AnaPro, Tool for Identification and Resolution of Direct Anaphora in Spanish

I. Toledo-Gómez[1], E. Valtierra-Romero[1], A. Guzmán-Arenas[*2], A. Cuevas-Rasgado[3], L. Méndez-Segundo[1]

[1] Escuela Superior de Cómputo
Instituto Politécnico Nacional
México, D. F., México
[2] Centro de Investigación en Computación
Instituto Politécnico Nacional
México, D. F., México
*a.guzman@acm.org
[3] Universidad Autónoma del Estado de México
Centro Universitario Texcoco, Estado de México

ABSTRACT
AnaPro is software that solves direct anaphora in Spanish, specifically pronouns: it finds the noun or group of words to which the pronoun refers. It locates in the previous sentences the referent or antecedent which the pronoun replaces. An example of a direct anaphora solved is the pronoun “he” in the sentence “He is sad.” Much of the work on anaphora has been done for texts in English; thus, we specifically focus on Spanish documents.

AnaPro directly supports text analysis (to understand what a document says), a non trivial task since there are different writing styles, references, idiomatic expressions, etc. The problem grows if the analyzer is a computer, because they lack “common sense” (which persons possess). Hence, before text analysis, its preprocessing is required, in order to assign tags (noun, verb,…) to each word, find the stems, disambiguate nouns, verbs, prepositions, identify colloquial expressions, identify and resolve anaphora, among other chores.

AnaPro works for Spanish sentences. It is a novel procedure, since it is automatic (no user intervenes during the resolution) and it does not need dictionaries. It employs heuristics procedures to discover the semantics and help in the decisions; they are rather easy to implement and use limited knowledge. Nevertheless, its results are good (81% of correct answers, at least). However, more tests will give a better idea of its goodness.

Keywords: I.2. Artificial Intelligence, I.2.7 Natural Language processing, Text Analysis, Anaphora resolution.

RESUMEN
AnaPro es un software que resuelve problemas con anáforas directas en español, especialmente pronombres: la herramienta encuentra el sustantivo o grupo de palabras al cual se refiere el pronombre. Localiza en oraciones previas la referencia o antecedente y lo reemplaza por el pronombre que le corresponde. Un ejemplo de anáfora directa resuelta es el pronombre “él”, en la oración “Él está triste”.

AnaPro apoya directamente el análisis de textos (para entender lo que el documento dice), tarea no trivial ya que existen diferentes formas y estilos de escribir, referenciar, expresiones regionales (mexicanismos, argentinismos, etc.). Todavía se complica más si el analizador es una máquina porque no tiene “sentido común” (que cualquier persona puede tener). Así, antes de analizar el texto, se requiere pre-procesarlo para asignar etiquetas (sustantivos, y verbos, entre otros) a cada palabra, encontrar las derivaciones, desambiguar sustantivos, verbos, preposiciones, identificar expresiones coloquiales, identificar y resolver anáforas, entre otras tareas.

AnaPro trabaja con oraciones en español. Es un proceso novedoso porque es totalmente automático (no requiere la intervención de un usuario) y tampoco necesita de diccionarios. Emplea un procedimiento heurístico para descubrir la semántica y apoyarse en las decisiones que debe de tomar. Ha resultado ser bueno (con un 81-100% de aciertos); sin embargo, más ejemplos nuevos darán mejor idea de su desempeño.

1. Introduction

Anaphora is a relation of coreference between linguistic terms. According to Webster's dictionary: "It is the use of a grammatical substitute (as a pronoun or a pro-verb) to refer to the denotation of a preceding word or group of words; *also*: the relation between a grammatical substitute and its antecedent." Therefore, anaphora is a discourse relation. Anaphora resolution is very important in Natural Language Processing (NLP).

This work is part of Project OM* (Ontology Merging), which seeks to build a large ontology by fusing smaller ontologies extracted from textual documents. An important part of the project is to analyze the sentences in a document with the goal to transform that text into an ontology that comprises its contents. A brief description of Project OM* follows.

