achieved by simple iteration of the Modals rule.

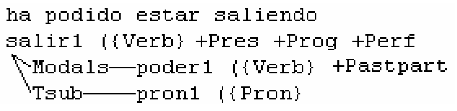


```
El sistema debe seguir ejecutando el programa.
ejecutar1 ({Verb} (.))
 Modals—seguir1 ({Verb} <1>)
          deber1 ({Verb} <2>)
  Tsub——sistema1 ({Noun})
  Tobj——programa1 ({Noun})
  LTopic
  ------------------------------
DECL1——NP1——DETP1——ADJ1*      "el"
              NOUN1*    "sistema"
         AUXP1——VERB1*    "debe"
         AUXP2——VERB2*    "seguir"
         VERB3*   "ejecutando"
         NP2——DETP2——ADJ2*      "el"
              NOUN2*    "programa"
         CHAR1    "."
  ------------------------------

El sistema debe seguir ejecutando el programa.
  ------------------------------
```

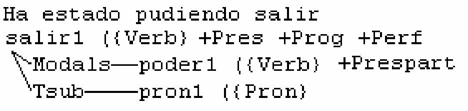
Figure 34: Regeneration of the multiple operator sequence "*debe seguir*"

3.3.6 Generation of Clitics

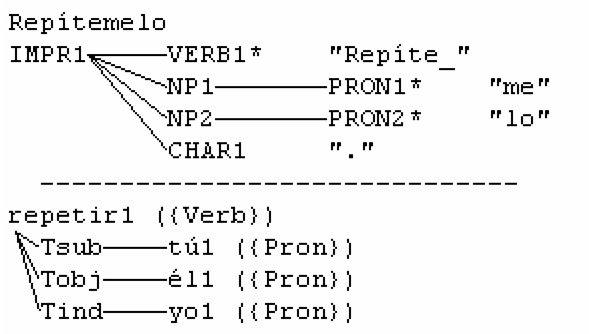
Clitics are personal pronouns in an oblique (non-nominative) case. In modern Spanish, clitics appear attached to the verb when this takes a non-finite form (infinitive, gerund), and preceding the verb as independent particles in all other cases. In Logical Form, clitics occupy full argument slots, functioning as direct or indirect object.



```
ha podido estar saliendo
salir1 ({Verb} +Pres +Prog +Perf
 Modals—poder1 ({Verb} +Pastpart
 Tsub——pron1 ({Pron}
```



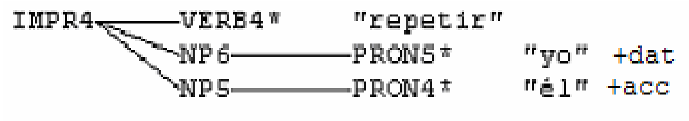
```
Ha estado pudiendo salir
salir1 ({Verb} +Pres +Prog +Perf
 Modals—poder1 ({Verb} +Prespart
 Tsub——pron1 ({Pron}
```



```
Repitemelo
IMPR1───VERB1*    "Repite_"
     \──NP1──────PRON1*    "me"
     \──NP2──────PRON2*    "lo"
     \──CHAR1    "."
  ------------------------------
repetir1 ({Verb})
 \─Tsub───tú1 ({Pron})
 \─Tobj───él1 ({Pron})
 \─Tind───yo1 ({Pron})
```

Figure 35: Clitics occupy argument positions in the LF (*repítemelo*)

Until the last stage of the Generation process, clitics are kept as full nodes of the tree, and rules are applied accordingly.



```
IMPR4───VERB4*    "repetir"
     \──NP6──────PRON5*    "yo"  +dat
     \──NP5──────PRON4*    "él"  +acc
```

Figure 36: Clitics are ordered nodes in the generated tree (*repítemelo*)

The rule that deals with Direct Objects applies first. The flowchart in Figure 37 shows the conditions that are checked by the rule for Direct Objects and the actions that are taken according to these conditions. The rule for Indirect Objects is shown next (Figure 38)

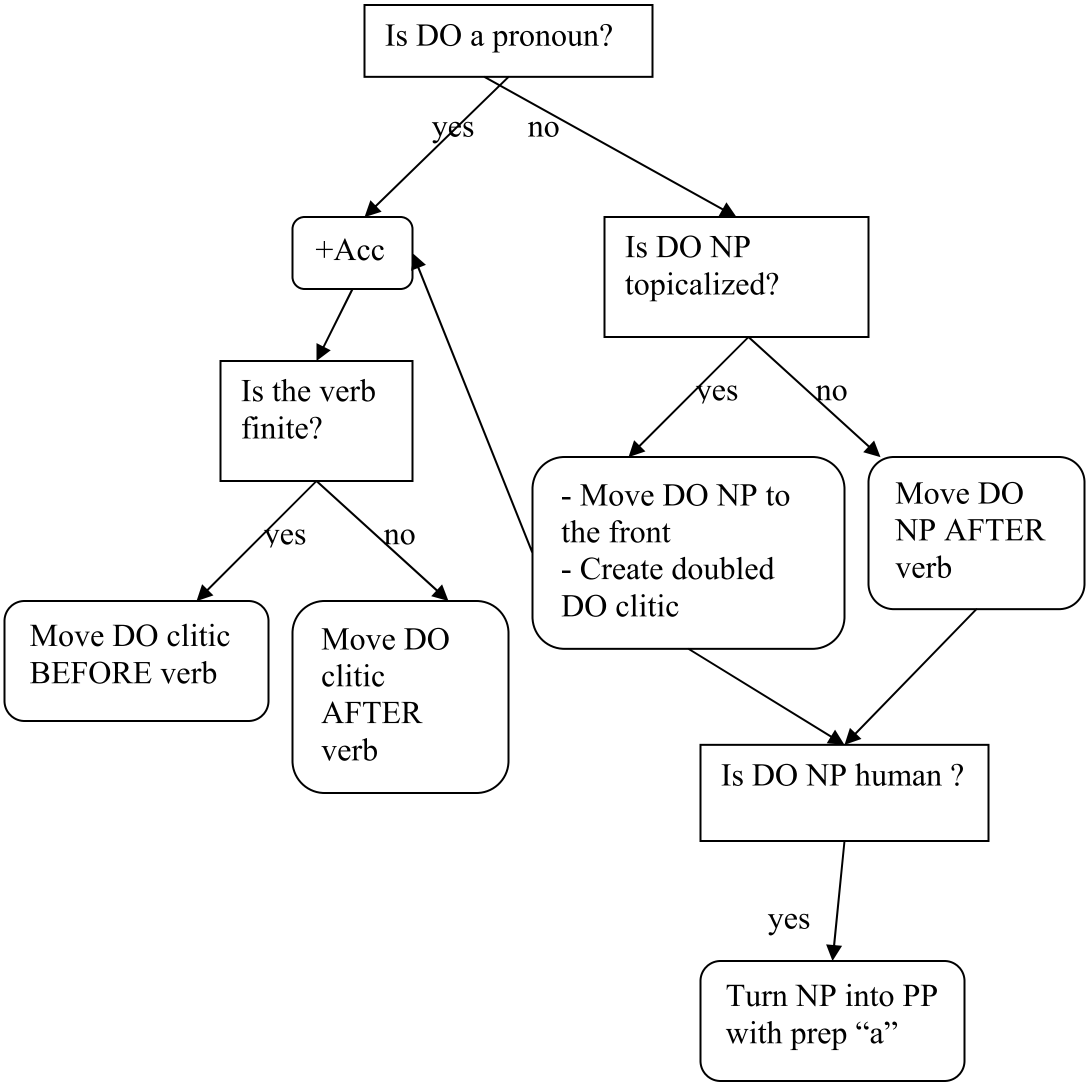

Is DO a pronoun?
yes
no
+Acc
Is DO NP topicalized?
Is the verb finite?
yes
no
- Move DO NP to the front
- Create doubled DO clitic
Move DO NP AFTER verb
yes
no
Move DO clitic BEFORE verb
Move DO clitic AFTER verb
Is DO NP human ?
yes
Turn NP into PP with prep "a"

Figure 37: Logical flow of the rule for Direct Objects

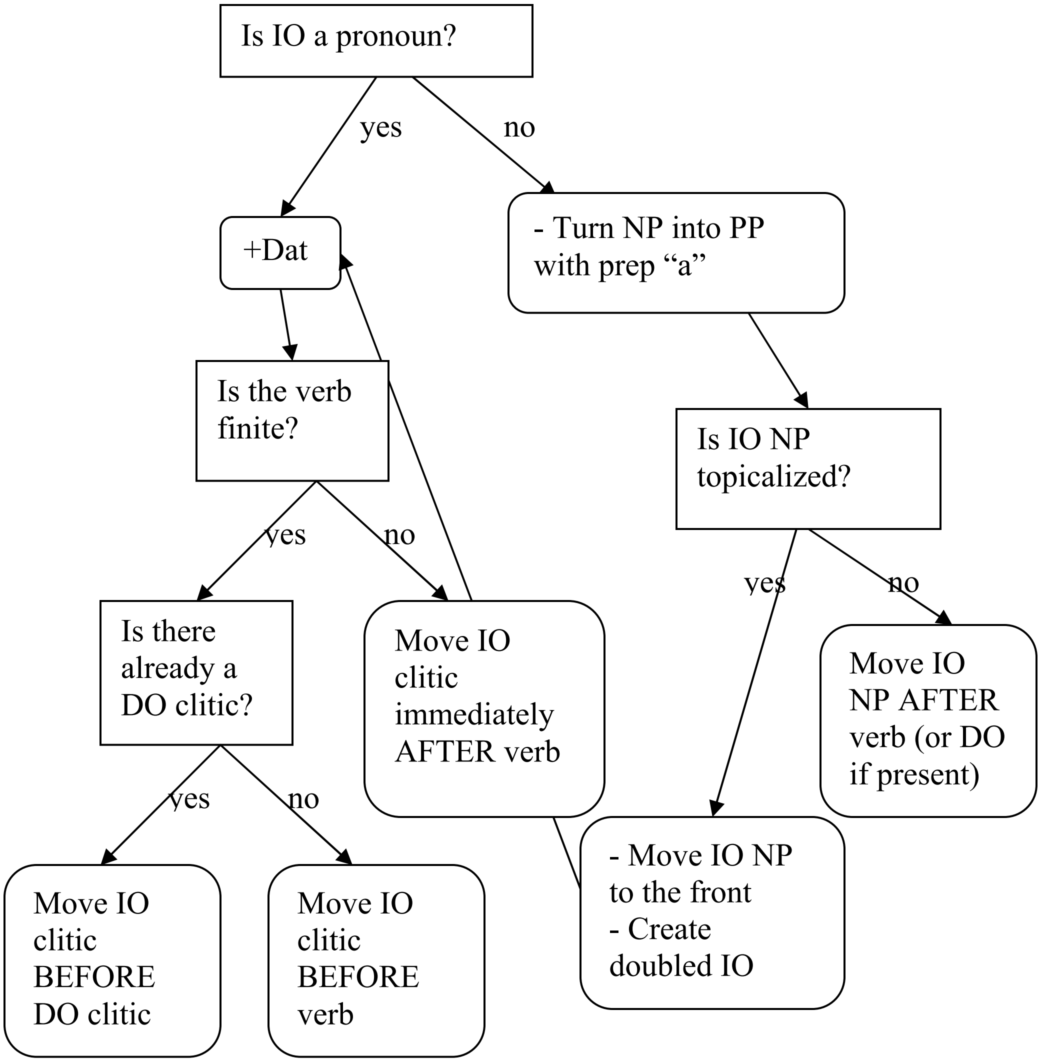

Is IO a pronoun?
yes
no
+Dat
- Turn NP into PP with prep "a"
Is the verb finite?
Is IO NP topicalized?
yes
no
yes
no
Is there already a DO clitic?
Move IO clitic immediately AFTER verb
Move IO NP AFTER verb (or DO if present)
yes
no
Move IO clitic BEFORE DO clitic
Move IO clitic BEFORE verb
- Move IO NP to the front
- Create doubled IO

Figure 38: Logical flow of the rule for Indirect Objects

Both rules add Case information to the clitic nodes and order them with respect to the verb and the rest of constituents of the sentence.

The rules take also into account the surface syntactic phenomenon known as **clitic doubling**. In Spanish, as in other romance languages, when the full DO or IO argument is topicalized, a clitic is added which has the same syntactic function, as exemplified by the following sentences.

(5) Al primer usuario que se registra, el sistema **lo** coloca en el grupo de administradores.

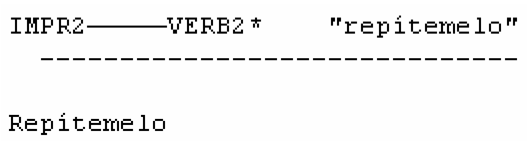
(6) Al usuario **le** conviene actualizar la aplicación[46].

(7) Al usuario, el programa **le** presenta diversas alternativas.

The clitic resulting from doubling is then dealt with in the same way as the regular DO or IO clitic[47].

Later, in Post-Generation, coocurrence of a clitic object with a third person indirect object clitic is checked, and if necessary the latter takes a reflexive form, so that [*dar*+*le*+*lo*] becomes [*dar*+*se*+*lo*].

The last rule of the grammar is the rule that contract clitics with verbs in non-finite form.



```
IMPR2———————VERB2*    "repitemelo"
  ------------------------------

Repitemelo
```

Figure 39: Clitics appear contracted to the inflected verb in the generated string

3.3.7 Word Order

Generating the surface string in a relatively free word order language such as Spanish is harder than doing it for languages with stricter rules. For one and the same Logical Form, often several alternatives are possible. Choosing one may be a difficult task. Whenever the language offers a variety of solutions for one situation, the Generation

[46] *Convenir*, as many other psychological verbs, subcategorizing for an infinitive clause as subject and an indirect object, routinely place the indirect object in a preverbal topicalized position and take a doubled clitic as well.

[47] Standard Spanish seems to favor IO clitic doubling even when the full argument is not topicalized, such as in: *El cliente **le** pidió al técnico que le solucionara el problema*. However, an informal survey of this phenomenon in technical corpora shows a less marked inclination to do so. Thus, between these two sentences:

a) El sistema proporciona al cliente soluciones eficaces.
b) El sistema le proporciona al cliente soluciones eficaces.

The first one seems to be preferable. Consequently, our grammar has opted to double the clitic only in cases of topicalization.

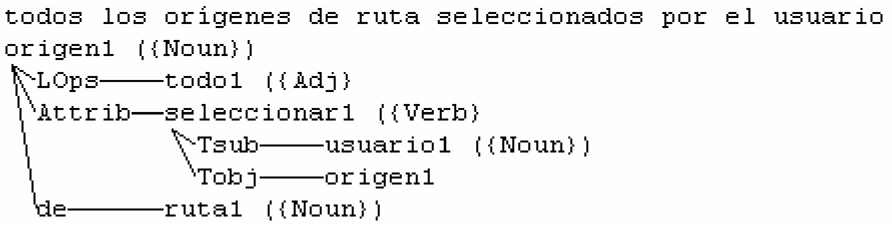
grammar has to avoid being too deterministic. Otherwise the resulting text, although correct, may sound artificial.

In the main set of rules, basic ordering decisions are taken, such as position of arguments or relative order of noun determiners. However, most of the heuristics is done in Post-Generation, once all the components are in place and all the elements that need to be removed are not there any more.

3.3.7.1 Word Order within the NP

Noun modifiers are represented in LF as a flat list without scope or order information from the source language. Since LF nodes are always terminal, representation of scope of logical operators and modifiers is not straightforward, if not impossible altogether[48].

For this reason, the task of deciding on the surface linear order of the generated string relies heavily on specific Generation strategies based only on monolingual information. These strategies are of a heuristic nature and, thus, need to be verified by means of extensive testing.



```
todos los origenes de ruta seleccionados por el usuario
origen1 ({Noun})
 \LOps────todo1 ({Adj})
  \Attrib──seleccionar1 ({Verb})
  |          \Tsub────usuario1 ({Noun})
  |           \Tobj────origen1
   \de───────ruta1 ({Noun})
```

Figure 40: Modifiers of *origen* do not reflect scope in LF

The rule that reorders noun modifiers to the right of the head has two parts. In the first part, a number between 1 and 9 is assigned to every modifier, based on a heuristics that has been hand tuned by testing thousands of sentences. In the second part of the rule, the actual reordering of modifiers takes place.

[48] This was one of the main motivations for the design of yet another level of representation, known as Language-Neutral Syntax or LNS. LNS is an annotated tree but constituents are not ordered and immediate constituents of a given node are identified by labeled arcs indicating a semantically motivated relation to the parent node. LNS was explored for a while as a possible alternative to LF, but was finally dropped.

The heuristics followed by the first part of the rule is based on an ad-hoc classification of modifiers, used for this purpose. This classification involves category, lenght (i.e. number of words), type of noun, bound preposition, etc. Table 3 shows the order assigned to each type of modifier.

Type of modifier	Order
Adjective and past participles without modifiers (e.g. *operativo*, *abierto*)	1
PP-de common noun, not determined, without modifiers (e.g. *de archivos*)	2
PP-de common noun, not determined, with modifiers (e.g. *de acceso directo*)	3
NP without modifiers (e.g. *Windows, archivo*)	4
PP-de proper noun not determined (e.g. *de Office*)	5
Adjective with modifiers	6
Non coordinated bound PP	7
Pastparts with modifiers; NPs with modifiers; PPs in general	8
The rest: relative and subordinated clauses, etc	9

Table 3: Order of modifiers within the NP

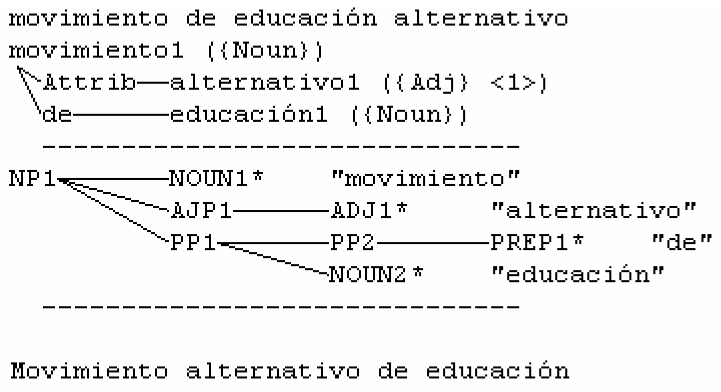
The second part of the rule is a simple loop that checks for each modifier the actual position in the list of modifiers, and the assigned order. If there is another modifier in the list such that its position is smaller and its number of order is bigger than the values of the current modifier[49], then the current modifier is moved to the front of the first modifier that fulfills the condition. This part of the rule does not need to change even if the heuristic assignment of order, or part of it, is modified during the fine-tuning process.

This algorithm has been tested with good results over thousands of Spanish sentences. One problematic area is the combination of Adjective and PP-de. The default order assigned by the rule is [Adj + PP-de] but it is not infrequent to find the reverse order [PP-de + Adj], provided that the adjective does not agree with the NP within the PP.

[49] Pos(mod)<Pos(rec) and Order(mod)>Order(rec);

In the example in Figure 41: [*movimiento*][*de educación*][*alternativo*], gender agreement tells us that *alternativo* unambigually modifies *movimiento*, and not *educación*; therefore, both orderings (*movimiento de educación alternativo* and *movimiento alternativo de educación*) are possible.

Obviously, if instead of *educación*, we had a masculine noun (such as *aprendizaje*, for example), then the only possible ordering would be: [*movimiento*][*alternativo*][*de aprendizaje*], because the other ordering would have a different meaning (and in structural terms, the adjective would belong to a different NP); namely, [*movimiento*] [*de aprendizaje alternativo*].



```
movimiento de educación alternativo
movimiento1 ({Noun})
\Attrib——alternativo1 ({Adj} <1>)
 \de———————educación1 ({Noun})
  ------------------------------
NP1———————NOUN1*    "movimiento"
   \———AJP1———————ADJ1*     "alternativo"
    \—PP1———————PP2———————PREP1*    "de"
         \——————NOUN2*     "educación"
  ------------------------------

Movimiento alternativo de educación
```

Figure 41: Adjective-PP flip-flop

In cases like Figure 41, both are usually correct but the default order is slightly preferred because it is less ambiguous. However, in other examples, which are structurally identical, one of the alternatives is clearly better. This is what happens in example (8) compared to (9), or example (10) and its more awkward counterpart in (11).

```
(8)  la página de códigos especificada
(9)  ?la página especificada de códigos
(10) orígenes de rutas seleccionados
(11) ?orígenes seleccionados de rutas
```

We find that collocations play an important role in surface ordering of NP constituents. Collocations are words that have a tendency to appear together in texts and that express a particular concept or term.

```
(12) [sistema operativo] de usuario
(13) [Disco duro] PCI
```

The nouns and adjectives involved in the collocations shown in examples (12) and (13) have a tendency to go together that can be proved on statistical grounds.

As a way of illustration, compare (13) with the NP in example (14) which has the same type of constituents [N + ProperN + Adj] but which require a different surface ordering.

```
(14) Dirección URL correcta
```

The fact that [*disco*+*duro*] is a collocation while [*dirección*+*correcta*] is not, gives us the clue to the correct surface ordering of these NPs.

It seems apparent that collocational information (whether statistically learned or hand-coded) would contribute very positively to our ordering heuristics[50].

3.3.7.2 Word Order at Sentence Level

Lack of a true scope treatment involves similar problems at the level of the sentence, especially in what concerns modifiers. The rule that reorders sentence constituents is more complex than the one that reorders elements within the NP because it takes care of several order-related problems:

- Extraposition
- Clitic relative ordering (i.e. relative ordering among them)
- Order of negation
- Order of relative pronoun in relative clause
- Relative ordering of constituents following the verb

[50] Collocation information is not provided by the NLPWin system.

The reordering of constituents following the verb is performed after all movements between 'Prmods' and 'Psmods' have already been sorted out[51]. This task is comparable with what is done for the NP, seen in Section 3.3.7.1. The coding strategy is very similar; it also consists of two parts. In the first part, again a number is assigned to each modifier, and in the second part of the rule, the actual reordering of modifiers takes place.

Again, this assignment is based on a heuristics that has been extensively tested on thousands of sentences with very good results.

Type of modifier	Order
Clitic (e.g. *lo*)	1
Degree adverb (e.g. *más*)	2
Verb argument (NP, AP or clause) without internal modifiers (e.g. *el programa*; *a la competencia*)	3
Other adverbs (e.g. *rápidamente, ayer*)	4
Verb argument (NP, AP or clause) with internal modifiers (e.g. *a la competencia interior*)	5
Time (non-adverb) modifiers (e.g. *esta mañana*)	6
Bound PP (e.g. *(integrar) en el sistema*)	7
PPs in general (e.g. *con gran eficacia*)	8
Infinitival subclauses (e.g. *para comunicarse*)	9
Finite subclauses (e.g. *si el sistema está apagado*)	10

3.3.8 Agreement Checking

As seen in Section 3.2.4, agreement is checked in the Post-Generation block of rules. Agreement needs to be verified at different levels and between different elements of the generated string. Agreement within the NP and agreement between the verb and the subject are the two main cases of agreement in Spanish.

[51] That is, elements that should precede the verb are moved to the front.

As a general rule, the subject of the sentence needs to agree in person and number with the finite verb of that sentence. Subject pronouns are able to inflect for person and number. Nouns generally inflect only for number and usually agree with the 3rd person of the verb. Verb-headed constituents (such as complement clauses and infinitival verbs) may also function as subjects, but they always agree with the 3rd person of the verb.

The general rule that states that the coordinated subject always agrees in plural has several exceptions, as described in [Melero, 2001]. For example, if all the members of the coordination are singular and the last one is not determined (and is not a proper noun), both agreements are possible.

(15) Se prohíbe la carga y descarga de mercancías.

Also, when the coordinated NP follows the verb, it may agree with the first member of the coordination, instead of agreeing with the whole coordinated term:

(16) Entre sus amigos destacaba el gran arquitecto Filippo Brunelleschi y el escultor Donatello.

Contrarily to Analysis, which must be ready to accept all possible inputs, Generation has to choose in a deterministic fashion between two or more equally acceptable solutions. The rule of thumb tends to favor more general solutions; therefore coordinated subjects are made to agree in plural with the verb, in all these cases.

On the other hand, sometimes only singular agreement of the coordinated subject is possible, as in coordination of clauses (example (17)) and in coordination (or disjunction) of nouns, if the coordinated nouns are not preceded by a determiner (with the exception of the first coordinate) and are not proper nouns (examples (18) and (19)).

(17) Interesa que el sistema sea compatible y que sea adaptable.

(18) Se prohíbe la carga o descarga de mercancías.

(19) El matemático y físico británico Isaac Newton describió la luz como una emisión de partículas.

Apart from Subject-Verb agreement, the following agreement conditions need to be checked in the Generation rules:

- Number and gender agreement between Subject and Predicative Adjective:

(20) El sistema permanece apagado.

- Number and gender agreement between Direct Object and Predicative Adjective:

(21) Deje el sistema apagado.

- Number agreement between the (doubled) clitic and the full Indirect Object to which it refers:

(22) El usuario le puede cambiar el nombre al fichero.

- Gender and number agreement between the (doubled) clitic and the full Direct Object to which it refers:

(23) El nombre sólo lo puede cambiar el usuario.

- Gender and number agreement within the NP, between the head noun and its modifiers: articles, determiners and adjectives[52]:

(24) El sistema propone todas las otras opciones posibles.

[52] The number and gender of the head noun determine those of its modifiers.

Most of the operations that take place in the Post-Generation rules, particularly those related to Agreement Checking, but also all the rules dealing with euphonic issues (Section 3.2.4), remind of the functionalities of a typical Grammar Checker [Melero, 2001].[53]

3.4 Conclusion

In this chapter we have presented a detailed overview of a Generation grammar for Spanish. This grammar has been developed in the framework of Microsoft Research Natural Language System and is currently being used by MSR-MT, a Machine Translation system with thousand of users on a daily basis.

Careful attention is given here to the most relevant linguistic strategies that are behind the decisions taken in the process of writing the grammar rules. These strategies are indeed linguistically motivated and thus hopefully are transferable to other frameworks or Generation grammars written for the Spanish language.

[53] In fact we can conceive a Grammar Checker for Spanish as a translator between Incorrect Spanish and Correct Spanish, where the Generation component ensures the well-formedness of the output. See to this respect the English example in Section 4.5.

Chapter 4
Pre-Generation: a Step before Generation

The problem of most generators is, in effect, how to convert water into wine, compensating in the generator for the limitations of the application (McDonald, 2000)

In this chapter, we discuss the actual problems encountered when dealing with real-world applications. The Generator presented in this work is part of an MT system that is used to translate Microsoft technical documentation accessed by thousands of users on a daily basis.

The Generation grammar described in Chapter 3 is designed to be application-independent and source-language-independent; moreover, it expects its input to be flawless. However, in real life this is rarely the case. Generation, being the last step of the translation process, it is the perfect candidate to provide a certain degree of robustness to the whole system. It may be able to do so, only if it is capable of handling incomplete and inconsistent structures.

In this chapter we will propose a modular architecture for the Generator that provides both application independence and robustness.

4.1 Application-Driven Generation

The Logical Forms that are automatically produced by Transfer and that are input to the Generation component will not always be *perfect* LFs from the perspective of the target language. If the input to Generation was guaranteed to be perfect, then the Generation grammar described in Chapter 3 should suffice to always generate a correct output. If that were effectively the case, any faulty output could safely be considered a grammar bug or lack of coverage.