*1.1 Project OM**

Project OM* is built upon the OM software (Ontology Merging; [3]), which takes two ontologies as input and creates another ontology comprising the knowledge residing in the inputs, without self-inconsistencies. The OM software has been built and it works [3], but it takes as input *ontologies* (in a special notation called OM notation, explained below), instead of textual *documents*. Thus, additional work is needed to transform a document into an OM ontology to be fed to OM. That transformation is the goal of OM*. Project OM* is in full development. The parts of OM* that have been built appear in Figure 1, as well as a simplified flow of its computation. These are:

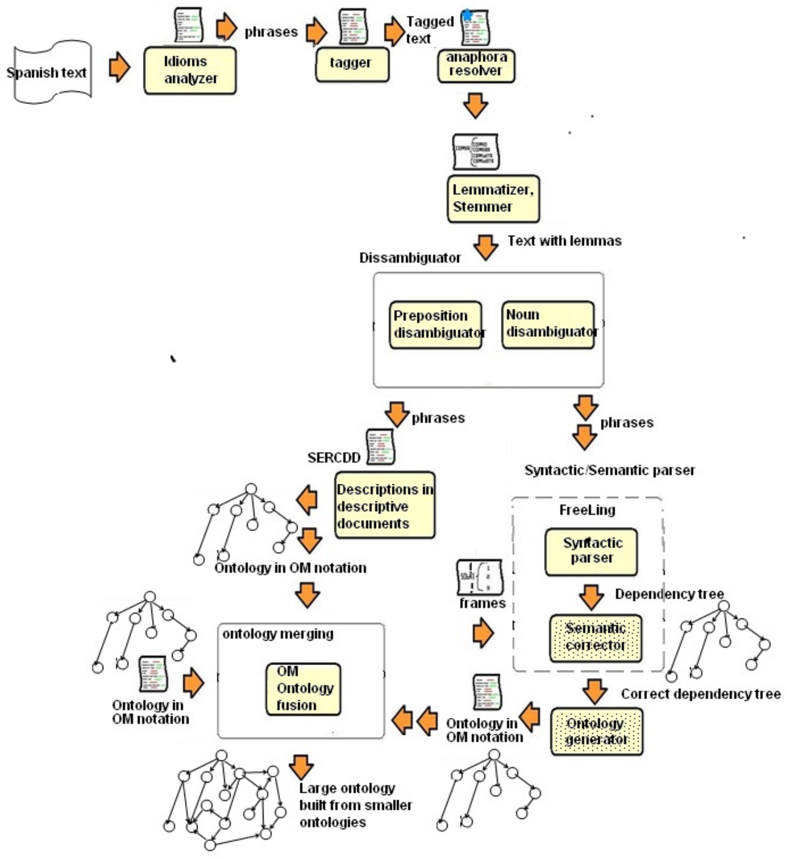


Figure 1. The parts of the OM* project. Parts under constructions are shown dashed. The diagram shows a simplified flow of computation from text to ontologies

Idioms analyzer. An idiom is a phrase whose meaning cannot be derived from the conjoined meaning of its elements. For instance, *vestida y alborotada* {dressed and agitated}[1] (meaning: *decepcionada* {adj. disappointed}); *de tal palo tal astilla* {chip off the old block} (adj. similar to his ancestor); *corazón de piedra* {stone heart} (adj. to be hard and inflexible). The idioms analyzer [12] finds and replaces them with their respective synonym. This increases the ability of subsequent modules of OM* to understand more clearly the sentences. We believe it is the first idioms finder for Spanish. It uses a dictionary of canonical idioms (stereotyped phrases) which are then compared with the text under analysis, looking for "legal" variations of the stereotype phrase.

The analyzer locates and converts the following types of idioms:

1. Colloquial expressions. Example: corazón de piedra {stone heart} (adj. to be hard and inflexible).

2. Specialized expressions. They describe an object in a specific or particular (specialized) context. Examples: línea de comandos {command line}, base de datos {data base}.

3. Proper names. Examples: Banco de México {The Bank of Mexico}.

4. Connective phrases, used to connect ideas in a sentence. Examples: en base a {based upon}, sin embargo {nevertheless}. These are classified as idioms since for instance the meaning of *sin embargo* cannot be derived from its constituents *sin* {without} and *embargo* {seizure, foreclosing}.

Tagger. It assigns tags or labels (verb, noun …) to each word in a phrase (Tagger TNT, Internet reference 7). Many current systems process natural language using a morphological tagger. The TNT tagger assigns a single tag (grammatical category) to each part of speech. This tag is used in subsequent levels of analysis, such as the syntactic parser. Since idioms are processed before the tagger acts (see Figure 1), tagging becomes better because idioms are already correctly tagged (by the idioms analyzer).

Usually, there are two ways to achieve morphological tagging. Recent comparisons in training sets and corpuses have shown that in most cases the taggers that use statistical approximations produce better results that finite state taggers, or those based in rules, or using an extensive a priori storage. The tip of the balance has not yet settled.

Anaphora solver. It finds and replaces the pronouns by the corresponding linguistic entity. It is AnaPro, the subject of this article.

Lemmatizer/Stemmer. It finds the lemma or root of each word. More details in [8].

Prepositions disambiguator. Reference [7] describes an algorithm and its implementation to disambiguate prepositions in Spanish phrases. This article focuses on the design of a preposition disambiguator, using a verb and noun lemmatizer that employs frames to identify the meaning of verbs and nouns. It finds the semantics of the prepositions contained in parts of speech. For instance, for the preposition *de* {of}, it finds that *de* in *olla de madera* {wooden pot} means "made of"; in *olla de arroz* {pot of rice} means "contains", whereas *olla de Oaxaca* {pot from Oaxaca} means place of origin.

Prepositions are words used to relate ideas composing a sentence; hence, by themselves, they lack expressive meaning of their own, but they contribute in a fundamental manner to provide sense to the whole sentence since they establish a semantic relation. Many times, prepositions are wrongly used and they form part of the vices of a language.

This work studies the separable prepositions, which are *a* {to}, *ante* {before}, *bajo* {under}, *cabe* {it should be noted}, *desde* {from}, *contra* {against}, *en* {in}, *por* {by}, *según* {as}, *sin* {without}, *entre* {between}, so {under}, con {with}, hacia {towards}, sobre {upon}, hasta {until}, de {of}, tras {behind}, para {for}.

Prepositions are classified in groups according to the sense they provide to the expressions they qualify, although some prepositions belong to more than one group:

[1] AnaPro is an analyzer of anaphora in Spanish. In order to help some readers, the authors (not AnaPro) have added translations to English for selected Spanish sentences. They can be omitted.

1. Place prepositions: They assign a place: a {to}, de {from}, en {in}, entre {between}, hacia {towards}, por {by}, tras {behind}.
2. Time prepositions**:** An eventis placed on the time line: a {to}, con {with}, de {of}, desde {since}, en {in}, para {before}, por {by}, sobre {after}.
3. Causal prepositions: They provide a link between origin and result: de {of}, por {by}.
4. Purpose prepositions: They determine a target or objective: a {to}, para {for}.
5. Company prepositions**:** They express association of subjects with respect to an action: con {with}.
6. Instrument prepositions: They express the means by which the action is or has been performed: a {to}, con {with}, de {of}, en {in}.
7. Modal prepositions**:** They indicate the way in which the action is carried out: a {to}, con {with}, de {of}, en {in}, por {by}.

More details for the algorithm to disambiguate prepositions are in [7].

Word disambiguator [9]. Mapping words to concepts. This work determines the correct meaning of ambiguous words in Spanish texts, using the context in which the ambiguous word is found. It presents a new non supervised computational approach to process ambiguous words in Spanish documents. It selects the appropriate word sense based on the context. The process is also known as *Disambiguation* or *Word Sense Disambiguation*. The model is based on the Lesk Algorithm with significantly modifications that enhance its performance. This approach does not need a previous markup of the text to analyze. The software detects automatically the ambiguous words within the text and defines its correct sense. The results obtained are evaluated with the highest international criteria with significant success.

To achieve its objective, the method of word coincidence was employed using:

1. A dynamic (moving) window of varying width as context, formed by words to the left and right of the ambiguous word.
2. The definitions (different senses) of the ambiguous word.
3. A lemmatizer applied to the words in the window and to the words in the definition of the ambiguous word.