Unfortunately, we often find that the input to Generation is far from perfect: either it lacks some critical information or the information it carries is ambiguous or contradictory.

This may happen for a variety of reasons:

- One of the main sources of faulty LFs are limitations in the coverage of the Transfer module itself, which, due to its statistical nature, may easily suffer from data-sparseness during the training process.
- The coverage of the SL Analysis grammar can also (although less often in our case), bring about errors that are amplified through the application of the rest of the components.
- Assimetries in coverage or encoding criteria of the different dictionaries involved in the translation process are also a potential source of problems. In this context, the criteria to decide what is a Multi-word entry and what is not, turns out to be particularly controversial. For example, a complex conjunction or preposition in the Source Language may very well have a compositional translation in the Target Language: e.g. the English conjunction *as_in,* is perceived as a unit by an English speaker, and constitutes a single entry in the English dictionary, and therefore, in the input LF. However, its generic translation into Spanish (*como en*) is clearly compositional to Spanish ears, and thus does not have a specific entry in the Spanish dictionary. As a result, the Multi-word goes untranslated.
- Another obvious cause of problems are errors in the original source sentence, such as typing or spelling errors. These errors will probably affect all components involved in the translation process, including Generation.
- Lastly, deficiencies of the semantic model chosen may also entail problems in the input. In our case, the definition of the Logical Form may still not be sufficient to represent the whole gamut of linguistic phenomena involved in all the languages tackled by the system.

4.2 Examples of Real-World Problems

The examples that follow are real examples encountered in the course of the development of the Generation grammar.

4.2.1 Incompatible Information

During Transfer, source LFs are matched against the left-hand side of the LF mappings stored in MindNet and linked with their corresponding target LF segments. These target LF segments are stitched together into a single target LF.

In cases where no applicable transfer mapping is found, the nodes in the source LF and their relations are simply copied into the target LF. Bilingual dictionaries, containing only word pairs and their parts of speech, provide translation candidates for the alignment procedure and are also used as a backup source of translations during transfer.

This strategy, known as MindMeld (Section 2.6) allows Transfer to produce higher quality translations that are more sensitive to context, specific to the type of data on which it has been trained, than conventional Transfer rules. However, it does not come without a penalty. Since most of the procedure is automatic, the resulting LF may sometimes be defective from the perspective of the target language[54].

One of the most common cases of incompatible or contradictory information is the **co-existence of certain features with certain parts-of-speech**. Figure 42 shows an example where a noun (*acceso*) carries verbal features (i.e. Present, Progressive and Conditional). The correct LF is shown in Figure 43, where the head of the structure is a verb (*acceder*).

```
acceso ({Noun} +Pres +Prog +Condition)
|_a----registro({Noun} +Indef +Pers3 +Sing)
```

Figure 42: Transferred LF

[54] It is difficult to measure how often this happens, although one can assume that as the system gets more mature, and the set of examples on which it learns gets larger, some of the problems disappear.

```
acceder ({Verb} +Pres +Prog +Condition )
|_a----registro ({Noun} +Indef +Pers3 +Sing)
```

Figure 43: Correct LF

More generally, one of the most frequent sources of ill-formed input are feature combinations that are improbable or impossible in the target language (e.g. Spanish). An example is shown in Figure 44.

```
solucionar ({Verb} +Impersn +Pass)
|_Tobj----problema({Noun} +Def +Sing)
```

Figure 44: Transferred LF

This example illustrates a frequent hurdle in the translation between English (and other SLs) and Spanish, namely translation of passive and impersonal structures into sentences with reflexive pronoun *se*. The discussion involving *se*-sentences has a long history in the Spanish tradition and the different authors do not agree on the way to classify them. In our example, an impossible combination of diathesis coexists in one single LF: *impersonal* (coming from certain LF fragments with *se* in MindNet) and *passive* (probably coming from the original English sentence).

Problems even harder to detect or to expect are related to inherent *non-nativeness* of the Logical Forms resulting from MindMeld. Such an example is shown in Figure 45. In this case the verb *depender* inside a small clause is marked as Passive. Such a combination is impossible in Spanish (**ser dependido*). The right feature in this case is +Gerund (e.g. 'La respuesta política debe ser diferente *dependiendo* de la autoría de los atentados').

```
depender {+Pass}
```

Figure 45: Transferred LF

```
depender {+Gerund}
```

Figure 46: Correct LF

4.2.2 Lack of Information (i.e. Underspecification)

As bad as too much, is too little information:

```
ejecutar ({Verb} +Pres)
|_Tsub---extensión ({Noun} +Def +Sing)
|_en------Windows1 ({Noun} +Sing)
```

Figure 47: Transferred LF

The LF in Figure 47 processed as it is, yields the following (incorrect) sentence in Spanish: “La extensión ejecuta en Windows”. In order for the *se* to appear in front of the verb *ejecuta,* a feature -PassReflx- is needed in the verb node. Figure 48 shows the complete LF.

```
ejecutar ({Verb} +Pres +PassReflx)
|_Tsub---extensión ({Noun} +Def +Sing)
|_en------Windows1 ({Noun} +Sing)
```

Figure 48: Correct LF

However, arguably, a Generation grammar for Spanish should be able to generate the correct sentence (i.e. “La extensión se ejecuta en Windows”) just from the structure in Figure 47, even without the PassReflx information and based solely on the fact that *ejecutar* is a transitive verb that needs a *se* pronoun in order to be able to license (i.e. do without) its direct object.

While nobody disputes the fact that the LFs in Figure 42, Figure 44 and Figure 45 are defective and that an unsuspecting Generation grammar is not to blame if it yields a wrong output (“garbage in, garbage out”), it is not clear whether the LF in Figure 47 falls in the same category.

The question then is: What information should be available to a Generation grammar? In other words, how can be characterized the completeness and integrity of the input to Generation? We will take up this issue again in Section 4.3.1, when addressing the "problem of the source".

4.2.3 Wrong Information

We have seen that we can have too much information or too little; we can also have just the right amount, but it may be wrong.

One of the things that can go wrong in Transfer, obviously is the choice of the lemmas. Although **lexical selection** based in context is one of the strongest points of an Example Based translation system such as MSR, there are categories such as prepositions where contexts of apparition are much harder to identify.

Our first example has to do with the **choice of the copula**. The LF in Figure 49 yields the sentence: "Los datos son en el archivo"[55]. The correct LF is shown in Figure 50: here the copula is correctly expressed as *estar*. We will tackle this issue extensively in Chapter 6.

```
ser ({Verb}  +Pres)
|_Tsub----dato ({Noun} +Plur)
|_en----archivo ({Noun} +Sing)
```

Figure 49: Transferred LF

```
estar ({Verb}  +Pres)
|_Tsub----dato ({Noun} +Plur)
|_en----archivo ({Noun} +Sing)
```

Figure 50: Correct LF

[55] From the English original "The data are in the file".

Another hard problem of lexical selection concerns **choice of prepositions**, illustrated by the example in Figure 51 (wrong sentence: “la información por el cliente”) and Figure 52 (correct sentence “la información para el cliente”[56]).

```
información ({Noun} +Det +Sing)
|_por----cliente ({Noun} +Det)
```

Figure 51: Transferred LF

```
información ({Noun} +Det +Sing)
|_para----cliente ({Noun} +Det)
```

Figure 52: Correct LF

Apart from wrong lexical selection, the stitched LF may suffer from structural flaws, the result being an impossible structure from the perspective of the target language. In the example below, an Imperative clause lacks Subject, which is not possible in Spanish.

```
abrir ({Verb} +Imper)
|_Tobj----fichero ({Noun} +Def +Sing)
```

Figure 53: Transferred LF

```
abrir ({Verb} +Imper)
|_Tsub----usted ({Pron} )
|_Tobj----fichero ({Noun} +Def +Sing)
```

Figure 54: Correct LF

It may also happen that the structure is a possible structure in the target language but is inadequate given the context, as the one shown in Figure 55 which should yield the sentence: “Se resuelve cerrando el fichero” and not the one that actually results, i.e. “Se

[56] From the English original “the information for the client”.

resuelve por cerrar el fichero" (inheriting the preposition from the English original: "It is resolved by closing the file").

```
resolver ({Verb} +Impersn +Pres)
|_por----cerrar ({Verb})
         |_Tobj----fichero ({Noun} +Def +Sing)
```

Figure 55: Transferred LF

```
resolver ({Verb} +Impersn +Pres)
|_Manner----cerrar ({Verb} +Gerund)
            |_Tobj----fichero ({Noun} +Def +Sing)
```

Figure 56: Correct LF

These are cases where, due to lack of evidence in the corpus or generally to data sparseness, the performance of the MindMelding process is similar to a by-default transfer rule, where the SL syntax is mapped into the TL output.

4.3 The Nature of the Input to Generation

4.3.1 The "Problem of the Source"

Generally speaking, the real-world problems described in the previous section affect the integrity of the Logical Form that is the input to the Generation module. Drawing a parallel between Generation and Analysis, the problems encountered by Generation are comparable to "sentences with errors", also known as *ill-formed input*. In the case of Analysis, ill-formed input falls conceptually outside the coverage of the grammar. In Generation, due to the ***problem of the (lack of definition of the) source*** things are not as clear-cut.

David Mc Donald discusses this problem in his chapter about Natural Language Generation in [McDonald, 2000]. The problem of the source, he says, is in large part responsible for the relative lack of sophistication in the field compared to other NLP disciplines. We know virtually nothing about what a Generation system should start from. According to Mc Donald, the source is a “state of mind inside a speaker with intentions acting in a situation”. All these are slippery terms. From a computational perspective, this *state of mind* has to have a representation. However, there are a lot of formal implementable representations.

The lack of a consistent answer to the question of the source has been at the heart of the problem of how to make research on Generation intelligible and engaging to the rest of the Computational Linguistics community, and it has complicated efforts to evaluate alternative treatments. Differences in what information is assumed to be available has an influence on what architectures are plausible for Generation and what efficiencies they can achieve.

4.3.2 The Need for a Formal Specification of the Input

The kind of problems that we have reviewed in Section 4.2 basically fall into two types:

- Wrong input (e.g. a nominal head plus verbal features, etc.)
- Underspecified input (e.g. lack of tense, etc.)

The two types of problems are conceptually different. The issue of underspecification is more related to the *problem of the source*. Depending on what information is supposed to be available to the system, the Generator may need to perform more or less calculations in its core rules.

While dealing with wrong input is clearly a problem of **robustness** for the Generator, underspecified input could be considered a problem of **coverage**.

Although they are two different types of problems, they are also interrelated. Let’s take for instance the issue of the lemma of the copula. In our case, the LF is expected to provide Generation with the right lexical value of the copula (i.e. *ser* or *estar*). If the transferred value is *ser* and should be *estar*, then Generation faces a problem of wrong input (i.e. robustness). However, we can imagine a different system where the lexical value copula is not specified at the logical level. In that case, it belongs to Generation the task of generating the right copula in any case.

To sum up, if we want our Generator to be application-independent and reusable, then a formal specification of the input to Generation is compulsory, as [Cardeñosa et al, 2003] and other authors point out; in our case, a well-formed and complete Logical Form. On the other hand if we want our Generator to perform –and perform well- on real-world situations it needs to be able to confront ill-formed input. We address this conundrum in the following section.

4.4 Redefining the Architecture of the Generator

4.4.1 Coverage of the Generator vs. Robustness

Whenever we find the problems that we have exposed in Section 4.2, the question that comes up is: are we facing a failure in the coverage of the Grammar rules, or are we in front of a more generic issue of robustness of the whole system, and of the Generator in particular.

The question may seem idle. After all, what matters most is that problems are solved, one way or another. However, we want to do things in the best possible way, and the answer to that question will determine eventually the architecture that we choose for our Generator.

Let's look at the alternatives that we have when faced with input that makes our Generation grammar misbehave:

- We consider it to be a **lack of coverage**:
    - The solution to the problem is then to enlarge the coverage by adding more rules to the grammar and/or making changes to the existing rules in order to adapt them to the problematic input.
    - (Bad) Consequences:
        - Impossible to (pre-)define coverage: coverage is defined in an ad-hoc manner in the process of testing and developing the grammar.
        - Code difficult to maintain: core rules include a lot of exceptions and particular cases.
- We consider it to be a **problem of robustness**:

➲ The solution to the problem is then to improve the robustness of the Generator by *fixing* or *repairing* the input **before** the Generation rules apply.

➲ (Good) Consequences:

- 👍 Modular solution that favors generalization in the grammar rules: core rules are well-designed, follow specifications and do not need to be modified *ad-infinitum*.
- 👍 Allows (and encourages) a clear definition of the input: what is and what is not an acceptable input to the grammar.
- 👍 It is not necessary to distinguish between types of problems (wrong input vs. underspecified input): both are dealt with in the same way.
- 👍 *Fixes* to the input can be adapted to the characteristics of the application, the source language (in MT) or the input text type.

4.4.2 Adding a new layer: Pre-Generation

The Generator, as we have seen in the preceding sections, needs to be robust enough to deal with ill-formed input, in much the same way that large-scale Analysis grammars need to be able to process ungrammatical strings. To some extent, and based only on information about the language being generated, the Generator has to *fix* the Logical Forms, converting them into LFs that comply with the constraints imposed by the target language.

Let's reproduce here for convenience the figure that illustrated the Generation process exposed in Chapter 3 and let's remind ourselves of one of our main goals for the Generator, namely **application independence**.

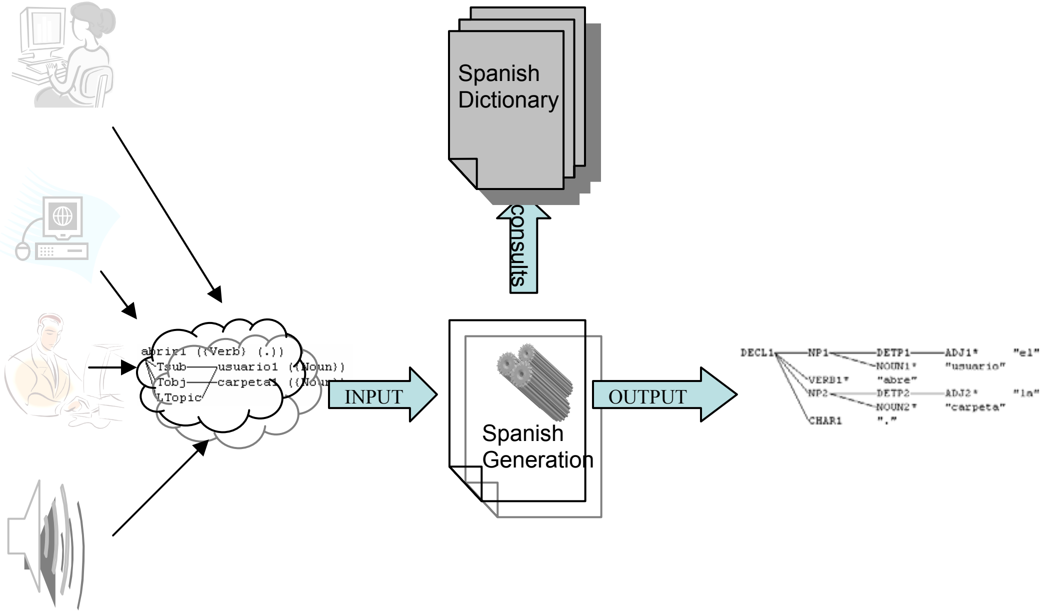


Figure 57: Input/Output of the Generation process

What is the best way to combine application-independence (and source-language-independence) of the Generation rules (as described in Chapter 3) and robustness?

Expanding the rule base to cover all the idiosyncrasies of the input would contaminate these rules and result in loss of generality. In order to maintain the integrity of the core Generation rules while accommodating imperfect input, we have opted to add a **Pre-Generation layer** to our Generator, thereby keeping the core rules free from ad-hoc or application-specific solutions.

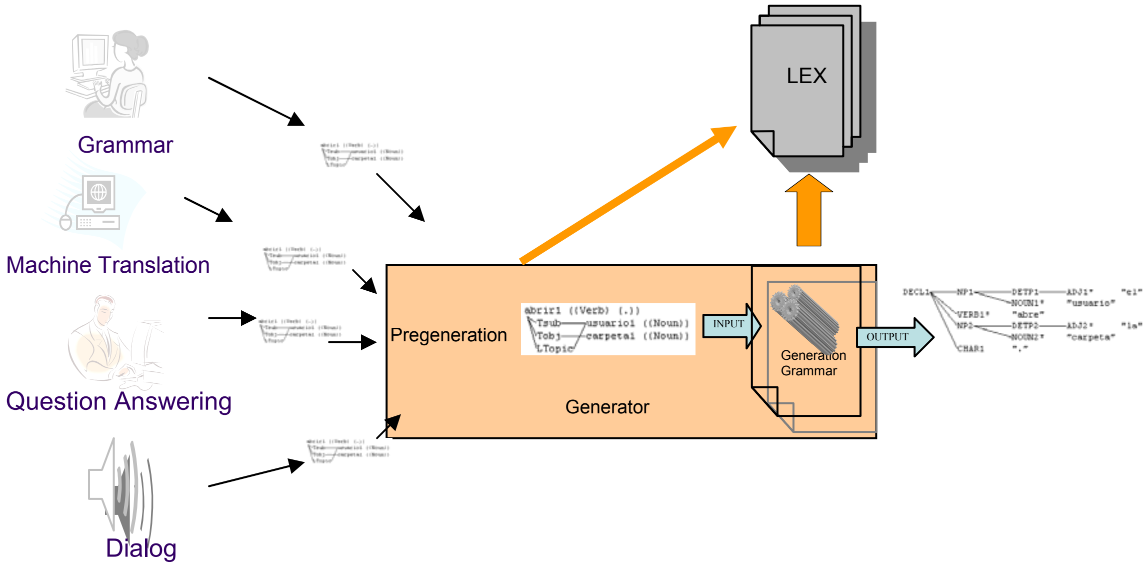


Figure 58: New Architecture for the Generator

Pre-Generation rules apply **before** the basic syntactic tree is built. They operate on Logical Forms, exclusively, taking as input a LF resulting from Transfer[57] and yielding a "*fixed up* LF", which can then be input to the Generation grammar. Pre-Generation rules can modify the input LF by adding or removing features, changing lemmas, or even changing structural relations.

Table 4 summarizes the difference between Generation rules and Pre-Generation rules:

Generation Rules	Pre-Generation Rules
They are designed to produce a Surface Syntactic Tree starting from a Logical Form which is both **complete** and **correct**.	They are intended to deal with Real-World Data and strive to establish the integrity of the Logical Form.
They operate on Syntactic Trees.	They operate on Logical Forms.

Table 4: Generation rules vs. Pre-Generation rules

[57] Or a different application, such as a DB Querying system, etc.

4.5 Examples of Pre-Generation rules in Multilingual Generation

In [Aikawa et al, 2001a], co-authored by the author of this thesis, we find examples of the use of the Pre-Generation module with examples coming from the Spanish, Chinese, Japanese and English Generation grammars developed in the framework of NLPWin. The examples in this section come from this paper.

- The **Spanish** component, has a Pre-Generation rule to deal with input LFs in which nominal nodes are assigned verbal bits (such as tense or aspect). Based both on the role of such a node in the LF and on the information present in the dictionary entry for the noun, this rule decides whether to turn the noun into a verb, remove the verbal bits from the noun, or generate a support verb for the noun. Figure 6 gives an example of an application of this rule in which the noun *acceso* (access), which is the head of a conditional clause, is replaced by the verb *acceder* (to access). This verb is retrieved from a link in the dictionary entry for the noun *acceso*[58].

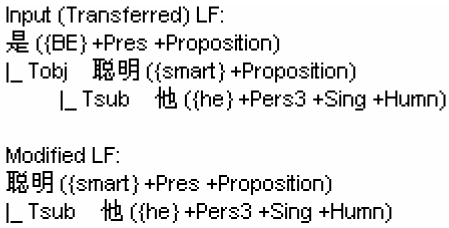
```
Input (Transferred LF):
acceso ({Noun} (si) +Pres +Prog +Proposition)
|_a----registro ({Noun} +Indef +Pers3 +Sing)

Modified LF:
acceder ({Verb} (si) +Pres +Prog +Proposition )
 |_a----registro ({Noun} +Indef +Pers3 +Sing)

 Generated string: Si está accediendo a un registro
```

Figure 59: Spanish Pre-Generation example

- From **Chinese**, we give an example of a rule that actually changes the structure of an LF. In our system, it is possible for the source and target languages to have different LF representations for similar structures. In English and other European languages, for example, the verb “BE” is required in sentences like “He is smart”.

[58] Links between verbs and predicative nouns (such as *acceso*) have been mostly automatically computed from VOX MRD dictionary, which identifies these nouns with lexical strings of the type “Acción de” or “Efecto de”.

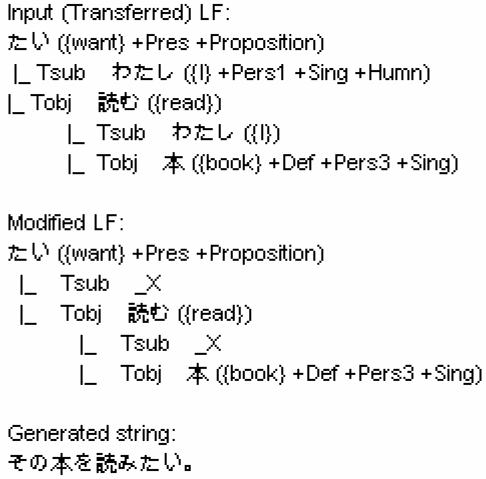
In Chinese, however, no copula is used. Instead, an adjectival predicate is used. Moreover, the LF is not intended to be an interlingua representation. Differences between languages and their LFs are tolerated. Therefore, Chinese uses a Pre-Generation rule to transform the be-predicate adjective LF into its Chinese equivalent as shown in Figure 7.



```
Input (Transferred) LF:
是 ({BE} +Pres +Proposition)
|_ Tobj    聪明 ({smart} +Proposition)
        |_ Tsub    他 ({he} +Pers3 +Sing +Humn)

Modified LF:
聪明 ({smart} +Pres +Proposition)
|_ Tsub    他 ({he} +Pers3 +Sing +Humn)
```

Figure 60: Chinese Pre-Generation example

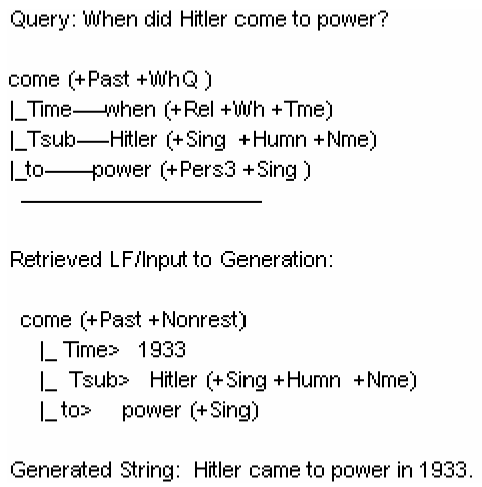
- Another example of a Pre-Generation rule, this time from **Japanese**, deals with the unspecified 1st/2nd person pronominal subject for particular types of predicates. The 1st/2nd person pronoun (わたし/あなた) is not used as the subject in sentences that express the speaker's/listener's desire (unless there is some focus/contrast on the subject). So, a Japanese Pre-Generation rule deletes the subject in input LFs that contain such predicates. For instance, below is the input LF, the modified LF, and the string produced from the English sentence "I want to read the book."



```
Input (Transferred) LF:
たい ({want} +Pres +Proposition)
 |_ Tsub    わたし ({I} +Pers1 +Sing +Humn)
|_ Tobj    読む ({read})
        |_ Tsub    わたし ({I})
        |_ Tobj    本 ({book} +Def +Pers3 +Sing)

Modified LF:
たい ({want} +Pres +Proposition)
  |_   Tsub    _X
  |_   Tobj    読む ({read})
          |_   Tsub    _X
          |_   Tobj    本 ({book} +Def +Pers3 +Sing)

Generated string:
その本を読みたい。
```

Figure 61 : Japanese Pre-Generation example

➲ The **English** Generation module of the MSR system has been used in experimental question-answering, dialog, and grammar-checking applications as well as in the MT application. The same module is used for all the applications. Here we describe some Pre-Generation rules motivated by applications other than MT. Among the Pre-Generation rules there is one that removes the marker for non-restrictive modification (*Nonrest*) from LF nodes that are not in a modification relationship with another LF node. So, for example, when the question-answering application is presented with the query "When did Hitler come to power," the system analyzes the question, produces an LF for it, searches its Encarta Mindnet (which contains the LFs for the sentences in the Encarta encyclopedia), retrieves the LF fragment in Figure 62, and sends it to the English Generation component. The LF that is the input to Generation is a portion of the LF representation of a complete sentence that includes the phrase "Hitler, who came to power in 1933." The part of the sentence that answers the question is the nonrestrictive relative clause "who came to power in 1933." Yet, we do not want to generate the answer to the question as a non-restrictive relative clause (as indicated by Nonrest in the LF), but as a declarative sentence. So, rather than pollute the core Generation rules by including checks for implausible contexts in the rule for generating nonrestrictive modifiers, we use a Pre-Generation rule to clean up the input.



```
Query: When did Hitler come to power?

come (+Past +WhQ )
|_Time——when (+Rel +Wh +Tme)
|_Tsub——Hitler (+Sing  +Humn +Nme)
|_to———power (+Pers3 +Sing )
 ____________________

Retrieved LF/Input to Generation:

 come (+Past +Nonrest)
   |_ Time>   1933
   |_  Tsub>   Hitler (+Sing +Humn  +Nme)
   |_ to>     power (+Sing)

Generated String:  Hitler came to power in 1933.
```

Figure 62: English Pre-Generation example

An example of a rule useful for an application such as grammar checking is the Pre-Generation rule that changes the quantifier “less” to “fewer”, and vice versa, in the appropriate contexts. When the LF input to the English Generation component specifies “less” as a quantifier of a plural count noun such as “cars,” this rule changes the quantifier to “fewer”. Conversely, when an input LF has “fewer” specified as a quantifier of a mass noun such as “luck”, the rule changes it to “less.” This rule would help transfer in the machine translation of Spanish “menos” to English “less” or “few”, and it would help the non-native, careless, or non-prescriptive English writer who interchanges “few” and “less”. The rule in no way hurts in the Generation of English from LFs that do not demonstrate this “non-native” characteristic. Rather this rule, along with the other Pre-Generation rules and the core Generation rules, make the Generation module robust and application-independent.

4.6 Conclusion

In order to increase robustness while keeping the independence of the Generator, we have presented in this chapter a slightly modified architecture of this Generator, which includes a Pre-Generation layer that applies before the Generation rules.

The Pre-Generation module, contrarily to the Generation grammar is very dependent on the input specificities. Each application, or source language in the case of MT, may (at least, partly) require different Pre-Generation rules.

This module is developed in an ad-hoc, empirical manner and is less stable than the Generation Grammar: Pre-Generation rules, such as the ones described in Section 4.5, which modify the transferred LF, may become inactive as the MT system is trained on larger and more varied data and produces more and more Target Language-like target LFs. However, at any state of development, Pre-Generation rules help add a very important degree of robustness to the system. By adjusting the structures that are passed on to Generation, they provide Generation with the best possible input to work with.

Because of its strong input-dependent nature, Pre-Generation rules seem good candidates to be modeled using statistical methods, as we will illustrate in Chapter 6 of this thesis.