The algorithm does not require supervision or training in order to work. It also does not require a set of tagged texts, nor the listing of the ambiguous words to be found.

Frames [17] give relevant details of concepts, generally nouns (agents) and verbs (actions). They are data structures containing semantic information about the concept: such as what it is, its use, its parts, when it happens, and so on. Frames help the prepositions disambiguator, the disambiguator of nouns and verbs, and the syntactic/semantic analyzer. Frames were perhaps introduced by [17], who defines them as data structures to represent generic situations, such as the representation of a type of behavior and things expected in a living room, in a restaurant, in a children's party. Several kinds of information are found in frames: how to use it, what activities or event sare performed there, what other events occur after the frame occur after the frame (a subsequent frame). A frame can be considered as a network of nodes and relations. Frames represent scenes, for instance: the frame representing a children's party has more semantics that the definition in a dictionary: [fiesta -- Reunión de gente para celebrar algún suceso, o simplemente para divertirse (Diccionario de la Real Academia Española). party: a social gathering; also: the entertainment provided for it. (Webster Dictionary)]. Table 1 elaborates the frame "children party".

Many concepts may be related to a frame, and they will enrich it semantically; therefore, most of the information intertwined in the represented scenario (concept, event, or activity) must be present in the frame. There is no standard notation about frames; we have proposed one. It is formed by:

1. Header, representing the name of the concept, a character indicating if it is a noun (s, for sustantivo noun}), verb (v), etc., and in parenthesis, its gender.
2. Body, containing elements such as: quees {what it is; its meaning}, agenteactivo {active

agent, who initiates or carries out the action}, *agentepasivo* {passive agent, who suffers the change or consequences of the action provoked by the active agent}, *hiperonimia* {hypernym, concept of which the frame is a subset}, *sinónimo* {synonym, other words used for this frame}, *hiponimia* {hyponym, a particularization of this frame, a frame which is a subset of this frame}, *comoseusa* {how it is used; example of the concept in a sentence}, *frasetemat* {idiom, colloquial sentence, where the word is used with a different meaning}.

3. Our notation allows adding semantic elements as needed, for instance the element *lugar* {place}.

More details in [7] and [8].

*Dress***:** better than everyday dresses.
Gift: Must be wrapped and given to the birthday child.
Celebrating hide-and-seek, guessing riddles, jokes.
Decoration: balloons, figures made of crepe paper.
Cake: candles, wishes, to blow.
*Songs***:** Happy Birthday.
*Place***:** Party room, home, kindergarten.

Table 1. Part of the frame "children's party"

Syntactic/Semantic parser. It analyzes each phrase, taking into account the syntax, as well as the semantics of each part of speech (given by the frames, the disambiguated prepositions, nouns, etc.). It finds the correct dependency between nouns, verbs, and articles. Basically, it is divided into three parts:

Syntactic Parser. We use FreeLing [24]. It builds a dependency tree mostly based in the syntax of each sentence.

Congruency detector. It analyzes each part of the dependency tree, and, with the help of the frames and other semantic elements, it measures how consistent that part is (how much sense it makes).

Generator of plausible moves. The parts of the dependency tree with low congruency are replaced by other candidates, which the generator proposes. Several changes of node position in the dependency tree are suggested, directed by the frames and measured by the congruency detector. The congruency of the resulting replacement should be larger than the replaced portion of the tree.

The congruency detector and the generator of plausible moves are under construction in Lisp.

SERCDD [13] is an analysis system that extracts information from "descriptive documents": documents describing an object (a tool, a planet), and uses the concepts extracted to form a semantic network. Its output is an ontology in the OM language, denoting the object described in the document. The analysis method starts with text which has previously suffered tagging and lemmatization. Using the structure of the sentences found in the text, analysis proceeds trying to identify the relations present in it. For instance, a text describing a carpenter tool will usually contain its definition, a description, common uses, parts and materials forming it, as well as the classification of such tool. The result is a formal representation of the extracted knowledge, embodied in an ontology written in the OM language, described in [2]. In order to produce such analogy, it is necessary to identify the entities (concepts), relations and properties described in the original text.

SERCDD finds ontological relations (hyponymy, hypernymy, part of, synonymy, size, made of, useful for…) by searching for text patterns and using a set of rules derived from the respective semantic relations. For the time being, these patterns and rules are manually constructed, but another part of OM* seeks to automatically derive them from the frames representing the semantics in the objects. The system extends the use of patterns and rules, which are usually used just to find hypernymy relations (the hammer is a tool, the mechanic hammer is a hammer…), to many other relations.

SERCDD processes documents written in Spanish, but it is easily extensible to other relations and other natural languages. It performs tagging and lemmatizing, prior to seeking ontology relations.

Ontology generator. This is a straightforward module that converts the correct dependency tree produced by the syntactic/semantic parser in an ontology expressed in OM notation. It has not been built yet.

Ontology Fusion. In order to compute intelligent answers to complex questions, using the vast amounts of information existing in the Web, computers have (1) to translate such knowledge, typically from text documents, into a data structure suitable for automatic exploitation; (2) to accumulate enough knowledge about a certain topic or area by integrating or fusing these data structures, taking into account new information, additional details, better precision, synonyms, homonyms, redundancies, apparent contradictions and inconsistencies found in the incoming data structures to be added; and (3), to perform deductions from that amassed body of knowledge, most likely through a general query processor.

OM [2, 3] solves point (2) by providing an algorithm to fuse two ontologies (coming from Web documents) without human intervention, producing a third ontology containing the knowledge residing in the input ontologies. It resolves synonyms, homonyms, articulates partitions, detects and solves some contradictions (inconsistencies), and removes redundant relations. Also, the accumulation of knowledge requires that a fair amount of ontologies about the same topic should be merged, which calls for repeated use of OM.

The correct merge must consider not only the syntax of the word and phrases describing the concept, but the semantics of the concept thus represented, too [its ‘‘meaning” as described by its relations to other concepts in the source ontologies, its similarities to other concepts, superficial likeness, homophones, etc.] Therefore, the representation to be used (OM Notation) should allow the characterization of these relations.

OM Notation. [2] proposed this ontology definition language with the goal to represent concepts and relations so that more semantics is captured in them. OM means “ontology merging,” since the language was first used to fuse or merge ontologies. It is a derivative of XML, described in [3].

The OM language structures ontologies through the following tags:

<concept>, used to define a concept. This tag allows concept nesting.

<language> represents a brief sentence (a *gloss*) in natural language describing the concept. It is used mainly for human consumption, not by the OM processor.

<word> is followed by that brief sentence or gloss. For instance, the concept *escalpelo* {scalpel} can be defined as

<concept> *escalpelo*

<language> Spanish <word> *escalpelo*, *bisturí* </word> </language>

<language> English <word> scalpel, a small straight thin-bladed knife used especially in surgery </word> </language>

… *here will go the relations pertaining to escalpelo* </concept>.

<relation> represents the type (name) of a relation (feature, property, usually obtained from a verb) that connects two concepts.

1.2 AnaPro

This paper describes AnaPro, one of the modules of OM*. AnaPro determines the direct anaphora (that resolves pronouns). Its output eases the building of the part of the ontology that corresponds to the text being analyzed. For instance

*Juan tomó una **botella** en la **tienda**. **Ésta** estaba rota.*

{John took a **bottle** in the **store**. **It** was broken. }

A person easily finds that the broken thing was the bottle, not the store, but a program may have difficulty to find out to which of the nouns *botella* {bottle} or *tienda* {store}, the pronoun **ésta** {it, this} refers to. In the example, a person asks herself: *what can be broken?* The bottle, most likely. This is so because a person has common sense, experience, previous knowledge, etc. For the computer, analysis and verification of several information sources and tools (frames, dictionaries, thesaurus, etc.) is usually needed in order to have a satisfactory solution.

Anaphora resolution has been an important task for NLP during years. Because of its importance, the problem has been tackled from several fronts. For instance, [10] uses semantic criteria, as described in §3. Since our work is for Spanish documents, our descriptions of previous works will be primarily focused on Spanish, too.