Chapter 5
Evaluation

This chapter presents two different evaluation exercises of the Generation component. The first evaluates the Generator in the context of the application in which it is used: the Machine Translation system MSR-MT. Part of the results of this evaluation have been published by [Richardson et al, 2001a]. The second evaluation exercise tests the Generator in isolation by re-generating Spanish sentences from source Spanish sentences.

To further help put the Generator in context, we provide two more results, which can be of interest to the reader: The first one is a user satisfaction survey of the English-Spanish MSR-MT system, which is currently being used in a real environment; the other is an evaluation of a quickly assembled French-Spanish MT system that proves the reusability of the Spanish Generator.

5.1 Evaluation of NLG

Evaluation procedures and metrics have been increasingly seen as an essential tool for assessing progress in NLP, both for internal assessment and with respect to competing systems [Dale and Mellish, 1998]. It is critical for developers to be able to assess quickly and frequently each stage of development, so that they can compare or decide between two alternative implementations.

Various ways of measuring quality for MT have been proposed, some focusing on specific syntactic constructions (relative clauses, number agreement, etc.) [Flanagan, 1994], others simply asking judges to rate each sentence as a whole on an N-point scale [White et al., 1992-1994; Doyon et al., 1998], and others automatically measuring the perplexity of a target text against a bigram or trigram language model of ideal translations [Bangalore et al, 2000], [Papineni et al., 2001] and [Ringger et al. 2001].

Evaluation of translation fidelity typically has required human experts to make judgments. However, as human judgments are an expensive resource, the use of automatic metrics is becoming more and more common. A significant degree of correlation of these metrics with human judgments has been claimed. However, as pointed out by [Hovy et al, 2002] the amount of agreement among such measures has never been studied. In the evaluations presented in this chapter, only the judgments of independent human experts have been used.

When it comes to Natural Language Generation, we quickly realize that comparison between two different Generators is very difficult for two main reasons: one is the problem of the source, mentioned in Chapter 4, and the other has to do with the different conceptions of the Generator depending on the task for which it has been designed (shallow or deep, template-based or linguistic-based, etc.).

An additional problem is that it is difficult to evaluate the Generator isolated from the rest of the components of the application in which it is used. To overcome this handicap [Langkilde-Geary, 2002] and [Callaway, 2003] use as a test set the Penn Treebank for English, which provides their Generation components with a reliable, already processed input. Naturally, this is feasible only for the few languages with this kind of costly linguistic resource. However, a mature Analysis grammar such as NLPWin's Spanish grammar may also be able to provide an acceptable input to test Generation as we will see.

Moreover, [Campbell et al, 2002b] claim that Machine Translation may be a useful application for evaluating and driving the development of NL components. In particular, their paper shows through error analysis that there is a strong correlation between the quality of the translated output and the subjectively determined goodness of the Analysis. A similar claim could be made on behalf of Generation.

Taking all these considerations into account, we suggest two ways of evaluating the performance of the Generator:

1. In the context of the application in which it functions as the end component; in our case, the Machine Translation system MSR-MT.
2. As monolingual Generation, i.e. generation of Spanish sentences taking as input Spanish sentences.

Evaluating the MT application in order to evaluate the Generation component has the advantage that we can compare it with other MT systems. However, as we do so we are inevitably evaluating at the same time the rest of the components: the Analysis module, Transfer, etc. as well as the integration of all the components.

On the other hand, evaluating same-language Generation does not allow comparison with other systems, for the reasons exposed above, but has less interference from other components, most notably, Transfer (although not from Analysis).

The methodology used in each case is as follows:

1. In the context of MT: Compare with the best (or one of the best) competing systems and check for the best translation. See next section.
2. Monolingual Generation. For each sentence S from an evaluation corpus in Spanish:
    1. Get the corresponding LF(S) by analyzing S
    2. Apply the Generator on LF(S) to get a new sentence S'
    3. Compare the input S with the resulting S'
    4. Ideally S=S'

5.2 Evaluation of Generation in the context of MSR-MT

Translation quality, whether human or software generated, is difficult to quantify. Counting the number of errors in a translated sentence is not revealing because languages do not correspond on a word-for-word basis. An incorrect analysis of one word in the source language, for example, could lead to incorrect translation of several words in the target language. In addition, many errors made by MT systems cause subsequent errors within the sentence. Different systems, and for that matter, different human translators, can produce intelligible, accurate, but different translations of the same sentence. Therefore, for any input sentence, there is no single, ideal output sentence. Finally, some errors are more serious than others; therefore, all errors should not be assigned the same importance.

In any new application of MT technology, the developer will inevitably begin with questions about translation quality, asking what MT system produces the best output for a specific language pair or a special subject area. For several reasons, this question can be difficult to answer. First, the MT industry has no shared standards for measuring the quality of translations. Most MT companies have developed internal metrics for measuring output quality, but these metrics tend to be marketing driven and less than objective. Second, human language is inherently difficult to quantify because of its complexity and variety and because no two translators will always agree on the best translation of a given text. Finally, evaluating MT effectively often requires specialized knowledge of the languages involved, of linguistics, and of the inner workings of translation systems.

In order to evaluate the commercial quality of MSR-MT a competing commercial MT system need to be previously chosen. There exists a wide range of choices of MT systems for the English-Spanish pair ([Hutchins et al, 2004]). In order to assess MT output quality for their clients, IDC[59], developed, in the year 2000, suites of English, French, German, and Spanish text sampled from online IT, financial, and general news sources. Sentences were mostly 25 to 35 words long. Six language-pair translations, in which English was either the source or target language, were performed on these texts using five MT platforms (*Systran*[60], *T1 (Comprendium*[61]*)*, *TranscendRT*, *Barcelona (L&H)*, and *Logos*)[62]. The systems were evaluated by translating testsuites drawn from varied news sources and by applying a quantitative evaluation method to the systems' output. The output sentences were scored by native speakers of the target languages for intelligibility, accuracy, and style. Translations into English from Spanish and German were noticeably better than those in the opposite direction, while English-French and French-English

[59] International Data Corporation: http://www.idc.com/home.jhtml

[60] http://www.systransoft.com

[61] [Alonso and Thurmair, 2003]

[62] Most of the system used in this evaluation have changed hands since this report was elaborated. Indeed, the period between mid-2000 and the present has been more eventful for MT than the entire previous decade. The collapse of Lernout & Hauspie, its spin-off Sail Labs, the series of failed acquisition attempts of the Barcelona technology, SYSTRAN's Internet deployments, the release of IBM's WebSphere translation server, the demise of Logos, and the acquisition of MT by localization companies such as SDL, which acquired Transcend technologies in 2002, and Bowne Global Solutions, which now owns the Barcelona system, are only a few of the events of the period [Hutchins, 2003].

results were rated about equally good. Average scores for the five engines across 243 trials ranged between 0.44 and 0.54, with an overall standard deviation of 0.12 .

Systran and *Barcelona* performed the best overall and competed closely in most domains and language pairs and domains. Overall, *Barcelona* was judged to be the strongest system for the English-Spanish language pair. Scores for the general and IT domains were higher than those for all other systems.

Barcelona, initially from *Lernout & Hauspie*, later acquired by *Bowne Global*, was then the system of choice for evaluating MSR-MT English-Spanish. In evaluating progress of our system, we chose to do periodic, blind human evaluations focused on translations of single sentences. The human raters used for these evaluations worked for an independent agency and played no development role building the systems they tested.

5.2.1 Methodology

For each evaluation, five to six evaluators were asked to evaluate the same set of 200 to 250 blind test sentences. For each sentence, raters were presented with a reference sentence in the target language, which was a human translation of the corresponding source sentence. In order to maintain consistency among raters who might have different levels of fluency in the source language, raters were not shown the source sentence. Instead, they were presented with two machine-generated target translations presented in random order: one translation by the system to be evaluated (the experimental system), and another translation by a comparison system (the control system). The order of presentation of sentences was also randomized for each rater in order to eliminate any ordering effect.

Raters were asked to make a three-way choice. For each sentence, raters might choose one of the two automatically translated sentences as the better translation of the (unseen) source sentence, assuming that the reference sentence represented a perfect translation, or, they might indicate that neither of the two was better. Raters were instructed to use their best judgment about the relative importance of fluency/style and accuracy/content preservation. We chose to use this simple three-way scale in order to avoid making any a priori judgments about the relative importance of these parameters for subjective judgments of quality. The three-way scale also allowed sentences to be rated on the same scale, regardless of whether the differences between output from system 1 and system 2 were substantial or negligible.

The scoring system was similarly simple; each judgment by a rater was represented as 1 (sentence from experimental system judged better), 0 (neither sentence judged better), or -1 (sentence from control system judged better). For each sentence, the score was the mean of all raters' judgments; for each comparison, the score was the mean of the scores of all sentences. Note that the raters performed their task blindly, i.e. they did not know which sentence was the output of which system. Scoring, in the way explained, took place after the raters had evaluated the sentences

5.2.2 Results

Evaluation results for three different stages of the development of the English-Spanish pair are shown in Table 5. Training data was held constant for each of these evaluations. Test sentences were not part of the training corpus, and had not been seen by system developers.[63]

English-Spanish systems	***Mean preference score (5-6 raters)***	***Sample size***
MSR-MT 2/01 vs. L&H (*Barcelona*)	0.078 ± 0.13 (at 0.95)	250 sentences
MSR-MT 4/01 vs. L&H (*Barcelona*)	0.19 ± 0.14 (at 0.99)	250 sentences
MSR-MT 11/01 vs. L&H (*Barcelona*)	0.41 ± 0.12 (at 0.99)	250 sentences

Table 5: English-Spanish vs. alternative system

The evaluations summarized in this table compared February, April and November 2001 versions of MSR-MT's English-Spanish output to the output of the L&H (currently Bowne Global) *Barcelona* English-Spanish system for 250 source sentences. Five raters

[63] Results for the 2/01 and 4/01 evaluations published in [Richardson et al, 2001].

participated in the first evaluation, and six in the other two. Figure 63 shows the progress of MSR-MT compared to the reference system, over time. While in the first evaluation, MSR-MT is evaluated slightly worse than the reference system, after one year of development its results were clearly better.

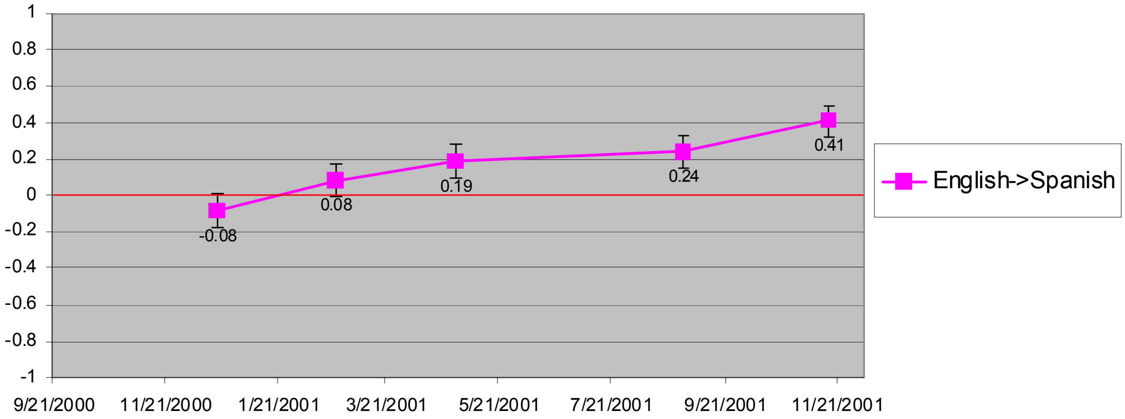


Figure 63: Average mean scores for all sentences in evaluations over time

Another dimension of the results of the last evaluation reported (November 2001) are graphically represented in Figure 64. This Figure clearly shows that the mean preference of the raters for MSR-MT is much higher than for the other system.

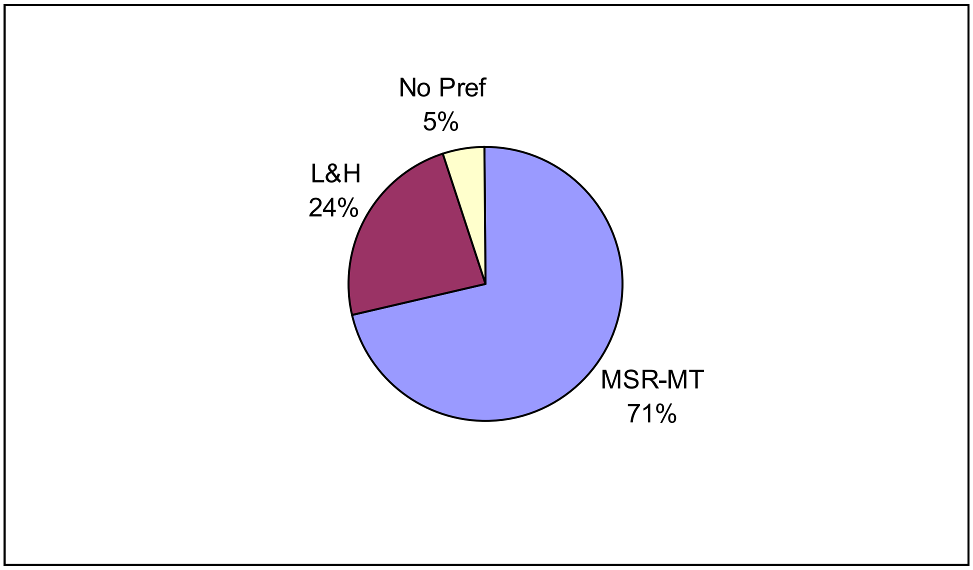


Figure 64: Sentence preference by mean score (11/01).

5.3 Evaluation of Monolingual Generation

5.3.1 Methodology

For the evaluation of monolingual Generation a new corpus, unseen to the developers was chosen, consisting in 903 sentences in Spanish.

The corpus was analyzed up to Logical Form with the Spanish analysis module and then re-generated into Spanish using the Generation grammar. The resulting 903 re-generated sentences were automatically compared to the original set and a total of 403 identical sentences were identified.

For the remaining 500 a human evaluation was then prepared:

Five evaluators with native knowledge of Spanish, from an independent agency (Butler-Hill), were asked to evaluate the same set of 500 test sentences. For each generated sentence, raters were presented with the original reference sentence.

Raters were asked to score each generated sentence according to its degree of similarity with the original sentence. Scores ranged between 1 (not similar) and 4 (greatest similarity).

5.3.2 Results

Table 6 shows the results for the human evaluation of the 500 re-generated Spanish sentences that were not identical to the original.

Ev 1	Ev 2	Ev 3	Ev 4	Ev 5	Mean
2.77	2.94	3.37	3.53	3.58	3.24

Table 6

If we assign a score of 4 to the set of 403 identical sentences and we combine this result with the obtained mean of 3.24 for the human evaluated sentences:

$$(4*403 + 3.24*500) / 903 = 3.58$$

We get a total score for the 903 re-generated sentences of **3.58** over a maximum score of 4 for maximum degree of similarity, which is a result remarkably good.

5.4 Survey on User Satisfaction

MT quality must be assessed in the context of each user's application. For example, using MT for a chat or instant messaging application is completely different from using MT to translate manufacturing assembly instructions. In words of Steve McClure, a research vice president in IDC's Software Research Group: “The key metric for MT is the quality of the resulting translation. Not only is this a somewhat subjective measure, but also its definition changes in the context of each application and user. Quality must be measured in the context of whether the user achieved its objective, not by what percentage of the translation was correct [McClure and Flanagan, 2003]”.

The English-Spanish MSR-MT is being currently used to translate Microsoft Product Support Services (PSS) Knowledge Base (KB) into Spanish. In this case, MSR-MT lowered the cost barrier to obtaining customized, higher-quality MT and PSS is now able to provide usable translations for its entire online KB. It can also keep current with updates and additions on a weekly basis something that was previously unthinkable both in terms of time and expense.

A pilot evaluation of user satisfaction was run before the actual deployment of the system. PSS surveyed during four months 1 in 20 users, which amounted to ca. 380 surveys. Users were advised that the article they were reading had been machine translated and then were asked for their overall satisfaction with the article (in a scale from 1 to 9).

The pilot was received well by customers; **overall satisfaction** was at 86%, (+/- 3% at 90% confidence).

The translation quality was considered satisfactory, 75.3% of respondents answered 5 or better on rating the **technical accuracy** of the articles, of those that answered 5 or lower, only 15.3% said it was due to the quality of the translation. In contrast, 82.3% of the human translated articles answered 5 or better on accuracy.

Finally the following question was asked to the user: “Did the information in the (machine translated) knowledge base article help answer your question?” The answer to this question was positive (yes) for machine-translated articles in a 49.7%. The same question got 51.2% of positive answers in the case of human translated articles to Spanish (compared to 53.6% for users who read US English articles).

The conclusion is that machine translated articles are as well received on average as human translated ones. The figures show that the English-Spanish system reaches a fair degree of acceptability and does not hinder comprehension of the technical documentation. This is confirmed by the very similar percentages obtained in the answer to the last question (49.7% for machine-translated articles and 51.2% for human translated ones).

5.5 An Experiment: French → Spanish in a Day

An experiment was carried on November 2002 to verify how fast a new language pair could be assembled in MSR-MT (for a description of a similar experiment, see [Pinkham et al. 2001]). Two mature linguistic modules were chosen: the French Analysis grammar and the Spanish Generation grammar. No specific linguistic tuning was performed on either module.

A bilingual Mindnet was built on French-Spanish parallel corpora (containing Technical Microsoft documentation) and only a small dictionary for function words was assembled by hand (7 person/hours).

SailLab’s well-established French-Spanish system was chosen as a commercial reference system to compare with.

A human evaluation of 500 sentences coming from Microsoft technical manuals (and not used for building the bilingual Mindnet) was performed following the procedure described in Section 5.2.1[64].

[64] As usual, five raters blindly evaluate two machine translated sentences accompanied by a reference sentences. Then the ratings are scored: 1 is assigned if the sentence translated with MSR-MT is better than the one translated by the control system, -1 if it is worse and 0 if none is better.

The result of the evaluation was very positive for the French-Spanish prototype, yielding an average score mean of 0.22 ± 0.1.

Figure 65 shows that the ready-made French-Spanish MSR-MT was preferred in more cases than Sail-Labs. In order to put these results into perspective, we have to remind that SailLabs is a general domain translator and that the bilingual Mindnet had been trained on specifically the same type of text used for the evaluation. However, what is interesting from our perspective in this successful experiment is that it proves the reusability of the Generator for a different language pair.

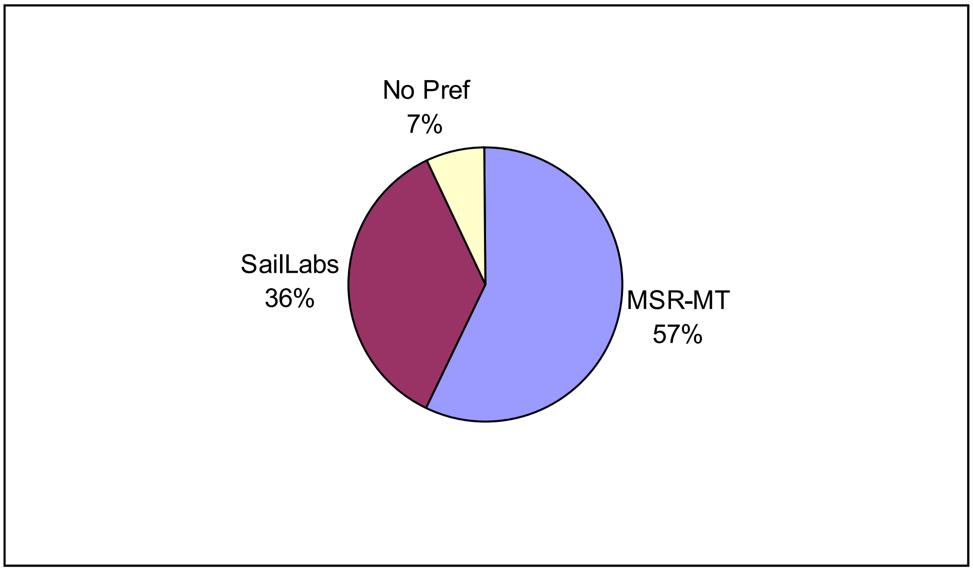


Figure 65: Sentence preference by mean score

5.6 Conclusions

In this chapter we have presented two evaluation exercises that aim at rating the Generator. Both exercises have used a team of human evaluators and the mean score of their ratings has been taken as final result.

In one case, we have evaluated the Generator through the evaluation of the performance of the complete MT system. In this case, we have successfully compared the

MT system of which the Generator is the end component to one of the best English-Spanish MT systems in the market.

In the other case, we have evaluated the Generator in isolation (as much as possible) from the rest of the components.

As both these evaluation exercises show, the Generator object of this thesis is a mature module with wide grammatical scope, which is used by a Machine Translation system of comparable commercial quality.

When evaluated in isolation, for the task of re-generating Spanish sentences, the rate of similarity with the original sentences is remarkably high.

In this chapter we also have provided the results of two other experiments that may be interesting to help put the Generator in context. One is a user satisfaction survey of the English-Spanish MT system that yields very encouraging results and shows the aplicability of the Generator described here in real-world situations.

The other is an evaluation of a quick assembled French-Spanish MT system that shows in practice the reusability and application independence of the Spanish Generator.

Chapter 6
Automatic Selection of the Copula

In this chapter we explore the use of decision tree classifiers (DT) for automatically learning the lexical selection of the Spanish copula in the framework of our Generation grammar.

Spanish has two different copulas, *ser* and *estar*, which are both translated into English as 'to be'. The use of one or the other in certain cases involve a complicated casuistry, over which there is not a total agreement among the linguists who have studied the phenomenon. Generally, s*er* is said to express permanence, identity or inherent quality, while *estar* is typically used for temporary conditions and location.

Here we use Machine Learning techniques to leverage large amounts of data for discovering the relevant conditioning features for the selection of the copula. As a ML technique for the problem at hand, we chose Decision Tree learning, a practical approach to inductive inference in widespread use.

In this chapter we also evaluate the usefulness of selecting the copula in Generation rather than doing it in Transfer. We show that it is possible to infer from examples, by means of decision trees, the contexts for this non-trivial linguistic phenomenon with high accuracy. We evaluate the impact of the linguistic domain of the training data on the quality[65] of the statistical model and discuss the differences between the results obtained using corpora from different domains.

[65] By quality we mean overall accuracy in the same type of text and across text types.

6.1 Motivation for the Experiment

6.1.1 Why Deal with Copula Selection in Generation rather than in Transfer

1. The correct generation of the copula is a specific instance of the general problem of lexical selection. In an Example-Based system, such as MSR-MT, this problem is generally tackled by Transfer. However, as pointed out in [Aikawa et al., 2001] and discussed in Chapter 4, the Generation component, being ultimately responsible for the fluency and grammaticality of the output, must reevaluate some of the decisions made by Transfer. The Pre-Generation module needs to identify contexts with conflicting or incomplete information and *repair* these contexts before they reach the Generation grammar proper. This solution is in line with others so-called "Generation-Heavy" approaches [Habash and Dorr, 2002], where most of the weight for output well-formedness is put on the Generation component of the system.
2. We find further motivation from a linguistic perspective: While it is true that the choice of the copula is an instance of lexical selection, the problem is in fact closer to the choice of a specific preposition than to the choice of a word with full semantic content, such as, for instance, the noun *mesa*. As discussed in Section 2.6, Transfer is able to make decisions that are context sensitive, learning from bilingual examples. It is also domain sensitive; for example, if trained on technical texts about computers, it will more likely choose *tabla* over *mesa* as possible translations for the English word *table*. The problem of selecting the right copula seems of a different nature.
3. Another practical consideration comes to mind: If the lexical value of the copula can be learned, it will be learned on Spanish texts or, if using parallel texts, only the Spanish part would provide information that is relevant for the chice at hand. Training of Example-Based Transfer, on the contrary, requires bilingual parallel texts, which are much more difficult to obtain than simple Spanish texts.
4. An additional bonus of shifting the burden to Generation is that the results can be used on inputs coming from different sources: DB querying, summarization, different transfer modules, etc.

5. Our last argument is of a more semantic nature. It has been argued [Campbell and Suzuki, 2002a, 2002b] that the lexical occurrence of a copula in surface syntax is a language-particular grammatical device, subject to different principles in different languages and that, as such, it should have no place in a language-neutral representation. This is sustained by evidence coming from languages that have very different surface requirements for the copula to appear, including English [Partee, 1977; Becker et al, 2000], Chinese, Japanese [Campbell and Suzuki, 2002a, 2002b], Russian and Arabic. If the copula does not have a lexical representation in Logical Form, then we could not expect it to be transferred, and Generation would have to create it based purely on information present in the Logical Form as well as on information coming from the target language lexicon.

Taking all these considerations into account, it seems therefore reasonable to deal with this problem in Generation rather than in Transfer.

6.1.2 Why use Decision Trees

6.1.2.1 Decision Trees: a Classification Tool

Decision trees (DTs) are a useful machine-learning tool for feature-based classification tasks that use discrete-valued features. The following is a brief description of their operation and background.

A classification space is defined by a set of features, and each classification instance or case is described by a set of values for those features (a set of feature-value pairs). The goal of the classification task is to decide which class does each case belong to, out of a given set of classes (typically two). A DT is a tree with (non-terminal) nodes and (terminal) leafs. Each node is a test on one of the features, with a different branch for each possible value of the feature. Each leaf defines the most probable class associated with that leaf. When presented with a case, the DT first poses the test corresponding to the root node. Depending on the outcome (depending on the value that the tested case has for the tested feature) the corresponding branch is selected. This branch leads to the following node where another test is formulated, and the process is repeated until a leaf is reached. The case is then classified in the class associated with the reached leaf. It is also possible to map DTs onto if-then rules in a straightforward way to improve human readability and manipulation.

The construction of DTs is a supervised learning procedure. A training set of labeled cases is required, that is, a set of cases for which the class is known beforehand. Defining a node, amounts to deciding which feature will be tested at that node. The usual, greedy, approach is to select the feature that best discriminates between the cases at the node; in other words, the feature that best partitions the cases in such a way that each subset belongs mostly to one of the classes (i.e. the feature that best classifies the cases). The whole training set is used for the definition of the root node, and the discriminating power of all the features is evaluated to decide on the first test. A new branch is defined for each value of the selected feature and the training set is partitioned accordingly. Then each subset is used, recursively, for the definition of the next node. Eventually, all cases in a subset belong to the same class, or no features remain to be tested, or the subset of cases for a branch is empty. A leaf is then defined associated with the most common class.

A test set of labeled cases, different from those used for training, is then used to evaluate the accuracy of the DT (i.e. the ratio of test cases properly classified), and its generalization capabilities or how well it performs on the unseen cases. Results are usually improved by means of pruning off the branches that are based on subsets of cases that are too small, since those are statistically non-significant and their capability for generalization is poor. A usual approach for pruning is to use yet a third set of labeled cases, the validation set. The pruning is then based on improvements of the accuracy of the DT over the validation set.

Thus, DTs are inductive learning algorithms, belonging to the field of Machine Learning, whose inherent nature makes them well suited for tasks with discrete-valued features, although they can be adapted to continuous-valued features. Alternative classification algorithms, mostly for the continuous case, include genetic algorithms [Holland, 1986], neural networks [McClelland and Rumelhart, 1988] and statistical approaches such as k-nearest neighbor classifiers [Hunt, 1975], among others.