[22] and others have proposed methods that try to resolve anaphora resolution (both direct and indirect) in Spanish, using dictionaries. Their methods do not solve all grammatical forms.

Nevertheless, simple methods using limited knowledge can be applied in many cases with satisfactory results. For instance, text from a restricted domain or culled from a controlled vocabulary. An example is [20], who solves anaphora with the help of the SUPP (Slot Unification Partial Parser) analyzer, which provides lexical and syntactic information to find out the antecedent of higher relevance in the list of antecedents (one of them is the referent). Although with some limitations, the work proposes interesting improvements to the algorithm, as described in §3.

Another approach suggests to use linguistic and structural knowledge which are domain-independent, as in [11]. It deals with anaphora in Spanish identified by third person pronouns and adjectival anaphora. Described in §3, it uses a statistical model to determine the complements of verbs, adjectives and some nouns, based on a corpus.

An important work, since it deals with unrestricted domains, is that of [4]. It locates and extracts information where answers to questions can be inferred. It does this in Spanish and in English. It is briefly described in §3.

AnaPro only solves direct anaphora, particularly pronoun resolution. Of the already mentioned previous works, only [11] resembles our solution.

The paper is hence organized as follows: §1 describes the state of the problem and related previous work; §2 gives the theoretical framework; §3 presents work related to resolution of direct anaphora and places AnaPro in this context. §4 details our solution and solves some examples. §5 explains how AnaPro works; §6 provides more examples taken from “real life” (Internet documents).

2. Theoretical Framework

Anaphora resolution has been the focus of attention of philosophers, linguists, cognitive and artificial intelligence scientists, psycholinguists and computational linguists. This is important because the anaphora:

- It is one of the most complex phenomena in natural language.
- It has been shown that syntactic, semantic and pragmatic factors take part in it.
- Its resolution is needed in a wide range of NLP tasks: natural language interfaces, document understanding [for the computer to answer complex questions about the knowledge that lies in them (this is the goal of the aforementioned Project OM*)], machine translation, information extraction, automatic summarization, finding differences or contradictions between statements on the same subject in two documents, among others.

Two important types of anaphora exist: direct and indirect anaphora.

Direct anaphora associates a pronoun or reference to some entity already mentioned in the text. For instance, *los alumnos estaban contentos pues ellos habían pasado a la semifinal* {students were happy because they had passed to the semifinal}; here the pronoun *ellos* {they} refers to *los alumnos* {the students}.

Indirect anaphora establishes an associative link between a linguistic entity (word or phrase) and an entity previously introduced through the text or the speech [22]. For example, *Eliseo entró a un circo, el payaso era muy simpático* {Eliseo went to a circus, the clown was very nice}. In this example the discourse focuses on the circus and it is the previous entity or antecedent to which *payaso* {clown} belongs or refers. A connection exists between *payaso* and the circus where Eliseo went, which has at least one clown. This information comes from the extension of the context by encyclopedic knowledge (common sense); as a consequence, the whole sentence renders a consistent interpretation, as implied by the relevance principle, used above, or by the scenario approach (frames, [17]), where the use of *circo* {circus} invokes a scenario (the frame of circus) that implicitly contains at least one clown.

AnaPro does not analyze indirect anaphora; this paper does not deal with it. It is mentioned here for completeness.

2.1 Contributions

AnaPro converts a Spanish text into another text without direct anaphoric references. The anaphors are solved, replacing for instance *ellos* {they} by *estudiantes* {students}. The final text, without anaphoric references, is an input to be used in project OM*. Some of its contributions are:

1) The focus of AnaPro is to find (automatically) the noun to which a determiner refers. Other works such as [22] require a user. Instead, [20] needs an information detection system to determine the most relevant preceding topics.

2) Another important point of AnaPro is that it resolves anaphora in complete texts, not in short dialogues [20]. Therefore, full text can be entered into the tool for its analysis and resolution.

2.2 Determiners

Determiner is a term that denotes the lexical unit that precedes a noun in a nominal phrase to specify its reference, including the quantity of the noun. For instance,

1.-Todos esos vehículos están en venta. {All those vehicles are for sale}

The determiner is ***Todos*** {all} which determines the quantity, and ***esos*** {those} is a demonstrative pronoun that indicates the place, relative to the emitter, where the vehicles are.

In general, the determiners give rise to the definite expressions where these parts of speech are used: the definite articles (*el*, *la*, *lo*, *las*, *los*) {the, it}; the demonstratives (*aquel*, *aquella*, *aquellas*, *aquellos*, *esa*, *esas*, *ese*, *esos*, *esta*, *estas*, *este*, *estos*, *tal*, *tales*, *semejante*, *semejantes*) {that, that, those, those, that, those, that, those, this, these, this, these, such, such, like, like}; the possessive determiners (*cuya*, *cuyas*, *cuyo*, *cuyos*, *mi*, *mis*, *nuestra*, *nuestras*, *nuestro*, *nuestros*, *su*, *sus*, *vuestra*, *vuestras*, *vuestro*, *vuestros*, *tu*, *tus*, etc.) {whose, whose, whose, whose, my, my, our, our, our, our, his or her, his or her, your, your, your, your, your, your, etc.}, and the quantifiers (*todo*, *algún*, *cada*) {all, any, each}. With any other determiner, an expression is considered an indefinite expression. This is important since in Spanish the *artículos indeterminados* (articles a, an) give rise to indeterminate expressions.

2.3 Reference resolution process

The task of whoever receives the information is to conduct the reference resolution process, which is achieved with the following sequence of steps:

1. To identify as accurately as possible the entity to which the emitter (or writer) is referring (the referred) with the referential expression.

2. To determine if this entity has already been previously mentioned (or referred to) in the context of the discourse, or if it is new.

3. If it has already been mentioned (coreference), it will establish the link between referential expressions in the linguistic context; the first entity already mentioned has the link to the entity in the discourse context.

4. If it is new, it should be integrated it as part of the linguistic context, creating the link from the referential expression to the entity in the linguistic context.

The whole process must be based on the morphological and syntactic features of the text, which involve the use of determiners, and where the concepts of referential expression, coreference and direct and indirect anaphora (already defined in §2) are closely related.

3. Related work

For the English language, there is plenty of work in anaphora resolution. Since AnaPro works for Spanish texts, we will not review many of those works. A comprehensive analysis of the state of the art can be found in chapter 21 of [6]. We also refer the reader to the excellent surveys by [23], and, more recently, [19]. Some interesting anaphora solvers for English are reviewed here.

- The *Resolution of Anaphora Procedure* (RAP, [15]) identifies the noun phrase antecedents of third person pronouns and lexical anaphors (reflexives and reciprocals). Written in Prolog, its inputs are the syntactic representations generated by McCord's Slot Grammar Parser. RAP focuses its attention on (a) the syntactic structure of the sentence (as given by the Grammar Parser), from which it extracts salience measures, and (b) a rather simple intentional model for the attentional state. These are used to select, from a list of candidates, the noun corresponding to the pronoun. In evaluating candidate antecedents, it does not employ semantic conditions (other than those implicit in grammatical number and gender agreement) or real-world knowledge. Also, RAP does not model intentional or global discourse structure. RAP has been tested and has an 86% efficiency; it exhibited a slightly higher rate of success (by 4%) than Hobb's algorithm. Its authors enhanced RAP with statistically modelled information about semantic and real-world relations, but these enhancements only caused a slight improvement (by 2%) in the algorithm's performance. The main components of RAP are:

  o Syntactic filter. It rules out anaphoric dependence of a pronoun on an NP on syntactic grounds.

  o Morphological filter. It rules out anaphoric dependence of a pronoun on an NP due to disagreement in person, number, or gender.

  o Pleonastic pronouns. A procedure to identify pleonastic (semantically empty) pronouns.

  o Lexical anaphors. An algorithm that identifies the possible antecedent binder of a lexical anaphor (reciprocal or reflexive pronoun) within the same sentence.

  o Salience evaluator. Values are assigned to several salience parameters of an NP: grammatical role, frequency of mention, proximity, sentence recency, parallelism of grammatical roles.

  o Equivalence class identifier. Anaphorically linked NPs are identified and placed into an equivalence class. For this class, a global salience is computed adding the individual saliences of its members.

  o Decision procedure. It selects the preferred element from a list of antecedent candidates for the pronoun.