A good introduction to DTs can be found in [Mitchell, 1997] and a detailed explanation of one of the most popular implementations, the C4.5 algorithm, together with practical issues can be found in [Quinlan, 1992]. DTs have their origin in Hunt's Concept Learning System, CLS [Hunt et al., 1966], and early DT learning systems include CART [Friedman, 1977; Breiman et al., 1984], ID3 [Quinlan, 1986] and ASSISTANT [Cestnik et al., 1987]. Several DT implementations are available nowadays, including the WinMine toolkit [Chickering, 2002] used in this work (the WinMine toolkit is in fact a Bayesian Network learner that includes DTs for modeling the statistical behavior of variables).

Different approaches exist in the literature for dealing with DT implementation issues, such as pruning strategies [Esposito et al., 1999] or discrimination measures [López de Mántaras, 1991]. DTs have been used for many different applications, from medical diagnosis to financial risk assessment.

DTs have been also extensively used in NLP. [Knight and Chander, 1994] use them to automatically generate the indefinite (*a*/*an*) or definite article (*the*) in document post-editing, in English. [Minnen et al., 2000] use a a memory-based learner (TiMBL 3.0) to address a similar problem but they also include the possibility of not generating any article. Both use the Penn Treebank as learning corpus. Of particular relevance to this work is Amalgam, the Machine-learned Generation grammar module described in [Corston-Oliver et al., 2002].

6.1.2.2 Our Option

The main reason why we use decision trees (DTs) rather than other statistical machine learning tools is their adequacy to the problem formulation, in other words, the structure of the problem fits the structure of the algorithm. The task of distinguishing between *ser* and *estar*, which involves a complex combination of linguistic conditions, can be easily expressed in terms of a binary classification task, based on the values of a set of discrete-valued linguistic features.

DTs learn rules that have the structure of disjunctions and conjunctions of attributes. While other machine-learning tools can also perform this kind of function, DTs are more suited to this problem both from a structural perspective and from the learning algorithm perspective. Neural networks, support vector machines, Bayesian nets, deal with discrete attributes as if they were continuous but decision trees are naturally defined in terms of discrete features.

Moreover, some of these techniques, such as Hidden Markov models, neural networks and support vector machines, use a fixed length analysis context, whereas DTs allow for variable length contexts corresponding to whole phrases or sentences. Bayesian networks, which use graphs, also allow for context flexibility, but the cost of their training is subject to combinatorial explosion.

The feature set for the classification task includes many features that are not always relevant to the current context, therefore acting as noise. DTs filter this noise by automatically performing feature selection, and automatically pruning the tree (i.e. the undesired features), as the contexts require. Bayesian networks or HMMs can properly deal with contextual information, but the structure of the problem must be known and programmed beforehand.

6.2 *Ser* or *Estar*, that is the Question: Some Linguistic Facts

The uses of *ser* and *estar* are one of the biggest hurdles that a student of Spanish as second language encounters. Numerous studies have been devoted to this issue and there have been many attempts to describe it in an adequate and systematic way. As Cirot[66] concludes at the end of one of his papers, "We are faced with a little mystery". More specifically, he says that the uses of *ser* and *estar* with an adjective have become a sort of enigma of mythical proportions, which has been interpreted in many different ways, none of them satisfactory.

Equally uncertain is the definition of what a copula is. Most linguists agree that *ser* and *estar* are the only copulas; other include verbs such as *parecer*, *semejar* or *quedar*; yet others consider that only *ser* can be considered a copula, because it is the only verb that is truly empty from a semantic perspective (for a discussion on this issues, see

[66] Cited by [Porroche ,1988, p. 13]

[Fernández Leborans, 1999]). There are also different approaches on what constitutes a copulative use versus what is a predicative use of *ser* and *estar*.

For the purposes of our experiments, we will use the term "copula" for both *ser* and *estar* in any of their uses, except for the auxiliary[67] (see below). And we will refer to the argument of the copula that is not the subject, as the "predicate"[68].

Among all the diversity of opinions, a certain common ground can also be found. Most of the studies agree on the following facts (summarized from [Porroche, 1988]):

1. Only *ser* can be used with a nominal predicate, which can be a noun, an infinitive form of a verb, or a pronoun[69].

   (25) Juan es {médico/mi mejor amigo/una buena persona/...}

   (26) *Juan está {médico/mi mejor amigo/una buena persona/...}[70]

2. *Ser* means "to exist"[71], "to happen" and "to take place"[72].
3. Only *estar* is used with verbs in gerund[73], such as:

   (27) La actriz está maquillándose.

   (28) *La actriz es maquillándose.

[67] Although the traditional distinction between copular and auxiliary uses is generally accepted, there have been proposals for the unification of both functions, on the basis of certain similarities: semantically empty verb carrying certain aspectual values, adjacency constraints with the predicate, etc. [Fdez Leborans, 1999]

[68] We use the term *predicate* in order to purposely avoid the more usual term, at least in the Spanish tradition, of *attribute*. Since *attribute* is also used in this thesis to describe a type of information present in the syntactic record or the logical form, the use of this term in the sense of "argument of the copula" would lead to confusion.

[69] Fernández Leborans, citing [Camacho, 1993], observes that the only NPs that are compatible with *estar* are those expressing a scalar measure, such as *El Barcelona está el segundo en la clasificación.*

[70] The reader is reminded that an asterisk preceding a sentence is a mark of ungrammaticality.

[71] The use of *ser* meaning "to exist" is very rare and belongs to fixed formulae, such as: *Érase una vez*.

[72] The use of *ser* meaning "to happen" is still alive; it usually requires an event noun as a subject as well as a space-temporal location: *El examen es esta tarde/ en el aula B.*

[73] This use of estar has been traditionally considered auxiliary, as we do in our experiments (and as is done in NLPWin). However, as some linguists have pointed out (Fernández Leborans, pp. 2432-35) there is ground for considering the <estar + gerund> constructions as being parallel to the copular constructions <estar + adj/pp>: *Ana está durmiendo/Ana está dormida*.

4. *Estar* is used to express location.
5. The past participle, used with *ser*, expresses the action in passive voice; used with *estar*, it refers to a state resulting from the action.

(29) El edificio fue destruido (por las bombas).

(30) El edificio está destruido.

The main source of disagreement is the interpretation of the alternative uses of *ser* and *estar* with adjectives. Many linguists, basing their observations on a reduced number of examples, explain the differences between the two copulas using opposite terms. For example, *ser* is said to express permanence, identity or inherent quality while *estar* is typically used for accidental or temporary conditions:

(31) La nieve es blanca.

(32) El agua está caliente.

It is easy to find counterexamples to the notion of permanence being expressed by *ser* and temporality being expressed by *estar*, such as the apparent paradoxes in (33) and (34), where *joven* is by no means permanent, and *muerto* does not seem to be a temporary condition[74].

(33) Él es joven.

(34) Él está muerto.

For this reason, Fernández Leborans prefers to speak of stable (non episodic) and unstable (episodic). In this way, a predicate requiring *ser* may be non permanent, but stable, such as:

(35) En su infancia, Juan era rubio, delgadito y muy travieso.

[74] The fact that *muerto* is a state resulting from an event (*morir*), makes it actually temporal.

Many authors approach the capability of a predicate adjective to combine with *ser*, *estar* or both based on their “perfectivity”:

- [+Perfective] adjectives: *contento*, *descalzo*, *harto*, *lleno*...
- [-Perfective] adjectives: *capaz*, *mortal*, *idóneo*, *válido*...
- [±Perfective] adjectives: *gordo*, *alto*, *alegre*, *amable*...

+Perfective adjectives combine with *estar*; -Perfective ones combine with *ser* and ±Perfective combine with both.

To describe the alternative use of *ser* and *estar* with the same adjectives, certain authors [Falk, 1979] distinguish, on pragmatic terms, between “general rule” and “individual rule”. By using *ser*, the speaker classifies the entity referred by the subject with respect to a category assumed by general consensus (the class of “*personas guapas*” in example (36)); on the other hand, by using *estar*, the speaker classifies the entity referred by the subject with respect to itself (the person called *María*, in example (37)); in this latter case, the quality expressed by the adjective predicate *guapa* would be assigned to the subject as a temporal deviation of its normal characterization.

(36) María es guapa.

(37) María está guapa.

However, Fernández Leborans points out that it is perfectly possible to refer to a temporal characterization of the subject without taking into consideration her “normal” characterization (i.e. the fact that *María está guapa* is independent of the fact that *María* ‘is pretty’ or ‘is not pretty’).

In Table 7 and Table 8, we present a summary of the different uses of *ser* and *estar* in terms of the function that they perform, as described by Porroche. Table 7 shows the functions that can be performed by both verbs, and Table 8 shows the functions that can only be performed by *ser*.

	PREDICATIVE	AUXILIARY	ATTRIBUTIVE
SER	**Existential** **(case 1)** *La reunión **es** a las 6* *La fiesta **es** en mi casa*	**Action (passive voice)** **(case 3)** *La casa **fue** construida por su padre*	**Inherent quality** **(case 6)** *Juan **es** (un) médico* *Juan **es** guapo* *El globo **es** de colores*
ESTAR	**Locative** **(case 2)** *Él **está** en casa.* *El libro **está** sobre la mesa.*	**State resulting from an action (stative passive)** **(case 4)** *La casa **está** construida* **Duration of the action (progressive)** **(case 5)** *Juan **está** leyendo*	**Temporary conditions** **(case 7)** *María **está** muy guapa* *Mi jefe **está** de vacaciones*

Table 7: Functions that *ser* and *estar* can perform

IDENTITY	TOPICALIZATION
Both arguments are in an equative relation **(case 8)** *Juan **es** el médico*	**One part of the sentence is topicalized** **(case 9)** *Con María **es** con la que se casa Juan*

Table 8: Functions only performed by *ser*

As mentioned above, we will leave aside the auxiliary uses of *ser* and *estar* in passive (case 3) and progressive (case 5) constructions. The main reason for this is that the auxiliary verb does not appear in the logical form as a lexical item but as a feature in the structure[75].

The information presented in Table 7 and Table 8 can be reformulated in configurational terms, i.e. according to which type of phrase can appear as predicate of each of the copulas, as shown in Table 9.

		AJP	**PP**	**NP**	**Past Participle**
SER	Case 1		X		

[75] The information about passiveness or progressiveness is expressed in the LF by means of the bits Pass and Prog, respectively (see section 2.5).

	Case 6	X	X	X	
	Case 8			X	
	Case 9	X	X	X	
ESTAR	Case 2		X		
	Case 4				X
	Case 7	X	X		

Table 9: Syntactic Phrases that combine with *ser* and/or *estar*

6.2.1 "Easy" Cases

In principle, from a Generation perspective, the non-shaded part of the table corresponds to the least problematic cases. Only *ser* can be used in constructions that involve Noun Phrases (NP) as predicates. Noun Phrases can be headed by a noun, a pronoun or an infinitive, as in the following examples:

(38) La ballena es un mamífero.

(39) La persona a la que más quiero eres tú.

(40) Eso es engañar a tus padres.

The combination of <*estar*+NP> as shown in the following examples is always ungrammatical:

(41) *María está la profesora.

(42) *Ese bolígrafo está mío.

(43) *Eso está mentir.

The same happens with constructions that involve a copulative (not auxiliary) verb and a past participle. Only *estar*, as a main verb, can appear in that situation[76]. When *ser* appears together with a past participle, it is usually a passive voice construction. In these

[76] But see below for a caveat on this assumption.

constructions, *ser* functions as an auxiliary, not as a main verb, as in the following example:

(44) El cadáver fue descubierto por una pareja de novios.

Not surprisingly, the cases that we call "easy" from a Generation perspective are in general terms the same cases that are considered non controversial by the scholars who have studied the uses of *ser* and *estar*.

6.2.2 "Difficult" Cases

The distinction between the cases in the shaded section of the table is clearly more challenging. It involves properties of the subject as well as of the predicate. Cases 6 and 7 are the hardest to predict. As discussed previously, the selection of the copula in those cases entails aspectual interpretations, sometimes difficult to deduce from context.

Some predicate adjectives can be used with both verbs, provided that the nature of these adjectives allows for the two aspectual interpretations. Thus, examples (45) and (46) are both possible.

(45) La nieve es fría

(46) La nieve está fría

Other adjectives do not have this flexibility. For example, *disponible* can only go with *estar* and *eterno* can only go with *ser*.

(47) *La cantidad todavía no es disponible.

(48) *Su amor siempre estará eterno.

The set of adjectives traditionally considered as "classificatory", which are used to assign the subject into a particular class, always take *ser*[77]. These adjectives are usually human attributes and can be recategorized into nouns. They express notions such as:

- Nationality or place of birth: *español, francés, londinense, madrileño...*
- Belonging to a religious, political, social or intellectual current: *cristiano, musulmán, ateo, burgués, socialista, renacentista...*

Modal adjectives, such as *posible*, *cierto*, *probable...* never use *estar*.

The largest class of adjectives corresponds to those traditionally known as "qualifying" adjectives. This class of adjectives traditionally denote qualities of different types: physical (*rubio, alto, grande...*); psychological (*inteligente, tímido, valiente...*); moral (*bueno, egoísta...*); sensorial perception (*agrio, suave, dulce,...*); relating to certain social norm (*soltero, pobre, vulgar,...*); temporal (*viejo, moderno,...*); by comparison (*semejante, distinto,...*) . All of these adjectives have a strong preference for *ser* but many of them can also go with *estar* in certain pragmatically acceptable contexts.

(49) El semáforo está rojo.

(50) El semáforo es rojo.

In example (49), the properties of the subject (*semáforo*) allow for an accidental or circumstantial interpretation of the condition of 'being red', while in (50) this condition is an inherent quality of the subject. Porroche says that if the subject is not able to experience change, the use of *estar* is not possible. We could add that if the subject is not able to experience change *with relation to the property expressed by the adjective*, the use of *estar* is not possible.

Other adjectives have different semantic readings depending on whether they are used with *ser* or with *estar*:

[77] With an ironic or humorous intention, there always exists the possibility of using them with *estar*, such as in: *Estás muy burgués últimamente.*

(51) El niño es muy bueno[78].

(52) El niño ya está bueno[79].

The cases involving <copula+PP> can be explained based on the same principles exposed above for <copula+AJP> constructions (cases 6 and 7); they also include predicative complements exemplified by cases 1 and 2. Again, the use of one copula or the other involves a complex interaction of properties among the subject and the predicate. In general, PPs in case 6 have an adjectival nature, i.e. they can be turned into noun modifiers, as shown in examples (53)- (54) and (55)- (56).

(53) Ese chico es de pocas palabras.

(54) Es un chico de pocas palabras.

These PPs tend to be headed by preposition *de*, but as example (55) shows, this is not a necessary condition.

(55) Ese vestido es sin mangas.

(56) Es un vestido sin mangas.

On the other hand, PPs headed with *de*, can also go with *estar* (case 7), as in example (57). In those cases, the PP has an adverbial rather than adjectival nature, and it involves a transitory condition.

(57) El comerciante está de paso.

(58) ?Aquí se aloja el comerciante de paso.

[78] *Bueno* meaning *bondadoso*.

[79] *Bueno* meaning *repuesto*, *bien de salud*.

Location is typically expressed with *estar* (example (59)) except in the cases when "some event is taking place", as in (60), then we would rather use *ser*.

(59) Juan está en mi casa.

(60) La fiesta es en mi casa.

In examples (61) and (62), the subject *clase* has two different senses. In (61) it involves the notion of activity. It has the same predicative sense that we find in a collocation such as *dar clase*; it can be paraphrased by *curso*. On the other hand, in (62), it refers to the physical entity; it can be paraphrased by *aula*.

(61) La clase es en la cuarta planta.

(62) La clase está en la cuarta planta.

6.3 Building DT Models for Predicting the Copula

6.3.1 Design of the Experiment

In the initial stages of the experiment, we considered dealing with the "easy" cases by means of a hand-coded rule, using basic morphosyntactic information, such as part-of-speech and morphological inflection, and explore the use of decision trees with the difficult cases only. However, our preliminary experiments with decision trees showed us that things are always more complex than theory makes them look. As a matter of fact, once we approached real data, we realized that the separation line between easy and difficult cases was blurred. As it happens, decision trees also proved to be a useful tool for exploring the data.

Constructions with past participles (case 4), for instance, looked like the "easiest" case. As it turns out, the distinction between "true" past participles and adjectives that morphologically "look" like past participles is far from clear in many NLP systems, and the Spanish NLPWin system is no exception.

Thus, *elevado* in example (63), or *variada* in example (64) have no adjective entry in the dictionary but are analyzed as the past participle of *elevar* and *variar*, respectively.

(63) El volumen del altavoz es demasiado elevado.

(64) La coloración del pelaje es muy variada

A rule that would prescribe unconditionally the use of *estar* with past participles would get those examples wrong.

Examples (65) and (66) constitute a minimal pair, where one of the past participles (*elevado*) behaves more like an adjective and the other (*rebajado*) keeps its verbal nature.

(65) El precio es muy elevado.

(66) El precio está rebajado.

Examples (67) and (68) share the same past participle predicate and the same subject, but the modifier is different. In (67), the predicate is modified by an intensifier, while in (68) it is modified by a manner adverb.

(67) Juan es muy educado.

(68) Juan está bien educado.

Taking all this into consideration, we decided to approach the problem globally, without making apriori distinctions among the different cases described in Section 6.2.

The task that we wanted our statistical model to learn was expressed in simple Boolean terms: given this context, should the lexical value of the copula in this clause be *estar* or not.

The sequential steps of the experiment are summarized here:

1. Collect two Spanish corpora from different domains
2. Parse the corpora using NLPWin to obtain Logical Forms

3. Extract linguistic features from parsed contexts of *ser* and *estar*
4. Build decision tree models using the features extracted, with the aim of predicting the Spanish copula in context
5. Use the predictions of the models to generate the copula in the Spanish Generation grammar
6. Evaluate and analyze the results

6.3.2 Training Corpora

Decision trees, as other ML techniques, learn linguistic information from an annotated text corpus. How this corpus is and how we annotate it is of great importance for the learning task. [Banko and Brill, 2001] argue that as long as the training set is big enough, it does not matter much which learning technique is used. Still, they claim, it is important to devise efficient ways in which to minimize the human effort spent in the annotation of the corpus.

Since logical forms constitute the input to the Generation component, it seemed reasonable to train the models using logical forms produced by analyzing text in Spanish. The advantage of our specific Generation task is that the annotation for the target feature comes at no cost, since the correct instantiation of the copula appears without errors in any reasonable text of native Spanish.

One of the interesting aspects of the experiment is that the models are trained on monolingual (i.e. Spanish) data and are then used on a multilingual environment, on logical forms that have been produced by transfer. As opposed to other systems where manual annotation is needed in order to produce good training material, we are able to use any Spanish text available, after it has been analyzed by NLPWin. This fact gives our approach an extra advantage over dealing with the problem in transfer, since transfer needs bilingual text to learn the right context. For obvious reasons, text in Spanish alone is much easier to come by than parallel corpora.

[Bangalore et al., 2001], among others, have considered the impact that the type of corpus has on the quality of the stochastic Generation components, although their main concern is the quality of the syntactic tagging (automatic versus manual, etc.). To our knowledge, little has been explored on the impact that different textual domains have on the learning of a specific linguistic task. We wanted to perform our experiments using different types of texts. For this purpose we built our models in parallel using two very different types of corpora: encyclopedic text from Encarta and technical text from computer manuals.

6.3.2.1 Encarta Corpus

The Encarta corpus is a collection of encyclopedic articles used for the Spanish Encarta encyclopedia. It contains approximately 400,000 (400K) sentences. It is a highly edited text with a language style characteristic of encyclopedic text. There is a great abundance of definitions and descriptions as exemplified by the following sentences,

(69) El haba es amarilla o negra, pequeña y redondeada, y se emplea en guisos, sopas y ensaladas.

(70) Lutero era un monje agustino y profesor de teología en la Universidad de Wittenberg.

In order to maximize the learning power of the training data, we wanted it to be as relevant as possible. To build the training corpus, we did the following:

- We ran a Perl script on the Encarta corpus that automatically extracted sentences containing any inflected form of the verbs *ser* and *estar.* This included sentences where *ser* and *estar* functioned as auxiliaries (as in examples (27) and (29)) as well as copulas. It also included sentences with words that were homographic to some form of *ser* or *estar* (e.g. the adverb *fuera);*
- We then processed the sentences that had been extracted by the Perl script, using NLPWin and automatically filtered out the sentences where *ser* or *estar* were not full verbs[80].

[80] To do this, we used a special type of function in NLPWin called tree_filter that allows filtering of sentences based on the presence of certain pre-defined contexts. In our case, we required that the sentences contain at least a clause where the main verb was *ser* or *estar*.

The number of sentences obtained through this method was 108K. We held out 10K sentences, picked at random, for evaluation purposes and used the remaining 98K as training data.

As could be expected, given the abundance of definitions and descriptions, there turned out to be a disproportionate number of sentences with *ser* in this type of text. We observed that the ratio of sentences with *ser* versus sentences with *estar* in Encarta was almost 5:1.

6.3.2.2 Technical Corpus

The technical corpus is a set of Microsoft manuals and technical documents that comprises around 340K sentences. The quality of the text is not homogeneus across the corpus, some parts being more carefully edited than others. The language is typical of the technical domain. Many of the sentences are instructions in imperative form like (71):

(71) Escriba la dirección de correo electrónico, nombre del equipo o la dirección de red de la persona a quien desea llamar.

To build our training corpus, we followed the same procedure used for Encarta and obtained, in this case, only 58K sentences containing forms of *ser* or *estar*. As with Encarta, we kept 10K sentences blind and used the remaining 48K as training data.

In this case, the ratio between *ser* and *estar* was found to be much lower, only 2:1. The differences in frequency of the copulas for the two types of corpus are shown graphically in Figure 66 and Figure 67.

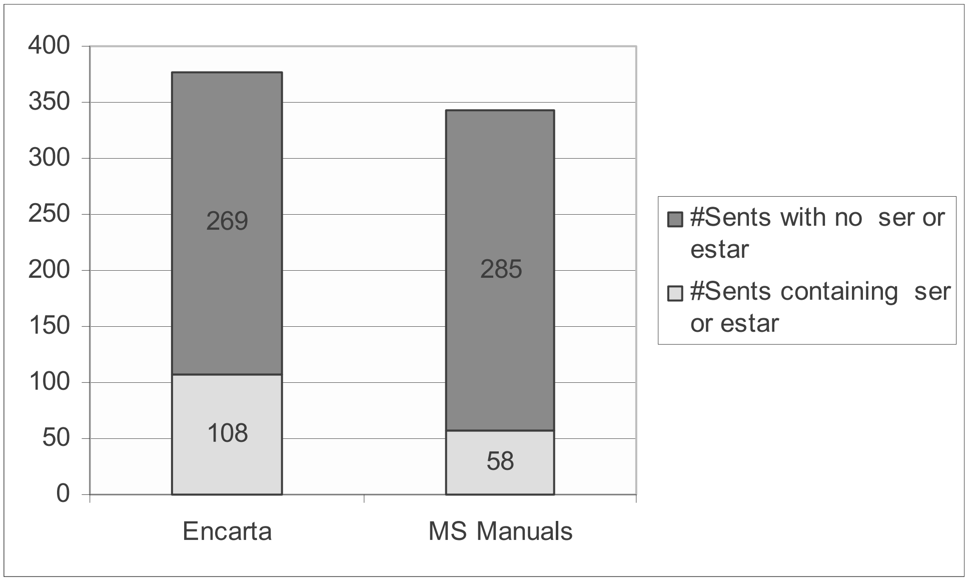


Figure 66: Number of sentences containing *ser* or *estar* relative to the total of sentences in each of the two corpora.

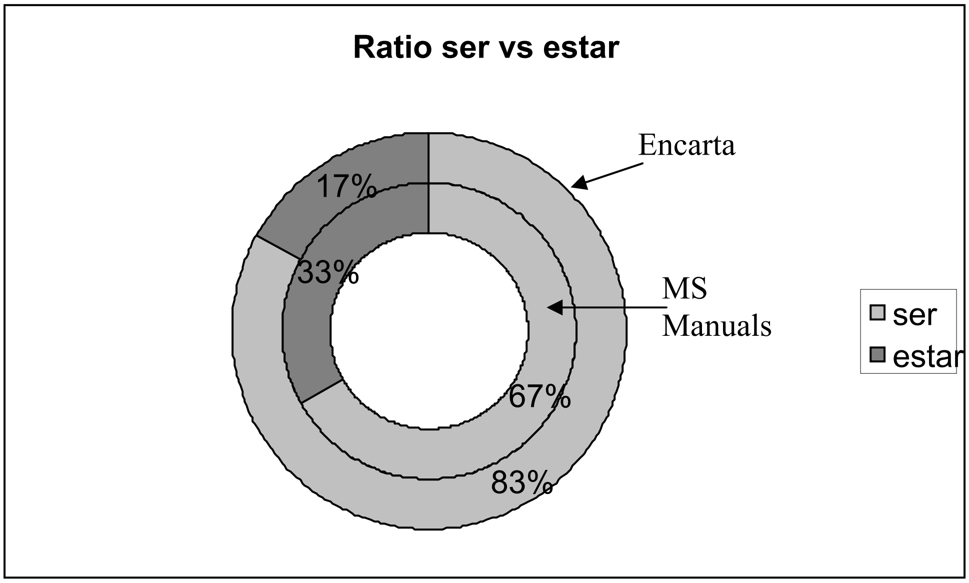


Figure 67: Disproportion between the percentage of sentences with *ser* vs. sentences with *estar* in each of the two corpora (data for Encarta appears in the outer circle and data for MS Manuals in the inside).

In view of this significant difference, we expect the models built on each type of text to show different properties.

6.3.3 Copulative Constructions in Logical Form

In Section 2.5 we gave an overview of the Logical Form in the NLPWin framework. Here we are going to review the representation that the copulative constructions get at this level of analysis. There are basically two types of copulative constructions in LF depending on the type of predicate:

i. Sentences with an adjective or noun phrase predicate (excluding pronouns and proper names), as in;

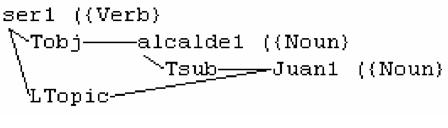
```
(72) Él es un hombre muy triste.
(73) El médico es el culpable
(74) Ya es primavera.
(75) Mi dormitorio es muy oscuro.
(76) Marta está alegre hoy.
```

ii. The rest, including pronouns and proper names, prepositional phrases, verbal phrases, etc., as in the following examples[81],

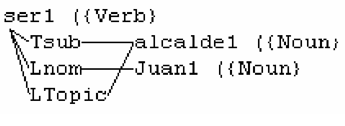
```
(77) Soy Juan
(78) No son estos.
(79) Este vestido es de seda.
(80) En Madrid estamos a 40 grados.
(81) ¿Dónde está tu abrigo?
```

The first type corresponds to constructions with a single argument, a Tobj, which is the AJP or NP predicate. All other arguments (including the Tsub) and modifiers are

[81] Let us point out that logically equivalent identity sentences, such as: "Juan es el alcalde" and "El alcalde es Juan", have different logical form representations:

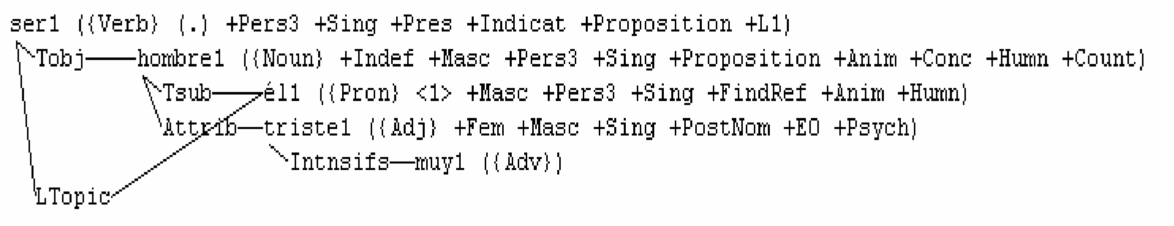


```
Juan es el alcalde
ser1 ({Verb}
 Tobj———alcalde1 ({Noun}
              Tsub———Juan1 ({Noun}
 LTopic
```



```
El alcalde es Juan.
ser1 ({Verb}
 Tsub———alcalde1 ({Noun}
 Lnom———Juan1 ({Noun}
 LTopic
```

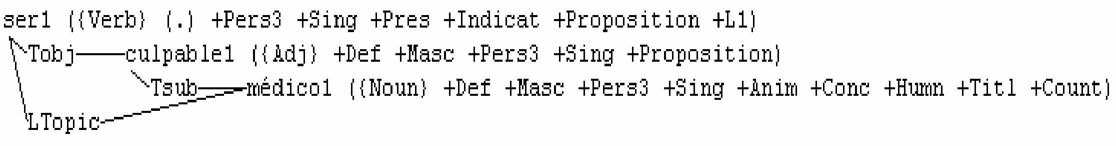
lowered onto the AJP or NP predicate. Figure 68 shows the LF corresponding to example (72); the noun *hombre* gathers as arguments both the syntactic subject (Tsub=*él*) and the modifying adjective (Attrib=*triste*).



```
ser1 ({Verb} (.) +Pers3 +Sing +Pres +Indicat +Proposition +L1)
 Tobj——hombre1 ({Noun} +Indef +Masc +Pers3 +Sing +Proposition +Anim +Conc +Humn +Count)
          Tsub——él1 ({Pron} <1> +Masc +Pers3 +Sing +FindRef +Anim +Humn)
          Attrib—triste1 ({Adj} +Fem +Masc +Sing +PostNom +E0 +Psych)
                  Intnsifs—muy1 ({Adv})
 LTopic
```

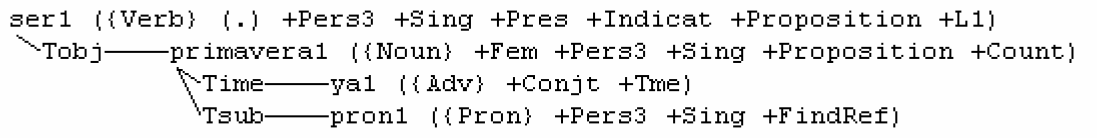
Figure 68: LF of *Él es un hombre muy triste*

The logical forms corresponding to examples from (73) to (76) are given below:



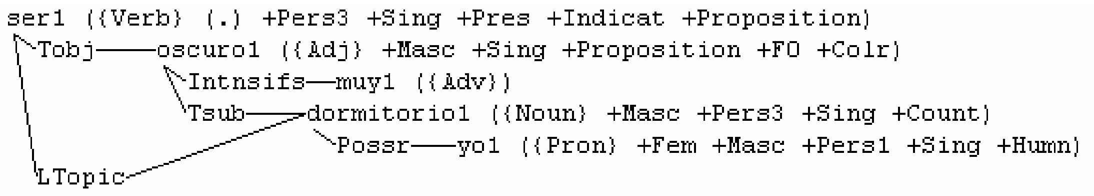
```
ser1 ({Verb} (.) +Pers3 +Sing +Pres +Indicat +Proposition +L1)
 Tobj——culpable1 ({Adj} +Def +Masc +Pers3 +Sing +Proposition)
          Tsub——médico1 ({Noun} +Def +Masc +Pers3 +Sing +Anim +Conc +Humn +Titl +Count)
 LTopic
```

Figure 69: LF of *El médico es el culpable*.



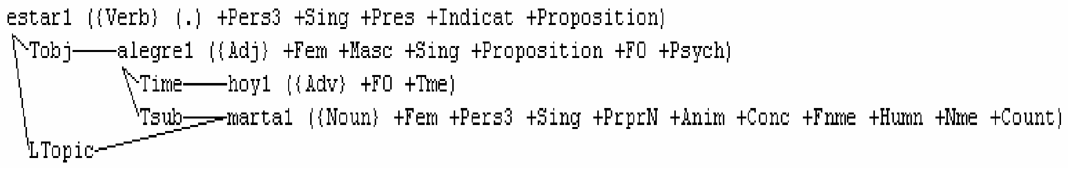
```
ser1 ({Verb} (.) +Pers3 +Sing +Pres +Indicat +Proposition +L1)
 Tobj——primavera1 ({Noun} +Fem +Pers3 +Sing +Proposition +Count)
          Time——ya1 ({Adv} +Conjt +Tme)
          Tsub——pron1 ({Pron} +Pers3 +Sing +FindRef)
```

Figure 70: LF of *Ya es primavera*.



```
ser1 ({Verb} (.) +Pers3 +Sing +Pres +Indicat +Proposition)
 Tobj——oscuro1 ({Adj} +Masc +Sing +Proposition +F0 +Colr)
          Intnsifs—muy1 ({Adv})
          Tsub——dormitorio1 ({Noun} +Masc +Pers3 +Sing +Count)
                  Possr——yo1 ({Pron} +Fem +Masc +Pers1 +Sing +Humn)
 LTopic
```

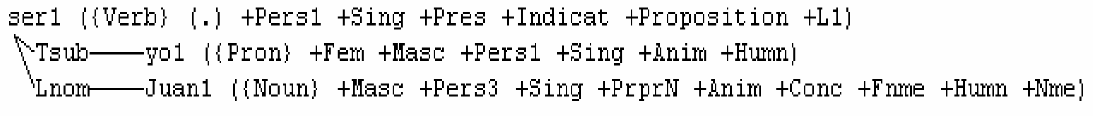
Figure 71: LF of *Mi dormitorio es muy oscuro.*



```
estar1 ({Verb} (.) +Pers3 +Sing +Pres +Indicat +Proposition)
 Tobj——alegre1 ({Adj} +Fem +Masc +Sing +Proposition +F0 +Psych)
          Time——hoy1 ({Adv} +F0 +Tme)
          Tsub——marta1 ({Noun} +Fem +Pers3 +Sing +PrprN +Anim +Conc +Fnme +Humn +Nme +Count)
 LTopic
```

Figure 72: LF of *Marta está alegre hoy*.

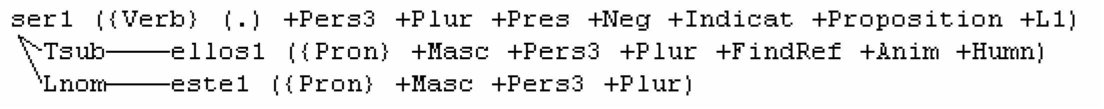
In the second type of copulative sentences, all the arguments and modifiers are left in place, as, for instance, the Lnom attribute, *Juan* as shown in Figure 73



```
ser1 ({Verb} (.) +Pers1 +Sing +Pres +Indicat +Proposition +L1)
 \Tsub———yo1 ({Pron} +Fem +Masc +Pers1 +Sing +Anim +Humn)
  \Lnom———Juan1 ({Noun} +Masc +Pers3 +Sing +PrprN +Anim +Conc +Fnme +Humn +Nme)
```

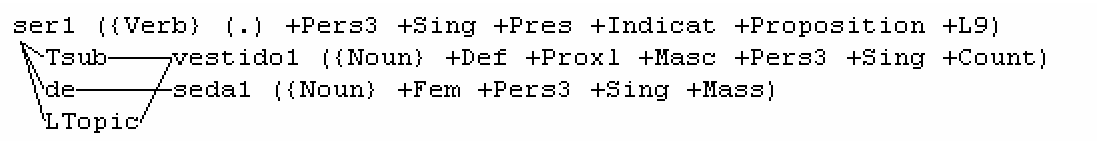
Figure 73: LF of *Soy Juan*.

Logical forms for examples from (78) to (81) are given below. In all these logical forms, the arguments and modifiers appear in an immediate dependency relation with respect to the copula.



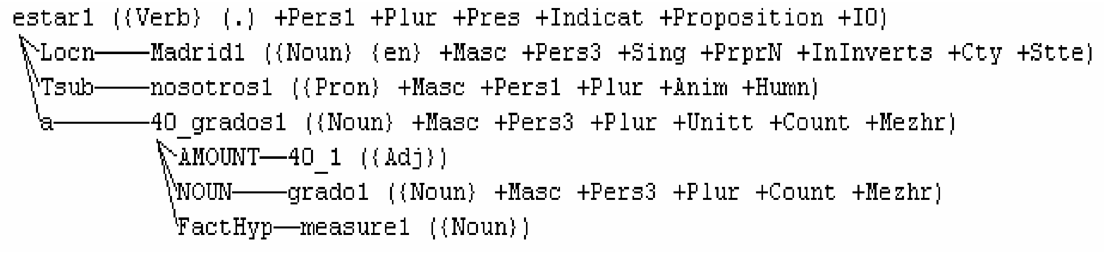
```
ser1 ({Verb} (.) +Pers3 +Plur +Pres +Neg +Indicat +Proposition +L1)
 \Tsub———ellos1 ({Pron} +Masc +Pers3 +Plur +FindRef +Anim +Humn)
  \Lnom———este1 ({Pron} +Masc +Pers3 +Plur)
```

Figure 74: LF of *No son estos*.



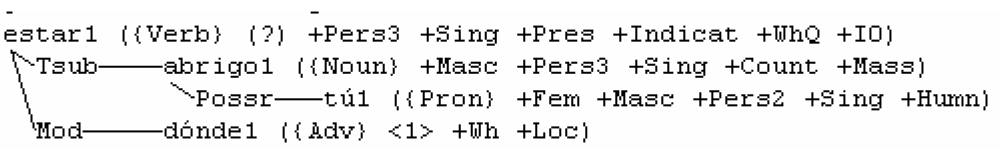
```
ser1 ({Verb} (.) +Pers3 +Sing +Pres +Indicat +Proposition +L9)
 \Tsub———/vestido1 ({Noun} +Def +Prox1 +Masc +Pers3 +Sing +Count)
 \de————/—seda1 ({Noun} +Fem +Pers3 +Sing +Mass)
  \LTopic/
```

Figure 75: LF of *Este vestido es de seda*.



```
estar1 ({Verb} (.) +Pers1 +Plur +Pres +Indicat +Proposition +IO)
 \Locn———Madrid1 ({Noun} (en) +Masc +Pers3 +Sing +PrprN +InInverts +Cty +Stte)
 \Tsub———nosotros1 ({Pron} +Masc +Pers1 +Plur +Anim +Humn)
  \a————40_grados1 ({Noun} +Masc +Pers3 +Plur +Unitt +Count +Mezhr)
          \AMOUNT—40_1 ({Adj})
          \NOUN———grado1 ({Noun} +Masc +Pers3 +Plur +Count +Mezhr)
           \FactHyp—measure1 ({Noun})
```

Figure 76: LF of *En Madrid estamos a 40 grados.*



```
-                   -
estar1 ({Verb} (?) +Pers3 +Sing +Pres +Indicat +WhQ +IO)
 \Tsub———abrigo1 ({Noun} +Masc +Pers3 +Sing +Count +Mass)
  |          \Possr———tú1 ({Pron} +Fem +Masc +Pers2 +Sing +Humn)
  \Mod————dónde1 ({Adv} <1> +Wh +Loc)
```

Figure 77: LF of *¿Dónde está tu abrigo?*

The representation where all arguments and modifiers are lowered onto the predicate (Figure 68 to Figure 72) is the first step to eventually get rid of the copula in the language-neutral representation of syntactic structure (see [Campbell and Suzuki, 2002] and footnote 48).

One consequence of having to deal with such an array of representations is that, when the training data file is created, we need to specify a larger number of positions in the logical form that need to be examined by the model, which increases exponentially the number of linguistic features or variables that the DT has to consider, as will become clearer in the next section.

6.3.4 Feature Selection

Maybe the most important task of the whole process is to devise a suitable set of features that might prove useful in predicting the target feature. This stage draws on knowledge of the linguistic phenomenon and on the peculiarities of its representation in LF.

Overall, there are two general strategies for feature selection:

i. Use linguistic expertise and an apriori knowledge of the phenomenon to carefully select each of the features relevant to the problem;

ii. Extract a huge number of features (e.g. all available bits in every node examined) and let the model building process determine which features have predictive value and which do not.

The first strategy appears more straightforward and efficient, but the second might allows us to learn new things, sometimes unexpected things, from the data.

We chose a mixed approach: we started by throwing in all the bits and attributes used by the Spanish LF and, on a first batch of experiments, we let the DT find the ones that had classificatory relevance. We did select manually which positions in the logical form had to be considered by the DT; those were:

- The node of the copula itself;
- The subject (Tsub);
- The predicate (Tobj, PrepRel or Mod).

As a second step, we inspected the resulting model with particular attention to the features that had a strong predictive value. Some of these features could be acting as *proxies* for our *target feature*[82]. For the particular needs of our experiment, we removed from the training all bits that could be giving away the lexical value of the copula, like e.g. subcategorization bits that might be different for *ser* and *estar*, or certain bits resulting from *smooshing* (see Section 2.3) a particular homographic form[83].

We repeated the experiment several times on different fractions of the training corpus. We observed that while the top features picked by the models were consistent across experiments, there was a considerable variability of the features that had less predictive value. For optimization purposes, in order to reduce the dimensions of the search space, we progressively removed bits that were clearly never used by the DTs.

We were also suspicious of bits that were very frequent in the data, and were actually selected by the DT but did not seem to have any classificatory relevance when inspecting the model. Two of such bits were Fem(inine) and Masc(uline). Since most nouns have gender, those bits occur very frequently on the training and are picked by the tree, but do not add any value to it. In fact, once removed from the training, the accuracy of the model improves slightly.

After the refinement process was finished, a total of 40 bits[84], plus 27 attributes[85], were used in the final version of the experiment. Those numbers combined with the different positions or nodes in the LF that were examined yielded a total of 673 variables or *features* that were extracted from each clause containing *ser* or *estar*[86].

What we call here a *feature* is the combination of a position or node in the LF structure we want the DT to consider, and a linguistic attribute or a bit that may or may not be present in this node. Thus, for instance: *Anim(Tsub)* is a feature that means

[82] The target feature in our case is the lexical value of the copula.

[83] For example, sentences with the copula *ser* in the past (*fue*) would carry the bit Mov(ement) (from the *ir* reading of *fue*).

[84] This is the complete list of bits that the DT checked: Allup, Anim, BdyPart, Conc, Colr, Comp, Completed, Condition, Condish, Conly, Continuous, Def, Derived, Futr, Humn, Imper, Indef, InInverts, Loc, Mannr, MarkedCap, MorC, Neg, Neut, Past, Pastpart, Perf, Plur, Pres, Proxl, PrprN, Prog, Proposition, Psych, Reflex, Resultat, Quant, Sing, TakesSubj and Tme.

[85] This is the list of attributes: Modals, CoCoords, FactHyp, Classifier, LOps, Tsub, Tobj, Tind, Intnsifs, LTopic, PrepRel, Purp, Manner, Possr, Means, Measure, Locn, Source, Goal, Time, Lnom, Lcmp, Attrib, Mod, SMods, Props and Appostn.

[86] Note that the actual search space is smaller because a number of features only have one value (e.g. 0). See Section 6.4.

“presence of the bit *Anim(ate)* in the (logical) subject (or Tsub[87])”; *Time(Tobj)* means “presence of a *Time* attribute in the (logical) object (or Tobj)”.

The features extracted belong to one of the following groups:

1. Attributes present in the node of the copulative verb;
2. Attributes and bits present in the Tsub;
3. Attributes and bits present in the Tobj;
4. Attributes and bits present in the PrepRel;
5. Syntactic category of the Tobj;
6. Lemma of the preposition in the PrepRel;
7. Lemma of the Tobj

The first four groups of features are binary, with 1 representing presence of the corresponding feature and 0 representing absence. The last three are categorial; they contain the actual value of the attribute: namely, the syntactic category (Noun, Adj, etc.) of the Tobj, the lemma of the Tobj, or the lexical value of the preposition in the PrepRel.

The *target feature*, i.e. the task that the decision tree has to solve, is defined as a Boolean variable called *is_estar*, with values *yes* and *no*, depending on the lexical value of the copula.

To extract the features that will be input to the DT, we used tree filters. A tree filter is a function[88] that is called on the root node of each parse (or logical form) at the end of the analysis process. Figure 78 shows the output of the filter after the sentence has been parsed. This is how the linguistic information is encoded for every *ser/estar* clause in the training data.

[87] As seen in Section 2.5 Tsub is a list; however, the DT is only looking at the first element of this list.

[88] In G language

```
   Sentence: [ que sea propietaria de una base de datos ]: Una cuenta que sea
propietaria de una base de datos siempre podrá abrir la base de datos.

   Values:"no",0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0
,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,"Adj","propietario",0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Figure 78: (Partial) Sample output of the tree-filter[89]

The list of values corresponds to the values of the variables (or features) for the clause between brackets (in the example, [*que sea propietaria de una base de datos*]). The first value ("no") corresponds to the target feature to be learned from this context, which in this case is "no" (i.e. *ser*); the rest of the values (0s, 1s, and the labels "Adj" and "propietario") correspond to each one of the features.

6.3.5 Using *WinMine* to Build the Models

To build the statistical models, we used the WinMine toolkit [Chickering., 2002], which is a set of tools for Windows 2000/NT/XP that allow building statistical models from data[90]. It has already been used in the NLP group at Microsoft Research to build a machine-learned sentence realization module called Amalgam [Corston-Oliver et al., 2002].

WinMine automatically splits the data 70/30 into training and validation sets, and then builds and evaluates decision trees. It builds the trees for various values of a factor called *kappa*, by manipulating the prior probability of tree structures to favor simpler ones. *Kappa* is a number between 0 and 1 used to determine the granularity of the decision tree; as *kappa* approaches 1, decision trees become more complex. If a model is

[89] In this figure we have only reproduced the first values of the list, not the complete list, which includes more 0's and a few more 1's.

[90] WinMine is freely available for research purposes and can be downloaded from http://research.microsoft.com/~dmax/WinMine/Tooldoc.htm . Further technical details of the tool can also be obtained in http://research.microsoft.com/~dmax/WinMine

less accurate, when measured against the 30% portion left for validation, than the previous model built, it means that we are beginning to overfit the data. Models built then with greater values of *kappa* will only be more overfit, so WinMine stops.

Prior to the building of the model, it is possible to manipulate the parameter that indicates the minimum number of data points where to split on a new branch, i.e. the minimum number of examples we require for the model to learn a classification[91]. To avoid overfitting the model to the training examples, the number *n* of branching should not be too small. The following values for *n* were tested: 5, 20, 30, 40, 50. For our task and the size of our training corpus, the best value for *n*, as measured on the resulting accuracy, turned out to be 30.

After the building and testing process finishes, WinMine produces an XML file of the model, which can be used to solve the task at hand.

6.3.6 Metrics Used to Evaluate the Performance of the Models

Together with the model, WinMine produces a summary or log file of the session with performance scores measured on the validation set. These scores comprise:

- Classification accuracy
- Baseline accuracy
- Precision and recall for each value of the target feature, plus F-measure

6.3.6.1 Classification Accuracy and Baseline Accuracy

Classification accuracy is the percentage of sentences that have been correctly classified. It is calculated by dividing the total number of correct classifications by the total number of classifications or cases.

$$Accuracy = \frac{Correct}{Classif}$$

[91] A branching value of n guarantees that a non-terminal node covers at least n cases. It does not guarantee that a leaf node covers n cases (e.g. we might specify 30, and end up with a leaf node covering twenty cases and another leaf node covering ten cases).

The overall accuracy is the most straightforward way to evaluate a model. To better rate the quality of the model it has to be put in relation to the baseline.

The baseline is the accuracy achieved by a trivial classification method or a by-default strategy. In our case, it represents the accuracy if the most frequent value would have been selected in all cases.[92]

6.3.6.2 Precision, Recall and F-measure

Overall accuracy is not sufficient for evaluating a model. It may also be interesting to see whether the model performs well for certain values of the target feature and badly for others. To determine this, we use precision, recall and F-measure for each value of the target feature.

The precision of a model with respect to a certain value X of the target feature (e.g. *estar*) records the success rate of the model on predicting X, with respect to the total number of cases that were classified by it as being X.

$$\Pr ecision(X) = \frac{Correct(X)}{Classif(X)} \times 100$$ [93]

Since the total number of classifications of X is the sum of the correct classifications of X plus the cases that were wrongly classified as X,

$$Classif(X) = Correct(X) + Wrong(X)$$

We have that precision of the DT for value X of the target feature is:

[92] This value is perhaps too rough to be considered a true baseline, but it helps put the accuracy numbers in perspective. A smarter baseline against which to compare the results of the experiment described here could be, as suggested by an anonymous reviewer, the accuracy achieved by an n-gram based approach in predicting the copula.

[93] Precision(X): Precision of the DT on predicting value X expressed in percent
Classif(X): total number of cases that were classified as value X
Correct(X): number of correct classifications of value X
Wrong(X): number of cases that were wrongly classified as X

$$\Pr ecision(X) = \frac{Correct(X)}{Correct(X) + Wrong(X)} \times 100$$

Recall of a model with respect to value X refers to the success rate of the model on predicting X when compared to the total number of cases that actually have value X in the data.

$$\operatorname{Re} call(X) = \frac{Correct(X)}{Total(X)} \times 100$$ [94]

Since the total number of cases that were actually X is the sum of the correct classifications of X plus the number of cases that were actually X but the DT failed to classify as such,

$$Total(X) = Correct(X) + Miss(X)$$

We have that recall of the DT for value X of the target feature is:

$$\operatorname{Re} call(X) = \frac{Correct(X)}{Correct(X) + Miss(X)} \times 100$$

When the target feature has only two possible values, X and Y (as in our case), then this equation holds

$$Wrong(Y) = Miss(X)$$

And we can reformulate recall and precision formulae in a way that shows that precision of one value is related, in a positive way, to recall of the other value. In other words, increasing the precision of value X will increase the recall of value Y.

[94] Recall(X): Recall of the DT on predicting value X in percent
Total(X): total number cases of value X in the test data.
Miss(X): number of cases that were value X and not classifed as such

$$\Pr ecision(X) = \frac{Correct(X)}{Correct(X) + Miss(Y)} \times 100$$

$$\operatorname{Re} call(X) = \frac{Correct(X)}{Correct(X) + Wrong(Y)}$$

It may be interesting to combine the numbers of precision and recall for a given value of the target feature; for this we use the F-measure. The F-measure is computed by calculating the harmonic mean of the two metrics.

$$F - measure(X) = \frac{2 \times \operatorname{Re} call(X) \times \Pr ecision(X)}{\operatorname{Re} call(X) + \Pr ecision(X)}$$

6.3.7 Inspection of the Models

The decision trees are XML files. They can be conveniently examined using DnetBrowser, which is an interactive tool, included in the WinMine toolkit, for viewing dependency networks and Bayesian networks. This tool shows the shape of the decision tree as well as the probability distributions at the leaf nodes.

In Figure 79, we show a screen snapshot of a decision tree being inspected using DnetBrowser. The non-terminal nodes of the tree are labeled with the name of the predictor that is used to distribute the examples at that level. The arcs or edges are labeled with the value of the predictor (1 or 0 if it is Boolean) as well as with the number of cases that have been used to learn that classification; the values of categorical predictors, such as part-of-speech or lemma, are distributed in a binary fashion too (e.g. lemma is equal to *disponible* or lemma is not equal to *disponible*).

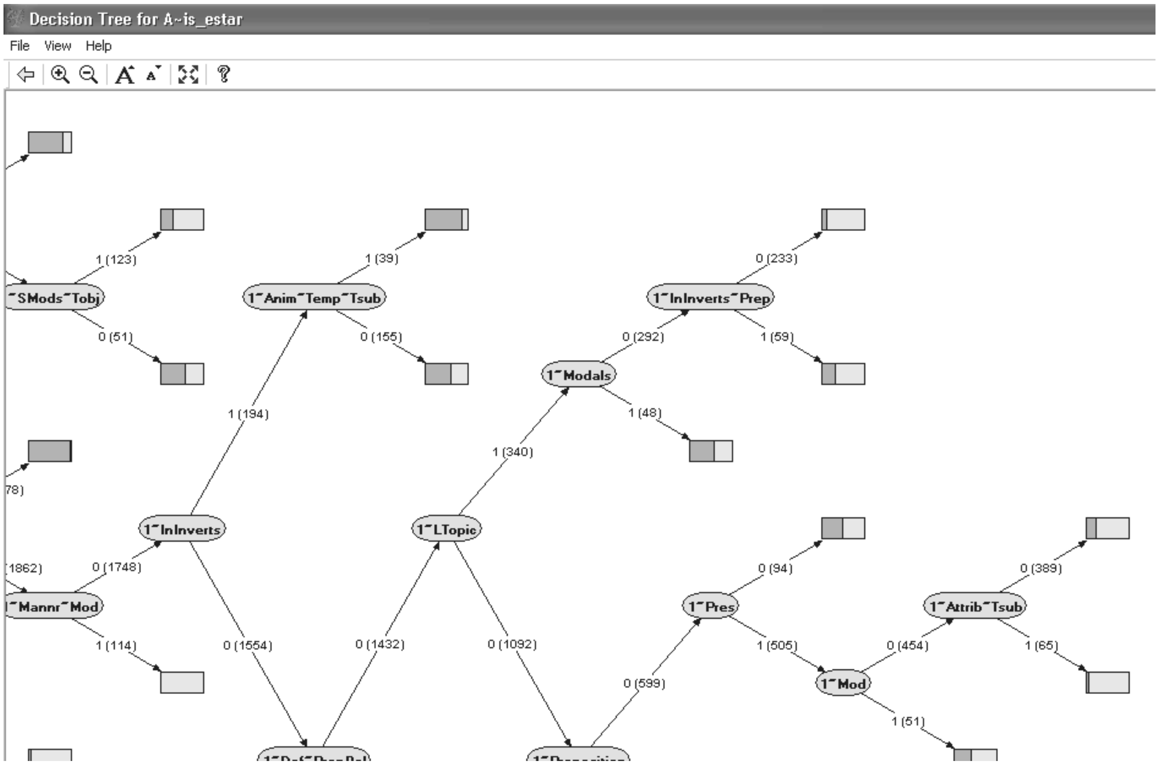
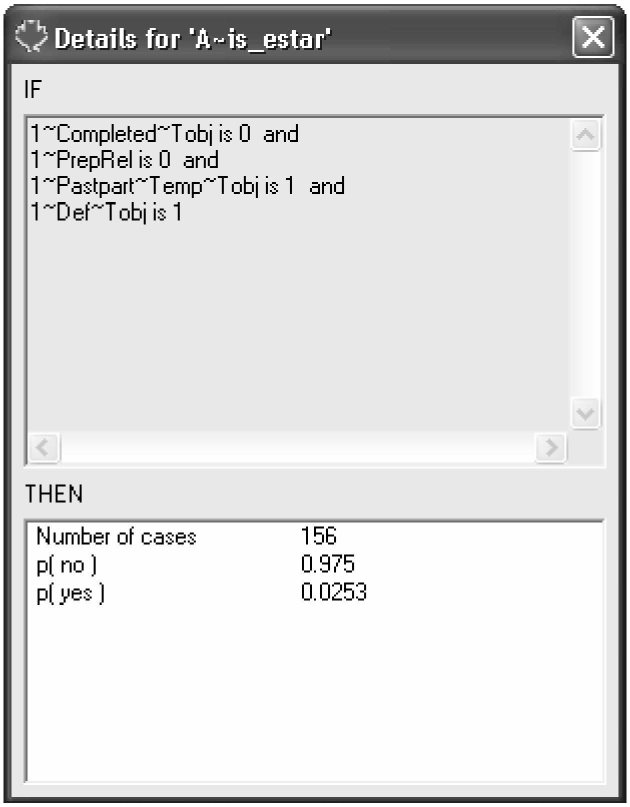


Figure 79: Using DnetBrowser to inspect a DT (fragment of a snapshot)

The colored boxes at the leaf nodes of the tree represent the probability distribution of the target feature; the dark area represents ‘yes’ (*estar* in our case) and the light grey area represents ‘no’ (*ser*). Double clicking on them, brings up a small dialogue box, like the one in Figure 80. This dialogue box provides information about the complete path from the root of the tree to this leaf node; the number of cases from the training data used for the last branching; and finally, the probability distribution for each value of the target feature.



1~Completed~Tobj is 0 and
1~PrepRel is 0 and
1~Pastpart~Temp~Tobj is 1 and
1~Def~Tobj is 1

```
Number of cases          156
p( no )                  0.975
p( yes )                 0.0253
```

Figure 80: Example of dialog box with probability distribution

6.4 Resulting DT Models for Predicting the Copula

6.4.1 Overall Accuracy of the Models[95]

The values for the overall accuracy of the model based on MS manuals and the model based on Encarta, as well as their complexity, as measured by the number of their branching nodes in the decision trees and the number of predictive features, are presented in Table 10 .

	Encarta	**MS manuals**
#Nodes	119	95
#Predictors	64	53

[95] As measured on the validation set.

Baseline	82.98%	67.53%
Accuracy	95.74%	97.15%

Table 10: Size, baseline and overall accuracy for each model

As mentioned before, the baseline is defined as the overall accuracy that would be obtained if the most frequent value (i.e. *ser*) would have been selected in all cases.

By looking at the numbers in Table 10, we observe that the accuracy is quite high in both models, all the more so in the model trained on MS manuals, especially when we put it in relation with the baseline, which is lower in this model than in the other. Indeed, by subtracting the baseline to the overall accuracy, we get an absolute improvement, or error reduction, of 29.62% for the model trained on MS manuals, while we only get 12.76% for the model trained on Encarta

If we put these numbers in relative terms, the accuracy of the model trained on the Manuals has improved 91.22% with respect to the baseline, while the accuracy of the model trained on Encarta has improved only 72.32%[96].

The complexity of the models also differs across domains, the model built on Encarta being more complex than the model built on MS manuals, both based on the number of predictive features and on the number of branching nodes. We hypothesize that the language from Encarta is richer and more varied than the language from MS manuals and therefore needs more rules. Actually, a quick browsing of the MS manuals shows a tendency to repeat the same patterns of sentences.

6.4.2 Which Model has Learned More?

By looking at the overall accuracy of the two models, it appears that their performance is quite good, especially in the case of the model trained on MS manuals, at least measured on the same type of text from which the model has learned. When we look

[96] B = baseline accuracy
A = accuracy of DT
A-B is the absolute improvement in accuracy (absolute reduction of error).
(A-B)/(100-B) is the relative improvement in accuracy (relative reduction of error).

closer to the precision and recall numbers for each of the two cases (*ser/estar*) an interesting fact becomes apparent.

As shown in Table 11, and more graphically in Figure 81, while the precision and recall of *ser* are similar, and equally high, in the two models, the values for *estar* differ considerably from one model to the other.

Precision for *estar* in the model trained on the MS manuals is almost as good as the precision for *ser*. Recall for *estar* in this model is also quite high.

However, the Encarta model shows significantly lower values both for precision and recall of *estar*. Recall is particularly low in this model.

	Precision (%)		**Recall** (%)		**F-measure** (%)	
	Encarta	**MS manuals**	**Encarta**	**MS manuals**	**Encarta**	**MS manuals**
Ser	96.58	97.66	98.34	98.13	97.45	97.93
Estar	91.13	96.07	83.03	95.10	86.89	95.58

Table 11: Precision and recall for *ser*/*estar*

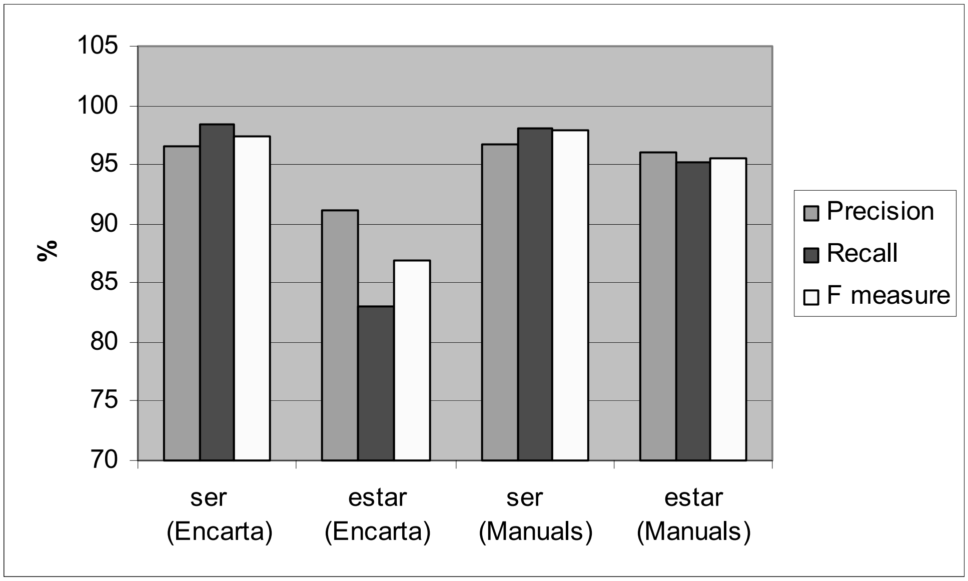


Figure 81: Precision and recall for *ser* and *estar*

Why does *estar* seem more difficult to learn from Encarta? We hypothesize that this may be attributable to the fact that the relative frequency of *ser* versus *estar* (see Sections 6.3.2.1 and 6.3.2.2) is much higher in the Encarta corpus than in the MS manuals corpus.

It would seem then that the more balanced the two values of the target feature are in the training data, the better the DT models perform. However, it should not be inferred from this that it is convenient to manipulate the training data by artificially reducing the disproportion between the two cases. Actually, by doing this we wouldn't be learning the actual distribution of the feature in the real data[97].

6.4.3 Predictive Linguistic Features

In the model that was trained on MS manuals, the number of predictors, or linguistic features that are predictive of the target feature, selected by the decision tree algorithm was 53 out of the original 673. The model that learned from the text based on Encarta chose 64 predictors out of the total 653 features considered. From these predictors, 33 appear in both models. This overlap in selected features indicates that decision trees indeed capture linguistic generalizations that are valid across domains, especially considering that most overlapping predictors are the ones that are ranked highest with respect to their classificatory relevance.

On the other hand, 20 predictors appear only in the MS manuals models and 31 are exclusive of the Encarta model. This fact would indicate that the models have some degree of domain specificity as well. By looking at the predictors that appear in one model and not in the other we can get a fair idea of the frequency of certain phenomena in one type of text or the other. For example, the presence of a modal operator is a predictor in the MS manuals model, but not in Encarta. Actually, in text coming from manuals we find many sentences containing *deber* or *poder*, such as:

[97] Actually, in a side experiment we manipulated the corpus from Encarta reproducing the size and distribution of the MS manuals corpus: 48K and 2:1 proportion. The model trained on the manipulated corpus had higher precision for *estar* (94.15% vs. 91.13 in the original Encarta) but a sensibly lower precision for *ser* (92.94% vs. 96.58%). As a consequence the overall accuracy was 2.47 points lower (93.27% vs. 95.74%)

(82) Un usuario agregado para un teléfono o una línea debe estar en el mismo dominio.

(83) Puede ser que el servidor de acceso remoto no esté en funcionamiento.

On the other hand, among the Encarta predictors we find features such as: Colr(Tobj), i.e. the logical object is a name of a color, or BdyPart(PrepRel), i.e. the prepositional complement is marked as being a body part[98]; or the fact that the logical subject is a proper name.

Table 12 presents all the predictors selected by both models, or rather the linguistic information they represent, sorted for the four different nodes in the LF that were inspected by the DT. It is not surprising that more information has been found to be predictive in the Tobj node than in the other nodes, if we consider that for many copulative sentences the arguments are lowered onto that node, as seen in Section 6.3.3[99].

Copula	**Tsub**	**Tobj**	**PrepRel**	**Mod**
Has nominal predicate (Lnom)	**Is Plural**	**Is Past participle**	**Value of Lemma of the preposition**	Is Manner
Has a topic[100]	**Is Singular**	**Value of Lemma**	**Is a topicalized sentential modifier (InInverts)**	Is Singular
Is a Proposition	**Is Animate**	**Is a verb or adj that takes subjunctive[101] (TakesSubj)**	**Is Definite**	Is Concrete
Is Plural	**Is Definite**	**Value of syntactic**	Is Singular	*Is Coordinated*

[98] Most examples classified using this predictor are of the type: El poder ejecutivo está *en manos* de el senado or El patriarca de Antioquía está *a la cabeza* de una importante comunidad de árabes cristianos en Siria.

[99] Some predictors linguistically redundant have been simplified, e.g.: Complete(Tobj) (bit assigned to past participles) and Pastpart(Tobj) have been merged into: Tobj "is Past participle". Or Anim(Tsub(Tobj)) and Anim(Tsub) into: Tsub "is Animate".

[100] The attribute LTopic contains whichever argument (Tsub or Tobj) is topicalized, i.e. pre-verbal.

[101] That is, that subcategorizes for a complement clause in Subjunctive, such as: Es *posible* que haya un error de disco

		Category		
Is Present	**Has an Attrib**[102]	**Has a Tsub**	Is Concrete	*Is Location*
Has a Modifier (Mod)	Is Present	**Is Definite**	*Is a Body Part*	
Has a Time modifier	Is Quantified (Quant)[103]	**Has an Intensifier (e.g. *muy*)**	*Has an Attrib*	
Is Singular	Has a Tsub	**Has a sentence level modifier (SMods)**	*Is morphologically Derived*	
Is a topicalized clause[104]	Has a PrepRel	**Is a Comparative phrase**[105]	*Has a quantifier (LOps)*	
Is a Condition	*Is Human*	**Has a PrepRel**	*Has a PrepRel*	
Has Modals	*Has a Possessor*	**Is Coordinated**	*Is a Proposition*	
Has a Prepositional compl (PrepRel)	*Is a Proper name*	**Is Plural**	*Is a Proper name*	
Has a Location compl (Locn)		**Has a Time mod**	*Has a Tsub*	
Is Coordinated		Is morphologically Derived[106]		
Is Past		Is Singular		
Has a Props		Is Indefinite		
		Has an Attrib		
		Is Negated		
		Is topicalized (InInverts)		
		Has a Location compl (Locn)		
		Has a Modifier		
		Has a Classifier		
		Is a Color		
		Has a Modifier		
		Is semantically marked as Psychological[107]		

[102] I.e. is modified by an adjective, past participle or a relative clause

[103] Example: *Todos los demás elementos* son ***True***.

[104] Such as: *Si está en los Estados Unidos*, póngase en contacto con QualiType

[105] E.g. más + adj

[106] That is, a word to which a morphological derivational rule has applied, e.g. adverbs ending in –mente, nouns derived from verbs, etc..

		Is Quantified		
		Is Reflexive[108]		

Table 12: Complete list of predictors

In Table 12, we have followed the following conventions:

- Predictors that have been selected by both models are in bold.
- Predictors that have been selected only by the MS manuals model are left unmarked.
- Predictors that have been selected by the Encarta model are in italics.
- Predictors that rank higher in the tree (i.e. supposedly the ones with more predictive value) are highlighted in grey.

The highest ranked predictors are also the ones that remain more stable across the different experiments, when using different data or applying variations to the initial set of bits. The other predictors –the ones with a lower ranking- are much less stable, as is usually the case with this type of classification techniques, which are very sensible to the training data[109].

The only two features in Table 12 that are among the first ten predictors of their models and yet are not common to both models are: the logical subject has present tense (for MS manuals) and the logical object is topicalized (in Encarta). Both are the consequence of some particularities in the analysis:

In the first case, the Spanish analysis inserts an infinitival *ser* in the logical form of the construction *puede que*, analyzing the complement clause as the logical subject of *ser*.

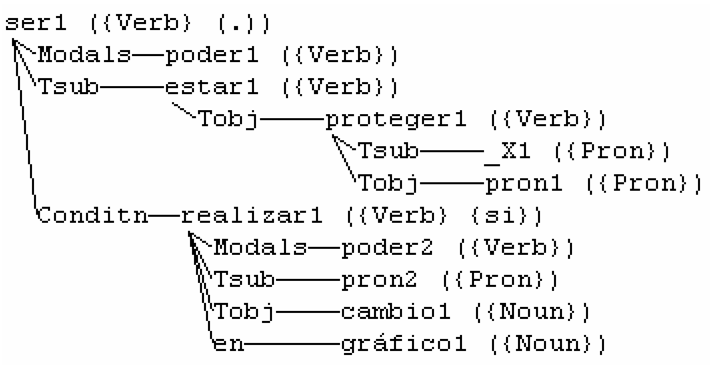
(84) Si no puede realizar cambios en un gráfico, puede que esté protegido.

[107] A type of verb that usually denotes a mental process. It is constructed as: NPdat + V + NP. Importantly, the passive of these verbs with ser and estar is analyzed as Copula + Adj. Examples: *El estilo artístico de Schulz es admirado por ser limpio y ordenado*; *Estaba* pron *fascinado por los reflejos en el agua*.

[108] Reflexive/pronominal verbs

[109] This fact is conveniently illustrated by a sideline experiment in which we built two new models using as training sets two equal-size, randomly generated halves of the MS manuals corpus. Two observations emerged from it: i) the accuracy did not suffer from dividing the training data by half; and ii) the lower portion of the ranked list of predictors was not consistent across the models built on the initial MS manuals corpus and on its two halves.

Figure 82 shows the logical form of example (84), where a verb *ser* has been inserted [*puede* **ser** *que esté protegido*] and the complement clause has been analized as Tsub.

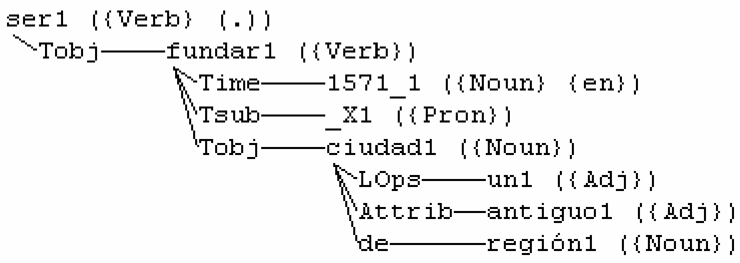


```
ser1 ({Verb} (.))
 \Modals——poder1 ({Verb})
 \Tsub————estar1 ({Verb})
            \Tobj————proteger1 ({Verb})
                       \Tsub————_X1 ({Pron})
                       \Tobj————pron1 ({Pron})
 \Conditn——realizar1 ({Verb} {si})
             \Modals——poder2 ({Verb})
             \Tsub————pron2 ({Pron})
             \Tobj————cambio1 ({Noun})
             \en——————gráfico1 ({Noun})
```

Figure 82: Logical form for example (84)

As for the case of the topicalized object, many sentences with extraposed absolute past participle are analyzed by the grammar as a Tobj, which is marked InInverts. This is a case of the model picking up a generalization over a systematic case of misanalysis. (85) is an example of this type of sentence and the resulting logical form is shown in Figure 83.

(85) Fundada en 1571, es una de las ciudades más antiguas de la región.



```
ser1 ({Verb} (.))
 \Tobj————fundar1 ({Verb})
            \Time————1571_1 ({Noun} {en})
            \Tsub————_X1 ({Pron})
            \Tobj————ciudad1 ({Noun})
                       \LOps————un1 ({Adj})
                       \Attrib——antiguo1 ({Adj})
                       \de——————región1 ({Noun})
```

Figure 83: Logical form of example (85)

6.4.4 Linguistic Analysis of the Models

6.4.4.1 Linguistic Significance of the Main Predictors

Among the highest ranked predictors common to both models, we find certain expected features, such as:

- Predicate is a past participle
- Predicate adjective belongs to a specific set of lemmas
- Prepositional predicate has a certain preposition
- Syntactic category of the predicate
- Copula has a nominal predicate (Lnom)

However, also among the important predictors, we find other features that are quite unexpected, such as:

- Presence of an intensifier, classifier or operator in the predicate
- The predicate is a verb or an adjective that subcategorizes for a clause in subjunctive
- Predicate (Tobj) has a logical subject
- Predicate is definite
- Prepositional predicate is inverted

An analysis of the examples classified using these predictors help us discover their relevance:

- The presence of an intensifier is used in both models to further classify the cases that have a past participial clause as predicate. As it turns out, it helps to identify cases of *ser* with adjectives that are morphological participles, such as:

(86) **Es** demasiado complicado de mantener.

- The presence of the bit TakesSubj (verb or adjective that subcategorizes for a subjunctive clause) helps identify a set of adjectives that typically take ser: *posible, necesario, imprescindible*, etc...

(87) **Es** posible que ese gráfico esté dañado.

- The fact that the Tobj has a logical subject (or Tsub) is related to the particular representation of copulatives in NLPWin. As explained in Section 6.3.3., sentences with an adjective or noun phrase predicate take only one argument (Tobj) onto which the rest of complements are lowered. Most of these sentences select *ser* in our data.

(88) El BIOS **es** invisible a los usuarios de los equipos.

- A definite predicate implies a nominalization of the predicate, even if the syntactic category is an adjective. In those cases *ser* is the most frequent choice:

(89) La técnica de el carbono radioactivo tal vez **sea** la más utilizada.

- Perhaps the least expected of all the main predictors is the presence of an extraposed prepositional complement. Sometimes the prepositional complement is the predicate (as in (90)), and sometimes it is a sentence modifier, as in (91) and (92):

(90) De igual importancia son el Museo de Artes y Ciencias Aplicadas y el Museo Nicholson de antigüedades.

(91) De las ocho especies que constituyen el género, una es común en las Antillas.

(92) En Microsoft Excel, el argumento Index no es opcional.

As it is the case with the other features found to have predictive value, this predictor cannot be considered relevant in isolation, but in the context of the other predictors that help build the tree. In the examples coming from MS Manuals, this predictor is invoked in the following path from the root of the decision tree to the leaf node:

```
Completed(Tobj)=0  &  Pred(Tobj)^="disponible" &
PrpCnjLem(PrepRel)="en" & InInverts(PrepRel)=1
```

By looking at the examples, we discover that InInverts(PrepRel) is in fact used to disambiguate between cases where the prepositional complement, with preposition *en* is the predicate (i.e. less prone to be inverted, most often selecting *estar*), as in (93), and cases where the prepositional complement, with preposition *en*, is not the predicate as in (92).

(93) Los datos que desea utilizar están <u>en un archivo de texto</u>.

6.4.4.2 A Special Feature: the Lemma of the Predicate

The problem of selecting the copula may seem, a priori, to be lexically determined, at least in the case of the <copula+Adj> constructions. For this reason, it is interesting to look at the lemmas that the DT has selected as being predictive. We expect to find adjectives that have a strong preference for one copula or the other. The first interesting thing that we notice is the small number of lemmas that have been identified as being predictive: 15 adjectives in the case of the model built on MS Manuals and only 6 adjectives plus 5 verbs[110], in the case of the model built on Encarta. Figure 84 combines all the lemmas that have been actually learned by the two models (verbs are given in their participial form).

Lemma	MS Manuals model	Encarta model
abierto	X	
activo	X	
apreciado		X
compatible	X	
conocido		X

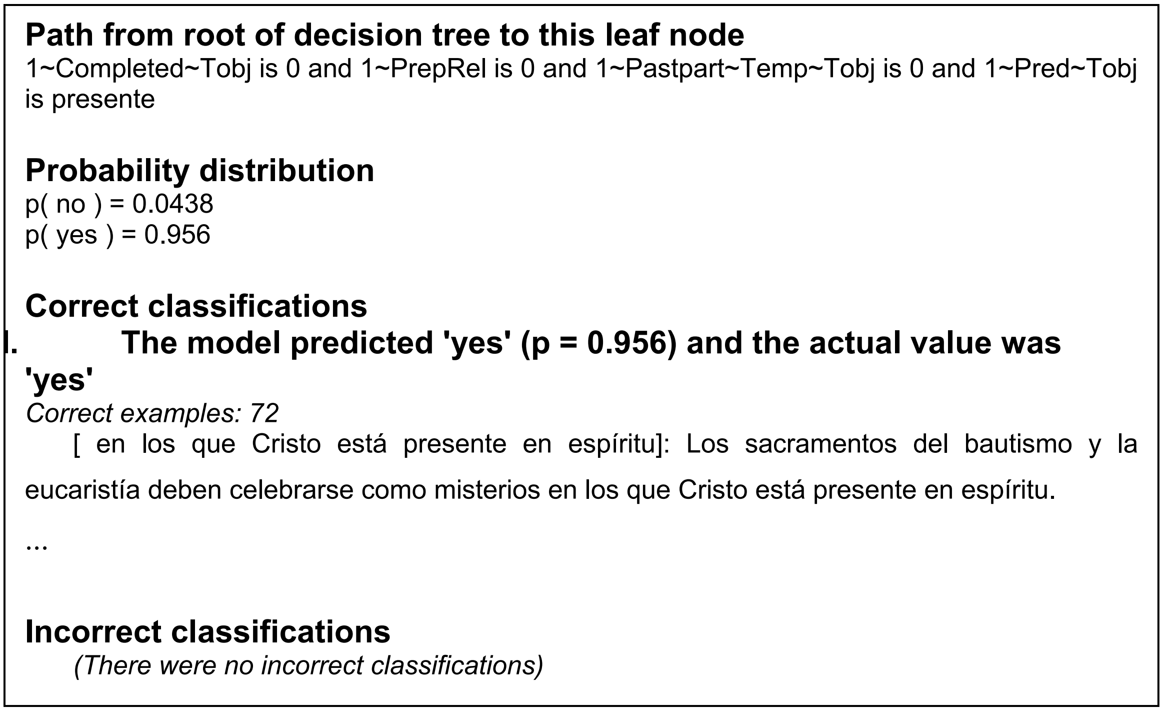
[110] In some cases, morphological participles are analyzed as verbs.

claro		X
desarrollado		X
determinado	X	X
disponible	X	
extendido		X
inactivo	X	
libre		X
listo	X	
lleno	X	
oculto	X	
ordenado	X	
parecido		X
presente	X	X
seguro	X	
sujeto	X	X
vacío	X	
variado		X
visible	X	

Figure 84: Lemmas identified by the models

The two types of text being so different, it is not surprising the limited overlapping of lemmas that we find in this list.

In a closer examination of the models, we see that certain lemmas have a strong influence on the decision, as, illustrated by the path from the Encarta model shown in Figure 85. Here the decision is strongly based on the lemma of the predicate being *presente*:



Path from root of decision tree to this leaf node
1~Completed~Tobj is 0 and 1~PrepRel is 0 and 1~Pastpart~Temp~Tobj is 0 and 1~Pred~Tobj is presente

Probability distribution
p(no) = 0.0438
p(yes) = 0.956

Correct classifications

l. The model predicted 'yes' (p = 0.956) and the actual value was 'yes'

Correct examples: 72

[en los que Cristo está presente en espíritu]: Los sacramentos del bautismo y la eucaristía deben celebrarse como misterios en los que Cristo está presente en espíritu.

...

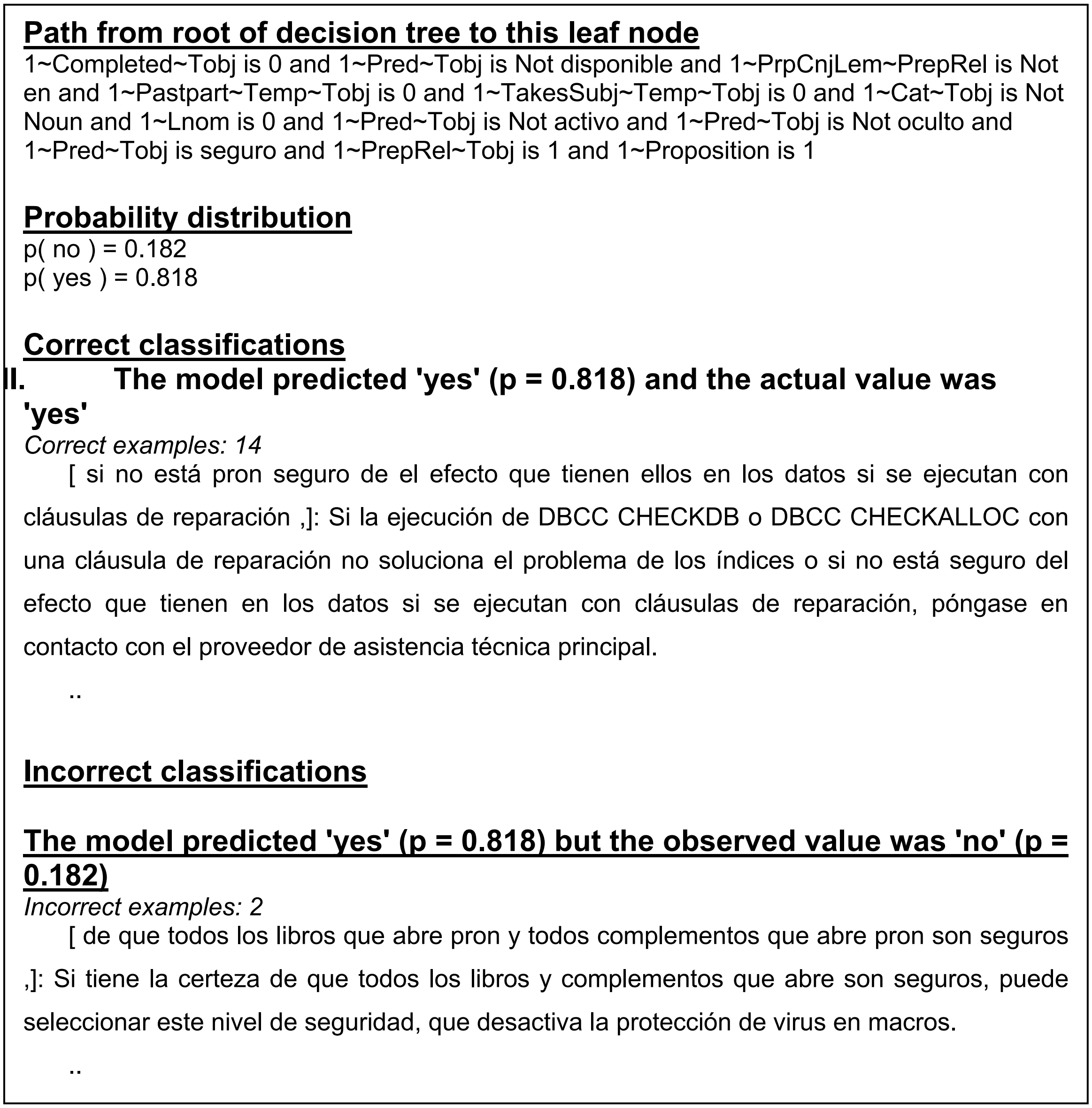
Incorrect classifications
(There were no incorrect classifications)

Figure 85: Branch containing lemma *presente*[111]

As we can see, the decision path is very short (only 4 nodes) and the probability of the final decision is very high (0.956).

By contrast, in Figure 86, we can see a branch with a longer path, where the lemma *seguro* plays a less definite role.

[111] The format of the nodes of the path from the root to the leaf node, as provided by the tool, follows a convention slightly different from the one adopted in this thesis: the prefix 1~ stands for the initial node (i.e. the node of the copula) and can be ignored. Thus, 1~Completed~Tobj is equivalent to Completed(Tobj). The attribute Temp is where the lexical bits are stored, and we have ignored it in our notation; thus 1~Pastpart~Temp~Tobj would read simply as Pastpart(Tobj). Moreover, "is 1" and "is 0" means respectively that the feature is or is not present.



Path from root of decision tree to this leaf node

1~Completed~Tobj is 0 and 1~Pred~Tobj is Not disponible and 1~PrpCnjLem~PrepRel is Not en and 1~Pastpart~Temp~Tobj is 0 and 1~TakesSubj~Temp~Tobj is 0 and 1~Cat~Tobj is Not Noun and 1~Lnom is 0 and 1~Pred~Tobj is Not activo and 1~Pred~Tobj is Not oculto and 1~Pred~Tobj is seguro and 1~PrepRel~Tobj is 1 and 1~Proposition is 1

Probability distribution

p(no) = 0.182
p(yes) = 0.818

Correct classifications

II. The model predicted 'yes' (p = 0.818) and the actual value was 'yes'

Correct examples: 14

[si no está pron seguro de el efecto que tienen ellos en los datos si se ejecutan con cláusulas de reparación ,]: Si la ejecución de DBCC CHECKDB o DBCC CHECKALLOC con una cláusula de reparación no soluciona el problema de los índices o si no está seguro del efecto que tienen en los datos si se ejecutan con cláusulas de reparación, póngase en contacto con el proveedor de asistencia técnica principal.

..

Incorrect classifications

The model predicted 'yes' (p = 0.818) but the observed value was 'no' (p = 0.182)

Incorrect examples: 2

[de que todos los libros que abre pron y todos complementos que abre pron son seguros ,]: Si tiene la certeza de que todos los libros y complementos que abre son seguros, puede seleccionar este nivel de seguridad, que desactiva la protección de virus en macros.

..

Figure 86: Branch containing the lemma *seguro*

The presence of the prepositional complement (PrepRel(Tobj)=1), "seguro+de" helps tilt the decision towards the 'yes' value (i.e. *estar*). In fact, the 2 examples of incorrect classifications are the result of wrong analysis (the "de que" clause is analyzed as a relative clause instead of a clausal complement).

It is important to consider that we are not using any kind of selectional information for adjectives in the dictionary that would be encoded specifically for our problem (i.e. a feature that tells whether an adjective selects *ser* or *estar*). This seems like a relevant fact, especially if we consider that the <copula + AJP> constructions are among the hardest to predict. For these reasons, it is reasonable to think that the information provided by the lemmas is comparatively important and has a sizeable impact on the resulting model, even though the actual number of lemmas that is being used is so small.

To test this hypothesis, we build new models without using the information of the lemma of the Tobj.

	Encarta (using lemma)	**Encarta** (not using lemma)	**MS Manuals** (using lemma)	**MS Manuals** (not using lemma)
Complexity	119	126	95	144
Predictors	64	60	53	73
Accuracy	95.74%	95.14%	97.15%	91.02%

Table 13: Complexity, number of predictors and accuracy

As the numbers in Table 13 and –more graphically- the chart in Figure 87 show, the accuracy of the model built on MS manuals drops considerably when the information about the lemma of the adjective is not used.

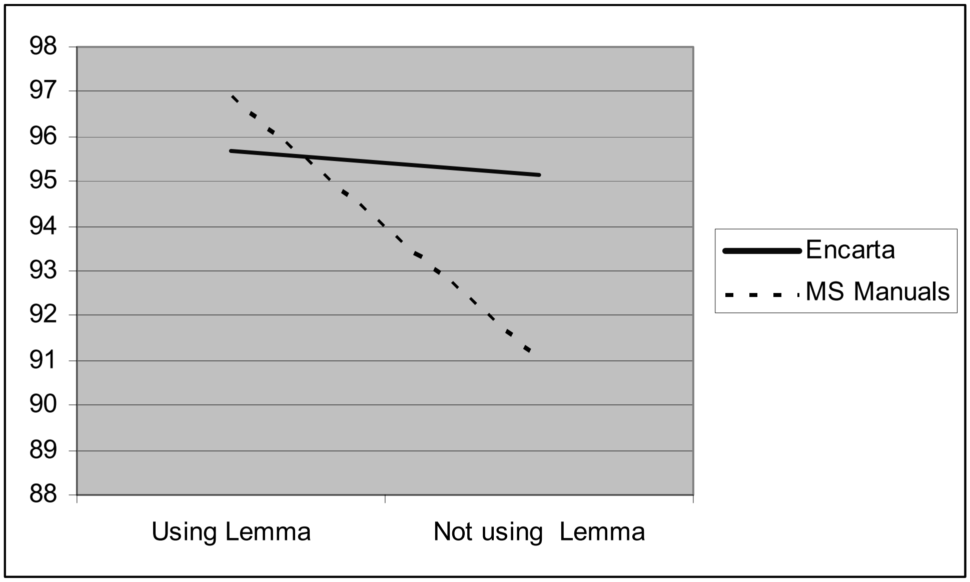


Figure 87: Comparison of overall accuracy of the models with and without lemma

The accuracy of the Encarta based model, on the contrary, appears to be quite independent from this type of information. One can hypothesize that constructions of the type <copula + AJP> are more frequent, relative to the total number of cases of copulative sentences in the MS manuals corpus[112]. A sharp increase of complexity, as well as of the number of predictors, is also observed in the model based on MS manuals. This seems reasonable since the classifier has to make up for the missing predictor by using more features.

6.4.5 Evaluation of the Models on a Blind Set: Methodology and Results

Even though our main interest was to use the result of this experiment in an application environment such as MT, we used Spanish texts for evaluation purposes. It may seem that evaluating the results using Spanish data constitutes an artificial environment: after all, we are generating Spanish sentences from structures resulting from the analysis of the same Spanish sentences. Nonetheless, this enables us to perform an automatic evaluation, since we can compare our results against the lexical value of the copula in the original sentence.

We took the two blind sets of 10K sentences that had been left aside when building the training corpora, one from MS manuals and the other from Encarta. Using NLPWin, we analyzed and regenerated the Spanish sentences (with the right copula in them) and then created a master file with the results. We then ran a regression testing against the master file by removing the copula and recalculating it using the model. The number of differences should be equal to the number of regressions. The result of the evaluation is shown in Table 14:

[112] Or rather constructions of the type <*estar* + AJP> taking into consideration the higher proportion of *estar* in this type of text.

	Encarta model	**MS Manuals model**
Encarta text	554/10K (94.46%)	702/10K (92.98%)
MS manuals text	1087/10K (89.13%)	379/10K (96.21%)

Table 14: Same text-type and cross-type evaluation of the two models

From these results we observe that there is an expected correlation between the type of text and the type of model: the model based on Encarta is the best model for the Encarta text and the model trained on MS manuals is the best model for that type of technical text[113]. The problem of selecting the right copula seems to be affected by domain-specific regularities.

Interestingly, the model trained on technical data also performs quite well on text coming form Encarta. Indeed, the value of the copula seems "easier" to predict in this type of text. On the other hand, Encarta performs poorly in the text coming from MS manuals. The explanation for this may be found in the numbers for precision and recall for each value that are shown in Figure 81 together with the observed disproportion of cases of *ser* in Encarta (5:1). In other words, the model based on Encarta is much worse classifying cases of *estar* than cases of *ser*, but this fact becomes more apparent when tested on text from Manuals, since in that type of text the proportion of *estar* is more than the double than the proportion of *estar* in text from Encarta.

6.4.6 Mixed Model

As a last experiment we prepared a mixed corpus, composed of 50% text coming from MS manuals and 50% coming from Encarta. The half from MS manuals was identical to the corpus used for building our MS manuals model, i.e. 48K sentences; the half corresponding to Encarta were 50K sentences picked randomly from the corpus used to train the Encarta model. The total was a set of 98K sentences from both domains.

[113] The accuracy numbers for same type of text evaluation are expectedly close to the ones obtained through WinMine. The slight difference may be explained in part because of misanalysis of a number of evaluated sentences.

Not surprisingly the features selected by the mixed trained model contain the most predictive features of the model trained on Encarta and the model trained on MS manuals. Also perhaps not surprisingly, the resulting values for accuracy, precision and recall were somewhere in the middle of the values of the two previous models, as shown in the graphic in Figure 88.

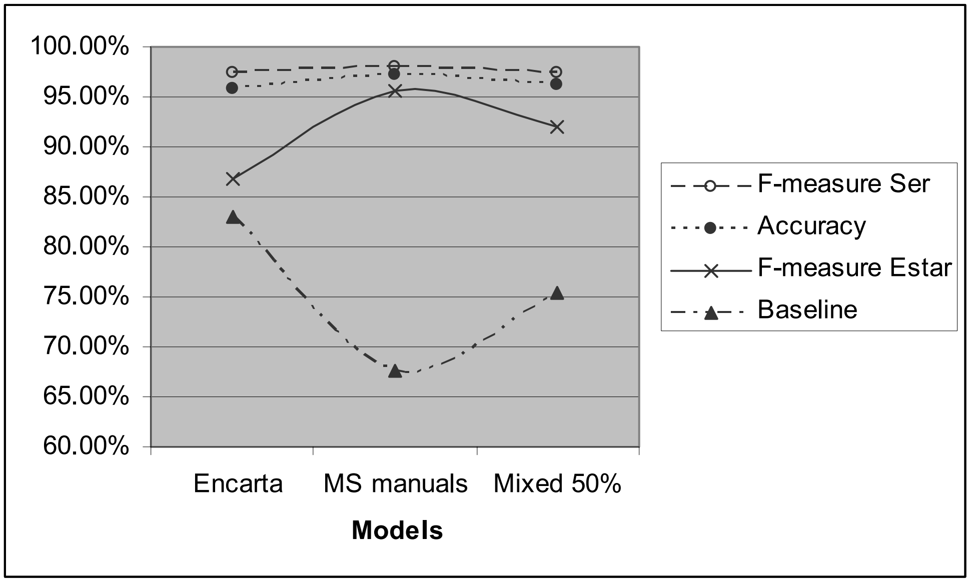


Figure 88: Comparison of baseline, accuracy and F-measures

To verify whether the mixed model was more portable than the two base models across domains, we repeated the evaluation described in Section 6.4.6 for the new mixed model. The results in Figure 89 show that indeed the mixed model is more independent from text domain than the original models.

	Mixed model
Encarta text	585/10K (94.15%)
MS manuals text	380/10K (96.20%)

Figure 89: Evaluation of the mixed model on the two types of text

As Figure 90 graphically shows, the accuracy of the mixed model on each type of text is comparable to the accuracy achieved by the model trained solely on that type of text. Consequently, the mixed model is the most portable of our three models. This result suggests that the more varied our training data from the point of view of type of text, the better. In the particular case of our linguistic problem, mixing the two types of text helps compensate the disproportion between *ser* and *estar* in Encarta.

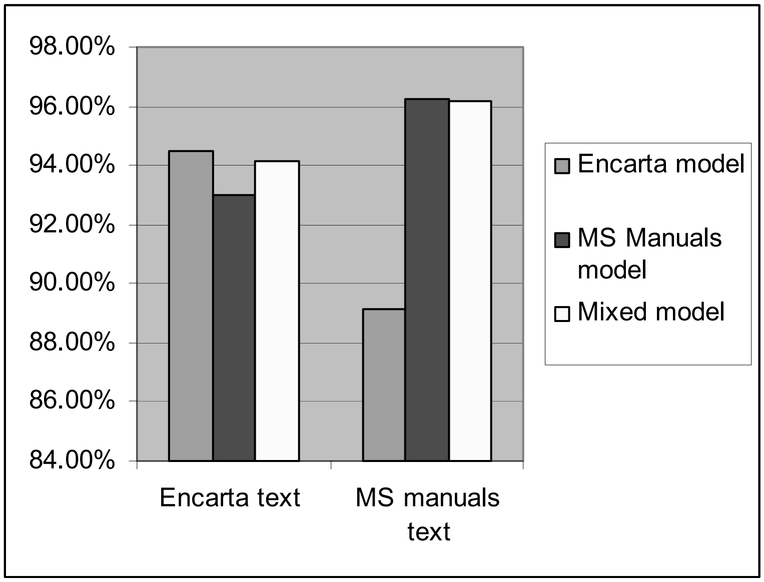


Figure 90: Comparative performance of the two models plus the mixed model on the two types of text

6.4.7 Integration of the DT in the Generation grammar

To perform the evaluation described in the previous sections, we had to integrate the models into our Generation rules. This is done in a straightforward way. The Generation rule that predicts the lemma of the copula calls the DT model by invoking a function that returns a Boolean value. This function takes as parameters the DT model we are using, the target feature we are trying to predict (*estar* in our case), and the LF node we are considering (in our case, the node of the copula). Figure 91 shows the rule that predicts the copula by checking the context of the current record (*seg*) against the DT (*manualcop.model.02.xml*). The result of the prediction (*yes* or *no*) is stored in a variable (*aValue*). Depending on the contents of this variable, the corresponding lemma is assigned to the *Pred* attribute.

```
GenLF_Predict_estar:
        SEMREC ( Cat=="Verb" & Pred in? set{ser estar})
        --> SEMREC {
          segrec rec;
          atom aValue;
         aValue=aBestDTLabel("manualcop.model.0.2.xml","A~is_estar", seg, 0);
         if (aValue=="yes") Pred="estar"; else Pred="ser";
             }
```

Figure 91: Rule that predicts the copula

The rule that predicts the lexical value of the copula applies in Pre-Generation. It applies on the incoming logical form and modifies it, before the basic syntactic tree gets built. In that way, the copula is already selected when the core Generation rules apply and different rules can apply depending on whether the copula is *ser* or *estar*.

6.4.8 Evaluation in the Context of MT

The Spanish generation grammar in the context of which this experiment has been performed is currently being used to generate the Spanish output of an MT system that has English as input. In this MT system, all lexical selections are, in principle, performed by transfer, which uses contextual information. As explained in Section 2.6, transfer rules are automatically extracted from parsed aligned corpora. The lemma of the copula is also computed by transfer rules, with a varying degree of accuracy. We wanted to perform a second evaluation of our best DT model, this time in an MT environment. We picked the model that had been trained on MS manuals. We had two goals in mind:

- prove that a model trained on a monolingual Spanish corpus could be used on structures coming from transfer;

- compare the degree of accuracy of the model vs. the transfer component in the task of copula selection.

We took 1,496 English sentences from computer manuals that contained the copula and processed them with our English-Spanish MT system, keeping the copula that transfer had decided upon. We then kept these results in a master file. We included a rule in the Generation grammar that removed the lemma of the copula and recalculated it using the DT model, and then ran regressions on the previous master file. We obtained 141 differences. Those were the cases for which transfer and the DT predicted a different copula. Since we were only looking at the differences, we were in fact ignoring the cases where transfer and DT were both right or both wrong. We reviewed all the differences manually and obtained the results shown in Table 15 and Figure 92.

	#differences
DT was better	100/141
Transfer was better	19/141
Neither[114]	22/141

Table 15: Results on copula selection by transfer and DT

[114] Those were cases were the output was too ill-formed to consider correctness of the copula.

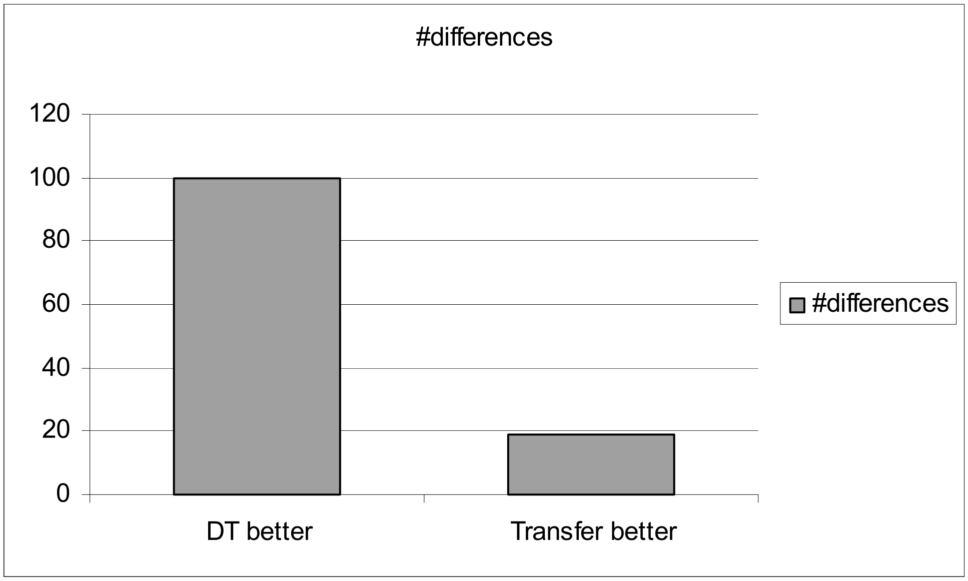


Figure 92: Comparison of transfer vs. DT results on the task of copula selection

Not counting the cases where neither was better, we have a total of 119 significant differences. Out of this total, 84% of the cases DT was right and transfer was right 16% of the cases.

These results are important because they prove the usability of the models on transferred structures, even though they have not been trained exactly on these structures.

6.4.9 Use of Probability Values to Improve Performance in MT

In spite of the overwhelming positive results reported above, we wanted to take advantage of the few cases where transfer was right and the prediction by the DT was less reliable. For this purpose we used the number indicating the probability distribution of each decision. For values of the decision tree that were between 0.4 and 0.6 we gave preference to the lemma chosen by transfer.

```
 GenLF_Predict_estar:
 SEMREC ( Cat=="Verb" & Pred in? set{ser estar})
 --> SEMREC
{ segrec rec;
  doubledProb;
  atom aValue;
  aValue=aBestDTLabel("manualcop.model.0.2.xml", "A~is_estar", seg, 0);
  dProb = dProbOfLabel ("manualcop.model.0.3.xml", "A~is_estar", "yes", seg, 0);
  if (dProb > 0.6 | dProb < 0.4 )
   { if (aValue=="yes") Pred="estar"; else Pred="ser";}}
```

Figure 93: Rule augmented with probabilities

By rejecting decisions that have the lowest probability, both the number of good differences and bad differences diminish slightly. However, not surprisingly, as the results in Table 16 show, the category that is affected more dramatically is the one that contains mainly ill-formed output (which we have labeled “Neither”). It goes from 22 cases in the first experiment, to 11 cases in the experiment using probabilities. That means that 50% of the ill-formed output cases involve a decision on the copula that has a very low probability.

	#differences
DT was better	95/121
Transfer was better	15/121
Neither	11/121

Table 16: Results on copula selection by transfer and DT, using probabilities

All in all, by using probabilities, we obtain a 2% improvement, going from 84% of better results using DTs to 86% (or 95 out of 110).

6.5 Conclusions

The results of the experiments presented in this chapter show that it is possible to machine learn the contexts for a non-trivial linguistic phenomenon such as the selection of the copula in Spanish.

The particular nature of the copula selection, which is a binary choice that involves a complex interaction of linguistic attributes, led us to the use of decision tree classifiers. The risk of overfitting was lessened by the use of large corpora; one containing 98K relevant sentences[115] from Encarta and the other with 48K relevant sentences from MS manuals. The training corpora did not have to be specifically annotated for the task because any reasonable text of native Spanish already contains the right lexical value of the copula.

The DT models that have been trained on these corpora show a high degree of accuracy for the problem at hand, somewhere between 95% and 97% depending on the type of text.

By directly inspecting the features considered as having predictive value by the DT classifiers, some interesting linguistic regularities surface, which are valid across different data and domains and which could be used as well in a hand-coded rule. Nonetheless, cross-type evaluation of the models and the variability of the predictors with lowest rank also show that DTs are extremely sensible to the training data, picking up on facts that are true only to a particular corpus. This capability makes them an extremely customizable tool, being able to adapt to each new set of data.

Our experiments show that the model trained on data coming from MS manuals has higher relative precision for the selection of the *estar* and also performs better across different text types than the model trained on Encarta. This is partly explained by the different proportions between the two values of the copula in the two types of text (1:2 in MS Manuals vs. 1:5 in Encarta). Decision trees seem to perform better when trained on data with a more balanced distribution of values of the target feature.

[115] That is, sentences containing at least one instance of the copula.

We have also shown how the models can be successfully used in the context of an application such as MT. With this experiment, we have also demonstrated that a model that has been trained on a monolingual corpus can be used on Logical Form structures coming from Transfer.

Chapter 7
Conclusions

7.1 Contributions of this thesis

The main contributions of this thesis can be summarized as follows:

1. Firstly, this thesis is a detailed description of a mature, large-scale, wide-coverage, rule-based Generation grammar for Spanish, developed within the framework of the Microsoft Natural Language Processing system. The description of the organization of the linguistic knowledge in the different rule blocks, as well as the careful attention given to the linguistic strategies behind these rules, constitute a complete set of specifications that can be used as a Reference Guide to build a rule-based Generation grammar for Spanish, using a different framework [Melero and Font-Llitjos, 2002].
2. Secondly, we discuss the role of the Generator as a provider of robustness to the application of which it is the last step. More specifically, we have analyzed this role within a real-life multilingual Machine Translation system which has commercial quality and is comparable to the best systems in the market. To attain the desired robustness, without affecting the independence of the Generator, we postulate the need for a Pre-Generation module that ensures the integrity of the input before it reaches the Generation grammar [Aikawa et al, 2001a, 2001b].
3. Thirdly, we claim that because of Pre-Generation's strong input-dependent nature, at least certain linguistic operations that take place at this stage are suited to be modeled using statistical methods. More specifically, this thesis shows that it is possible to machine learn the contexts of a non-trivial linguistic phenomenon, such as the selection of the Spanish copula, using Decision Trees, with a high degree of precision [Melero et al, 2002]. While there exists previous work on using Machine Learning techniques to perform automatic Generation operations [Minnen et al., 2000; Corston-Oliver et al., 2002], to our knowledge this is the

first time that these techniques have been used for Spanish Generation, more specifically to address a complex phenomenon such as copula selection. We have also shown that linguistic domain has an impact on the overall accuracy of the DT models, and that the more varied the input corpus is in terms of text type, the more portable the resulting model is. Lastly, we have proved that the models trained on a monolingual corpus, can be successfully used in the context of a bilingual application such as MT.

4. Finally, this thesis contains a detailed study of the use of Spanish copulas *ser* and *estar*, both from a descriptive perspective and from a computational perspective. It also contributes to this interesting issue, with a linguistic analysis of the statistical models obtained in the experiments, as well as of the classifiers having more predictive value. From this analysis, some relevant linguistic regularities surface, which are valid across different data and domains.

7.2 Future work

Natural language Generation has a wide variety of applications. Such applications include Machine Translation, human-computer dialogue, summarization, report creation, automatic technical documentation, proof/decision explanation, customized instructions, item and event descriptions, question answering, tutorials, stories, and more. While many applications use a custom-built generator, a general-purpose system can facilitate reuse of resources and reduce the costs of building applications [Langkilde-Geary, 2002].

The English language has a long array of general-purpose, off-the-shelf Surface Realizers, including FUF/Surge [Elhadad and Robin, 1996b], RealPro [Lavoie and Rambow, 1997], Penman/KPML [Bateman, 1997] and Nitrogen [Langkilde and Knight, 1998]. These systems have demonstrated their general usefulness by being deployed in a variety of different applications.

Two of the aforementioned systems (Surge and KPML) have been ported to Spanish, in both cases to be used in a specific application context: story narration in the first case and a chemistry database querying system in the other. However, there isn’t any general-purpose Generator for Spanish that can be plugged into any application needing to convert a syntactic/semantic representation of a sentence into actual text.

We propose to use the linguistic specifications contained in this thesis to implement a comprehensive, general-purpose, rule-based Spanish Generation Grammar, in an open-source environment, which can be plugged into a variety of applications.

At the same time, in line with the mainstream tendency towards hybrid systems, we propose to expand the statistical nature of the Pre-Generation module to cover other linguistic operations. The automatic selection of the lemma of the Spanish copula is only an example of linguistic operation that can be machine learned through the use of a statistical tool such as Decision Trees. The good results obtained suggest that the same procedure can be applied to other linguistic phenomena.

By covering more phenomena in a machine learned Pre-Generation module, we empower the Generator and at the same time, we enhance the robustness of the applications that use it. Following the Generation-Heavy approach presented by [Habash and Dorr, 2002], as well as recent experiments in the framework of the Metis project [Badia et al, 2005; Carl et al, 2005; Markantonatou et al, 2005; Vandeghiste et al, 2005], we claim that shifting the weight towards TL Generation in applications such as MT, allows for the other components be less complex, for example Transfer (which can be reduced to a simple bilingual dictionary) and even SL Analysis.

The selection of the copula illustrates a complex phenomenon in the TL that can be efficiently learned by Generation using monolingual corpora and that does not need to affect other components related with the SL, such as Transfer.

An extra bonus of the Generation-Heavy approach (or the do-it-in-Generation approach) is that results can be reused on inputs coming from different sources: DB querying, summarization, different transfer modules, etc. Otherwise the computation for the different TL phenomena needs to be recalculated in each language pair or application.

A tentative list of linguistic phenomena in Spanish to be machine-learned includes:

- Insertion of the article. Similar works are reported by [Minnen et al., 2000] for English and [Murata et al, 2000] for Japanese.

- Insertion of the reflexive pronoun *se*. We anticipate that the inspection and analysis of the resulting models will reveal interesting facts that can be contrasted with the theoretical assumptions that are commonly hold about this phenomenum.
- Number of the noun in certain syntactic contexts
- Finiteness of the verb
- Selection of the lemma of the preposition (e.g. *por* / *para* opposition)

Lastly, one of the advantages of the experiment presented in this thesis is that the corpora used to train the DT classifiers had not been manually annotated (contrarily to most previous works in English, that used the Penn Treebank [Minnen et al., 2000; Knight et al., 1994]), but had been processed using a high level Analysis grammar (NLPWin's). In further experiments, we expect to get equally good results by training the classifiers on text tagged using only low level tools, such as a POS tagger and a shallow parser or a chunker.

References

Aguado G., A. Bañón, John A. Bateman, S. Bernardos, M. Fernández, A. Gómez-Pérez, E. Nieto, A. Olalla, R. Plaza, and A. Sánchez (1998). ONTOGENERATION: Reusing domain and linguistic ontologies for Spanish text generation. In Proceedings of the ECAI'98 Workshop on Applications of Ontologies and Problem Solving Methods, pages 1--10, Brighton, U.K.

Aikawa, T., M. Melero, L. Schwartz, and A. Wu. (2001a). Multilingual Sentence Generation. In *Proceedings of 8th European Workshop on Natural Language Generation (ACL-2001),* Toulouse.

Aikawa, T., Maite Melero, Lee Schwartz and Andy Wu (2001b). Generation for Multilingual MT. In *Proceedings of the VIII MT-Summit*, Santiago de Compostela, Spain.

Alonso, JA and Thurmair, G. (2003) The Comprendium Translator System. In: Proceedings of the Ninth Machine Translation Summit, New Orleans, USA.

Automatic Language Processing Advisory Committee (ALPAC) (1966). Language and Machines: Computers in Translation and Linguistics.

Arnold, D., L. Balkan, R. L. Humphreys, S. Meijer, and L. Sadler (1994). Machine Translation: An Introductory Guide, NCC Blackwell, Manchester.

Arnold D. and L. des Tombe (1987). Basic Theory and Methodology in Eurotra. In S. Nirenburg, editor, Machine Translation: Theoretical and Methodological Issues, pages 114-135. Cambridge University Press, Cambridge, England.

Badia, T., G. Boleda, M. Melero and A. Oliver (2005). An n-gram approach to exploiting monolingual corpus for MT *In Proceedings of the Second Workshop on Example Based Machine Translation*. MT Summit X. Phuket, Thailand.

Bangalore, S., Chen, J. and Rambow, O. (2001). Impact of Quality and Quantity of Corpora on Stochastic Generation. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, Pittsburgh, Pennsylvania.

Bangalore, S., Rambow O. and Whittaker S. (2000). Evaluation Metrics for Generation. *In Proceedings of the International Conference on Natural Language Generation (INLG 2000)*, Mitzpe Ramon, Israel.

Bangalore, S. and Owen Rambow (2000). Exploiting a probabilistic hierarchical model for generation. In COLING-2000: Proceedings of the 18th International Conference on Computational Linguistics, Saarbruecken, Germany.

Banko, Michele and E. Brill. (2001). Scaling to Very Very Large Corpora for Natural Language Disambiguation. In *Proceedings of the Association for Computational Linguistics*, Toulouse, France, pp. 23-26.

Banko, Michele and V. Mittal, M. Witbrock. (2000). Generating Headline-Style Summaries. In *Proceedings of the Association for Computational Linguistics*, Hong Kong.

Bateman, John A. and Hartley Anthony F. (2000). Target Suites for Evaluating the Coverage of Text Generators. In Proceedings of LREC, 2000. Athens.

Bateman, John, A. Christian M. I. M Matthiessen, and Licheng Zeng (1999). Multilingual natural language generation for multilingual software: a functional linguistic approach. Applied Artificial Intelligence, 13(6):607--639.

Bateman, John A (1997). Enabling technology for multilingual natural language generation: the KPML development environment. Journal of Natural Language Engineering, 3(1):15—55

Becker T. and Stephan Busemann, editors (2000). Impacts in Natural Language Generation: NLG between Technology and Applications. Workshop at Schloss Dagstuhl, Germany, number D-00-01 in DFKI Document, Saarbrücken.

Belz Anja (2005). Statistical generation: Three methods compared and evaluated. In Proceedings of the 10th European Workshop on Natural Language Generation (ENLG'05).

Brew, C. (1992). Letting the Cat Out of the Bag: Generation for Shake-and-Bake MT, in Proceedings of the 15th International Conference on Computational Linguistics, COLING-92, Nantes, France, pp. 603–609.

Becker, T., Anne Kilger, Patrice Lopez, and Peter Poller (2000). An extended architecture for robust generation. In Proceedings of INLG 2000, Mitzpe Ramon, Israel.

Bosque, I. y Demonte, V. eds. (1999): Gramática Descriptiva de la Lengua Española. Espasa Calpe, Madrid.

Breiman L., Friedman J.H., Olshen R.A., and Stone P.J. (1984). Classification and regression trees. Belmont, CA: Wadsworth International Group.

Bresnan, J. (2001). Lexical-Functional Syntax. Malden, MA and Oxford: Blackwell.

Busemann, Stephan (2002). Issues in generating from interlingua representations. In Edmundo Tovar and Carolina Gallardo, editors, Proceedings of the 1st International Workshop on UNL, other Interlinguas and their Applications, pages 1-7, Las Palmas.

Busemann S. and Helmut Horacek (1998) A Flexible Shallow Approach to Text Generation, International Workshop on Natural Language Generation, Niagara-on-the-Lake, Ontario, Canada.

Busemann, Stephan (1997). Putting semantic-head-driven generation to the limits: Experiments with multi-purpose semantic representations. In T. Becker, S. Busemann, and W. Finkler, editors, *DFKI Workshop on Natural Language Generation, April 1997*, number D-97-06 in DFKI Document, pages 8-14, Saarbrücken, Germany.

Callaway, C (2003). Evaluating coverage for large symbolic NLG grammars. In Proceedings of IJCAI 2003.

Callaway, C. and James C. Lester (2002). Narrative Prose Generation. Artificial Intelligence, 139(2):213–252. Natural Language Generation Systems. Cambridge University Press, Cambridge, England.

Callaway, C., B. Daniel, and J. Lester (1999). Multilingual natural language generation for 3D learning environments. In Proceedings of the 1999 Argentine Symposium on Artificial Intelligence, pages 177–190, Buenos Aires, Argentina.

Campbell, R. (2002). Computation of modifier scope in NP by a language-neutral method. In Proceedings of COLING 2002.

Campbell, R., T. Aikawa, Z. Jiang, C. Lozano, M. Melero and A. Wu. (2002a). A language-neutral representation of temporal information. Workshop on annotation standards for temporal information in natural language, LREC 2002.

Campbell, R., Carmen Lozano; Jessie Pinkham; Martine Smets (2002b) Machine Translation as a Testbed for Multilingual Analysis Proceedings of the Workshop: Grammar Engineering and Evaluation (COLING-02).

Campbell, R. and H. Suzuki. (2002a). Language-neutral representation of syntactic structure. In R. Malaka, R. Porzel and M. Stube (eds.), Proceedings of SCANALU.

Campbell, R. and Suzuki, H. (2002b): Language-Neutral Syntax: An Overview. MSR Technical Report MSR-TR-2002-76

Cardeñosa1, J., Carolina Gallardo and Edmundo Tovar (2003). Standardization of the Generation Process in a Multilingual Environment in Proceedings of the International Conference on the Convergence of Knowledge, Culture, Language and Information Technologies December 2 - 6, Alexandria, Egypt.

Carl, M., P. Schmidt, and J. Schütz (2005). Reversible Template-based Shake & Bake Generation. *In Proceedings of the Second Workshop on Example Based Machine Translation*. MT Summit X. Phuket, Thailand.

Carl, M. Way A. and Schäler R. (2002). Toward a Hybrid Integrated Translation Environment. In Proceedings AMTA '02: Proceedings of the 5th Conference of the Association for Machine Translation in the Americas on Machine Translation: From Research to Real Users, Springer-Verlag London, UK.

Cestnik B., Kononenko I., and Bratko I. (1987). ASSISTANT-86: A knowledge-elicitation tool for sophisticated users. In I. Bratko and N. Lavrac (Eds), Progress in machine learning. Bled, Yugoslavia: Sigma Press.

Chickering D. M., Heckerman D. and Meek C. (1997). A Bayesian approach to learning Bayesian networks with local structure. In *Uncertainty in Artificial*

Intelligence: Proceedings of the Thirteenth Conference, D. Geiger and P. Punadlik Shenoy, ed., Morgan Kaufman, San Francisco, California, pp. 80-89.

Chickering, D. Max. (2002). The WinMine Toolkit. Microsoft Technical Report. MSR-TR-2002-103. Redmond, WA.

Corston-Oliver, S. and M. Gamon. (2003). Combining decision trees and transformation-based learning to correct transferred linguistic representations. In Proceedings of the Ninth Machine Translation Summit. 55-62. New Orleans, USA.

Corston-Oliver S., Gamon M., Ringger E. and Moore B. (2002). An overview of Amalgam: a machine-learned generation module. In *Proceedings of the Second International Natural Language Generation Conference 2002*, New York, USA.

Corston-Oliver, Simon, Michael Gamon and Chris Brockett. (2001). A Machine Learning Approach to the Automatic Evaluation of Machine Translation. In *Proceedings of the Association for Computational Linguistics*, Toulouse, France, pp. 140-147.

Dale R., Moisl H., and Somers H. (eds.), (2000): *A Hand-book of Natural Language Processing: Techniques and Applications for the Processing of Language as Text*. Marcel Dekker, New York.

Dale, R. and Mellish, C. (1998): Towards the Evaluation of Natural Language Generation. In *Proceedings of the First International Conference on Evaluation of Natural Language Processing Systems*, May 28-30, Granada, Spain.

Dale, R., Hovy, E. H., Rösner, D. and Stock, O. editors (1992). *Aspects of Automated Natural Language Generation*. Number 587 in Lecture Notes in AI. Springer-Verlag, Heidelberg.

Dolan, William B., L. Vanderwende, and S. Richardson. (1993). Automatically Deriving Structured Knowledge Base from On-line Dictionaries. In Proceedings of the Pacific Association for Computational Linguistics, April 21-24, 1993, Vancouver, British Columbia.

Dorr, Bonnie J., Pamela W. Jordan, and John W. Benoit, (1999) A Survey of Current Research in Machine Translation, Advances in Computers, Vol 49, M. Zelkowitz (Ed), Academic Press, London, pp. 1--68.

Doyon, J., K. Taylor, and J.S. White. (1998). The DARPA MT evaluation methodology: Past and present. In Proceedings of the AMTA Conference, Philadelphia, PA.

Duda, R. O., and Hart, P. E. (1973). Pattern classification and scene analysis. New York: Wiley & Sons.

Durand, J., P. Bennett, V. Allegranza, F. van Eynde, L. Humphreys, P. Schmidt and E. Steiner. (1991) "The Eurotra Linguistic Specifications: An Overview", Machine Translation, 6.2 (Special issue on Eurotra), pp. 103-147.

EAGLES (1995): Evaluation of Natural Language Processing Systems (Final Report) http://issco-www.unige.ch/ewg95/ewg95.html

Elhadad, M. and Jacques Robin (1996a). A reusable comprehensive syntactic realization component. In Demonstrations and Posters of the 1996 International Workshop on Natural Language Generation (INLG '96), pp. 1-4, Herstmonceux, England, June.

Elhadad, M., and Robin, J. (1996b). An overview of SURGE: A reusable comprehensive syntactic realization component. *Technical Report Technical Report 96-03, Dept of Mathematics and Computer Science, Ben Gurion University*, Beer Sheva, Israel.

Elhadad, Michael (1991). FUF: The universal unifier user manual version 5.0. Technical Report CUCS-038-91, Department of Computer Science, Columbia University.

Esposito, F., D. Malerba, G. Semeraro and V. Tamma (1999). The Effects of Pruning Methods on the Predictive Accuracy of Induced Decision Trees, Applied Stochactic Models in Business and Industry, 15, 4, pp. 277-299.

Falk, Johan (1979). Visión de norma general vs visión de norma individual. Ensayo de explicación de la oposición ser/estar en unión con adjetivos que denotan belleza y corpulencia. StN 51, pp 275-293

Fernández Leborans, M. Jesús. (1999) “La predicación: Las oraciones copulativas”. In: Bosque, Ignacio / Demonte, Violeta (eds.): Gramática descriptiva de la lengua española. Madrid: Real Academia Española / Espasa Calpe. 1999, vol. II, § 37.6.

Flanagan, M. (1994). Error classification for MT evaluation. In Proceedings of the AMTA Conference, Columbia, Maryland.

Friedman J.H. (1977). A recursive partitioning decision rule for non-parametric classification. In IEEE Transactions on Computers. pp 404-408.

Gamon, M., Ringger, E., Corston-Oliver, S. and Moore, R. (2002a). Machine-learned contexts for linguistic operations in German sentence realization. In *Proceedings of the Association for Computational Linguistics 2002*, Pennsilvania.

Gamon, M., Ringger, E., Zhang, Z., Moore, R. and Corston-Oliver, S. (2002b). Extraposition: A case study in German sentence realization. In *Proceedings of COLING 2002*, Taipeh.

Gamon, M., Suzuki, H. and Corston-Oliver, S. (2001): Using Machine Learning for System-Internal Evaluation of Transferred Linguistic Representations. In *Proceedings of MT Summit VIII*, Santiago de Compostela, Spain, 18-22 September 2001. pp. 109-114.

Grishman, R. and M. Kosaka (1992). Combining Rationalist and Empiricist Approaches to Machine Translation. In Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages, pages 263-274, Montreal, Canada.

Habash, N. and Dorr, B.J. (2002). Handling Translation Divergences: Combining Statistical and Symbolic Techniques in Generation-Heavy Machine Translation. *In Proceedings AMTA '02: Proceedings of the 5th Conference of the Association for Machine Translation in the Americas on Machine Translation: From Research to Real Users,* Springer-Verlag London, UK, pages 84—93.

Halliday, M.A.K. (1976), Halliday : System and function in language: selected papers, (ed by G. Kress). London: Oxford University Press.

Heidorn, G. E. (2000). Intelligence Writing Assistance. In Dale R., Moisl H., and Somers H. (eds.), *A Hand-book of Natural Language Processing: Techniques and Applications for the Processing of Language as Text*. Marcel Dekker, New York, 1998 (published in August 2000), pages 181-207.

Holland J.H. (1986). Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems. In R.S. Michalski, J.G.Carbonell, and T.M. Mitchell (Eds), Machine Learning: An Artificial Intelligence Approach (Vol 2). San Mateo, CA: Morgan Kaufmann.

Hovy E., King M. and Popescu-Belis A., Principles of Context-Based Machine Translation Evaluation, In *Machine Translation*, vol. 16, 2002, p.1-33.

Hunt E.B. (1975). Artificial Intelligence. New York: Academic Press.

Hunt E.B., Marin J., and Stone P.J. (1966). Experiments in Induction. New York: Academic Press.

Hutchins, J., Hartmann, W. and Ito, E. (2004): Compendium of translation software: commercial machine translation systems and computer-based translation support tools. 8th edition, January 2004. [Available at http://www.eamt.org]

Hutchins, J. (2003) Machine translation and computer-based translation tools: what's available and how it's used in A new spectrum of translation studies, ed. José Maria Bravo (Valladolid: Univ. Valladolid) [Available at http://ourworld.compuserve.com/homepages/WJHutchins]

Hutchins, W.J. (1999). The development and use of machine translation systems and computer-based translation tools. International Conference on Machine Translation & Computer Language Information Processing. Proceedings of the conference, 26-28 June 1999, Beijing, China, ed.Chen Zhaoxiong, 1-16.

Hutchins, W.J. (1986): Machine translation: past, present, future. Chichester: Ellis Horwood.

Jensen, K., Heidorn, G.E., and Richardson, S.D. eds. (1993). Natural Language Processing: The PLNLP Approach. Kluwer Academic Publishers.

Jiménez, M.L. (2001). Generation of Named Entities. In Proceedings of the MT Summit VIII. Santiago de Compostela, Spain.

Knight, Kevin and Ishwar Chander (1994). Automated Postediting of Documents. *In Proceedings of the 12th National Conference on Artificial Intelligence*: AAAI-94, pages 779–784, Seattle, USA.

Langkilde-Geary, Irene (2002). An empirical verification of coverage and correctness for a general-purpose sentence generator. In Second International Natural Language Generation Conference, pages 17-24, Harriman, NY,

Langkilde, I. and Knight, K. (1998). Generation that exploits corpus-based statistical knowledge. In *Proceedings of COLING-ACL'98*, Montreal, Canada.

Lavoie, Benoit and Owen Rambow (1997). A fast and portable realizer for text generation. In Proceedings of the Fifth Conference on Applied Natural-Language Processing (ANLP-1997), pages 265-268.

Lester, J. and Porter, B. (1997). Developing and Empirically Evaluating Robust Explanation Generators: The KNIGHT Experiments. *Computational Linguistics*, 23(1), pp. 65-101.

Levine, J. and Mellish, C. (1995). The IDAS user trials: Quantitative evaluation of an applied natural language generation system. In *Proceedings of the Fifth European Workshop on Natural Language Generation*, pages 75--94, RijksUniversiteit Leiden.

Lenci, A., Roberto Bartolini, Nicoletta Calzolari, Ana Agua, Stephan Busemann, Emmanuel Cartier, Karine Chevreau, and José Coch (2002). Multilingual summarization by integrating linguistic resources in the mlis-musi project. In Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC'02), May 29-31, Las Palmas, Canary Islands, Spain.

Lozano, C and Melero, M. (2001). Spanish NLP Projects at Microsoft Research in *Proceedings of the 2nd International Workshop of Spanish Language Processing and Language Technologies*, Jaén (Spain).

R. López de Mántaras. (1991). A Distance-Based Attribute Selection Measure for Decision Tree Induction. In *Machine Learning*, vol. 6(1), 1991. Págs. 81-92.

McClure, S. and Flanagan, M. (2003) Overcoming the MT Quality Impasse Study #29917 – IDC http://www.systransoft.com/IDC/29917.htm

McClure, S. and Flanagan, M. (2000) Machine Translation Engines: An Evaluation of Output Quality - IDC

McDonald, D. (2000). *Natural Language Generation* In Dale R., Moisl H., and Somers H. (eds.), *A Hand-book of Natural Language Processing: Techniques and Applications for the Processing of Language as Text*. Marcel Dekker, New York, 1998 (published in August 2000), pp. 147-181.

Manning, C.D. and H. Schütze. (1999) Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge, Massachussetts.

Markantonatou, S., S. Sofianopoulos, V. Spilioti, G. Tambouratzis, M. Vassiliou, O. Yannoutsou, N. Ioannou (2005). Monolingual Corpus-based MT using chunks. *In Proceedings of the Second Workshop on Example Based Machine Translation*. MT Summit X. Phuket, Thailand.

Mariani, J (1996): *Evaluation In Studies in NLP: Survey of the State of the Art in Human Language Technology*. CUP, Cambridge. http://cslu.cse.ogi.edu/HLTsurvey/

Maybury, M. T. (1990). Evaluation Spaces: A Framework for Evaluating Natural Language Generation Systems. *AAAI-90 Workshop in Evaluating Natural Language Generation Systems*.

McClelland, J. and Rumelhart, D. (1988). Explorations in Parallel Distributed Processing: A handbook of Models, Programs, and Exercises. Cambridge, Mass.: MIT Press

Mel'čuk, I. (1988). Dependency Syntax: Theory and practice. New York: State University of New York Press.

Melero, M., T. Aikawa and L. Schwartz. (2002). Combining machine learning and rule-based approaches in Spanish and Japanese sentence realization. Second International Natural Language Generation Conference, New York.

Melero, M. (2001). Cuestiones relacionadas con la especificación y el desarrollo de un corrector gramatical del español. Master thesis. Universitat Pompeu Fabra.

Melero, M. and Font-Llitjos, A. (2001). Construction of a Spanish Generation module in the framework of a General-Purpose, Multilingual Natural

Language Processing System. In *Proceedings of the VII International Symposium on Social Communication*, Santiago de Cuba, pages 283-287.

Mellish, C. and Dale, R. (1998) Evaluation in the Context of Natural Language Generation. *Computer Speech and Language*, 12, 349-373.

Menezes, Arul and Steve Richardson (2001). A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL-2001)*, pp. 39-46, Toulouse.

Minnen, Guido, Francis Bond and Ann Copestake (2000). Memory-Based Learning for Article Generation. *In Proceedings of CoNLL-2000 and LLL-2000*, Lisbon, Portugal.

Mitchell, T. (1997). Decision Tree Learning. In T. Mitchell, *Machine Learning*, The McGraw-Hill Companies, Inc., 1997, pp. 52-78.

Molina Redondo, J.A. de and Ortega Olivares J. (1996) *Usos de ser y estar*. SGEL, Madrid.

Moore, Robert C. (2001). Towards a Simple and Accurate Statistical Approach to Learning Translation Relationships among Words. In *Proceedings, Workshop on Data-driven Machine Translation, 39th Annual Meeting and 10th Conference of the European Chapter, Association for Computational Linguistics*, Toulouse, France, pp. 79-86.

Murata, M., K. Uchimoto, Q. Ma, H. Isahara (2000): Machine Learning Approach To Estimating a Referential Property of a Noun Phrase, Journal of Natural Language Processing, Vol. 7, No. 1, pp. 31-50.

Natural Language Processing Group, Microsoft Research. (2001). *Tools for Large-Scale Parser Development. In Proceedings for the Workshop on Efficiency in Large-Scale Parsing Systems COLING 2000*, Saarbruken, Germany, p. 54.

Nyberg, E. and Mitamura, T. (1992) The KANT system: Fast, accurate, high-quality translation in practical domains. In Proceedings of Fourteenth International Conference on Computational Linguistics, Nantes, France.

Okumura, A., K. Muraki, and K. Yamabana (1992). A Pattern-Larning Based, Hybrid Model for the Syntactic Analysis of Structural Relationships among Japanese Clauses. In Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages, pages 45-54E[, Montreal, Canada.

Pan, S. and J. Shaw (2004). SEGUE: A hybrid case-based surface natural language generator. In Proceedings of INLG 2004.

Papineni, K., Roukos, S., Ward, T. and Zhu, W. (2001): Bleu: a Method for Automatic Evaluation of Machine Translation. In *Computer Science*

Partee, B. (1977). John is easy to please. In A. Zampolli, ed., *Linguistic Structures Processing*, pp. 281-312. North Holland, Amsterdam.

Project Penman (1989) Technical report, USC/Information Sciences Institute, Marina del Rey, CA.

Pianesi, F., Pianta, E. and Tovena L. (IRST) (1999), Comparing methodologies for evaluating the generator in a speech-to-speech translation system. In *7th European Workshop on Natural Language Generation*. Toulouse, 13, 14.

Pinkham, J. and Smets, M. (2002). Modular MT with a learned bilingual dictionary: rapid deployment of a new language pair. In *Proceedings of the 19th international Conference on Computational Linguistics - Volume 1* (Taipei, Taiwan, August 24 - September 01, 2002). International Conference On Computational Linguistics. Association for Computational Linguistics, Morristown, NJ, 1-7.

Pinkham, J, and M. Corston-Oliver. (2001). Adding Domain Specificity to an MT System. In Proceedings of the Workshop on Data-driven Machine Translation at 39th Annual Meeting of the Association for Computational Linguistics, Toulouse, France, pp. 103-110

Pinkham, J., M. Corston-Oliver, M. Smets and M. Pettenaro. (2001). Rapid Assembly of a Large-scale French-English MT system. In Proceedings of the MT Summit VIII, Santiago de Compostela, Spain.

Porroche Ballesteros, M. (1988). *Ser, estar y verbos de cambio*. Arco/Libros, Madrid.

Quinlan, J. Ross. (1992) C4.5: Programs for Machine Learning. Morgan Kaufmann

Quinlan, J. Ross. Induction of Decision Trees. In *Machine Learning*, 1:81--106.

Ratnaparkhi, Adwait (2000). Trainable methods for surface natural language generation. In Proceedings of the First North American Conference of the ACL, Seattle, WA, May.

Reiter, E. and Dale, R. (2000): Building Natural Language Generation Systems, Cambridge University Press.

Richardson, S., W. Dolan, A. Menezes, and J. Pinkham. (2001a). Achieving commercial-quality translation with example-based methods. In *Proceedings of MT Summit VIII,* Santiago de Compostela, Spain, pp. 293-298.

Richardson, S., W. Dolan, A. Menezes, and M. Corston-Oliver. (2001b). Overcoming the customization bottleneck using example-based MT. In *Proceedings, Workshop on Data-driven Machine Translation, 39th Annual Meeting and 10th Conference of the European Chapter, Association for Computational Linguistics* Toulouse, France, pp. 9-16.

Richardson, S. (2000): The evolution of an NLP System. NLP Group Microsoft Research. Presentation at the LREC'2000 Athens, Greece.

Richardson, Stephen D., Dolan, William B., and Vanderwende, Lucy (1998). MindNet: acquiring and structuring semantic information from text. In Proceedings of COLING '98.

Richardson, S. (1994). Bootstrapping Statistical Processing into a Rule-based Natural Language Parser. In *The Balancing Act: Combining Symbolic and Statistical Approaches to Language. Proceedings of the Workshop*, Las Cruces, New Mexico. pp. 96-103.

Richardson, S., L. Vanderwende, and W. Dolan. (1993). Combining Dictionary-based and Example-based Methods for Natural Language Analysis.. In . Proceedings of the Fifth International Conference on Theoretical and Methodological Issues in Machine Translation, Kyoto, Japan. pp 69-79.

Ringger, E., Corston-Oliver, M. and Moore, R. (2001): Using Word-Perplexity for Automatic Evaluation of Machine Translation. Unpublished Draft.

Schmitt, Cristina, Holtheuer Carolina and Miller Karen (2004) Acquisition of copulas ser and estar in Spanish: learning lexico-semantics, syntax and discourse. Boston University Conference on Language Development (BUCLD) 28 Proceedings Online Supplement

Somers, H. (1999). Review Article: Example-based Machine Translation. *Machine Translation* 14, 2 (Jun. 1999), 113-157

Stent, A., Matthew Marge, Mohit Singhai (2005). Evaluating Evaluation Methods for Generation in the Presence of Variation. CICLing 2005: 341-351

TEMAA (1997): A Testbed Study of Evaluation Methodologies: Authoring Aids (Final Report) http://www.cst.ku.dk/projects/temaa/temaa.html

Tomita, M. and Nyberg E. (1988): The GenKit and Transformation Kit User's Guide. Technical Report CMU-CMT-88-MEMO, Centre for Machine Translation, Carnegie Mellon University.

Toole, J., Popowich, F., Nicholson, D., Turcato, D. and McFetridge, P. (1999). Explanation-Based Learning for Machine Translation, in Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI99), Chester, U.K., pp. 161–171

Vandeghinste, V. ,P. Dirix, and I. Schuurman (2005). Example-based Translation without Parallel Corpora: First experiments on a prototype. *In Proceedings of the Second Workshop on Example Based Machine Translation*. MT Summit X. Phuket, Thailand.

Walker, M., Litman, M., Kamm, C. and Abella, A. (1997): PARADISE: A Framework for Evaluating Spoken Dialogue Agents. In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL-97), pp. 271-280, Madrid.

White J.S. et al. (1992-1994). ARPA workshops on MT (series of four workshops on comparative evaluation). Technical report, PRC Inc., McLean, Virginia.

White M. and Ted Caldwell (1998). EXEMPLARS: A practical, extensible framework for dynamic text generation. In Proceedings of the Ninth International Workshop on NLG, pages 266-275, Niagara-on-the-Lake, Ontario.

Whitelock, P. (1994) Shake and Bake Translation, in C. Rupp, M. Rosner, and R. Johnson (eds), Constraints, Language and Computation, Academic Press, London, pp. 339–359.

Zamorano Mansilla, Juan Rafael (2003). A Spanish grammar for KPML. http://www.fb10.uni-bremen.de/anglistik/langpro/kpml/genbank/generation-bank.html

Zhong H. and A. Stent (2005), "Building surface realizers automatically from corpora using general-purpose tools, in Proceedings of the Workshop on Using Corpora for Natural Language Generation (UCNLG 2005).