Compared to RAP, AnaPro has a simpler design, which nevertheless works quite well for Spanish texts.

- The work *Anaphora for everyone* [1] extends the previously described work of Lapping and Leass, producing similar results. The main difference is that it does not require an in-depth, full, syntactic parsing of the text. Instead, it uses a part-of-speech tagger, enriched with annotation of the grammatical function of the lexical terms found in the input sentence. Thus, it can be used in a wide range of text processing frameworks for which there are insufficient or inexistent full syntactic parsing capabilities. It requires the tagger to provide, in addition to the tag, an integer for each token in the input sequence (the *offset* of such token). For instance, for the input sequence "For 1995 the company set up ...", the output of the tagger is

```
"For/off139" "for" PREP @ADCL
"1995/off140" "1995" NUM CARD @<P
"the/off141" "the" DET CENTRAL ART
SG/PL @DN>
"company/off142" "company" N NOM
SG/PL @SUBJ
"set/off143" "set" V PAST VFIN
@+FMAINV
"up/off144" "up" ADV ADVL @ADVL
...
```

For anaphora resolution, the primary input data set is a complete list of all noun phrases, reduced to modifier-head sequences. The *offset* indicates the position of the noun phrase in the text, hence providing important information for precedence relations. The algorithm also uses information about the syntactic contexts in which the noun phrase appears. These observations are obtained by the use of patterns that detect nominal sequences that occur locally to adverbs, relative pronouns, and complementing sequences. These patterns are regular expressions. In terms of overall strategy, the resolution procedure follows that of [15].

The most common types of errors of the algorithm consists in gender mismatch (35% of the errors) and certain long range contextual (stylistic) phenomena, for instance, quoted passages in-line (14% of the errors). The authors correct these mistakes by implementing a gender disagreement filter. Such errors simply reflect the lack of a consistent gender slot in the tagger output.

AnaPro resembles this work in several aspects, notably in the use of a tagger and regular expressions for extracting additional positional and relative information.

- The Anaphora solver of [21] is notable because it works using the Wordnet ontology coupled with heuristic rules. It identifies both intra-sentential and inter-sentential antecedents of anaphors. The judicious analysis of the hierarchical relation of nouns and verbs in the surrounding text supplies information about animacy (animacy denotes the living entities, notably persons and animals, which can be preferred by some gender-marked pronouns). This is important, because the identification of animacy entities and of the use of pleonasms (for instance, pleonastic pronouns that do not have an antecedent to refer to) contributes to a higher resolution accuracy. The main procedure has these steps:

    - The text is parsed by a POS tagger, which provides essential information (sentence offset, word offset, word POS, base form, etc.) for subsequent processing.

    - (a) An NP finder module finds base noun phrases in each sentence. Also, (b) capitalized nouns, if found in a name gazetteer, identify person names. (c) Number agreement is implemented on the head noun. (d) Gender feature is attached to each name, if uniquely rendered. WordNet is useful here to provide gender clues. If the noun can be masculine or feminine, the gender attribute is ignored.

    - Anaphors are checked sequentially from the beginning of the first sentence. Also, pleonastic-it is checked, to avoid its resolution.

    - The remaining noun phrases provide antecedent candidates for resolution, which are identified using predefined window sizes. They are further refined by gender and animacy agreement. WordNet is useful here for the recognition of animate entities, which are usually those hyponyms of {animal, fauna} and {person, human being}. The procedure is not so straightforward since a noun may have several senses. WordNet is also used for finding hyponyms of certain verbs starting from the synsets {cognition}, {communication}, {emotion} and {social}.

    - The remaining candidates are evaluated using heuristic rules. They are of two types: preference rules and constraint rules. A formula assigns a score to each candidate, taking into account the candidate noun phrase, the anaphora to be resolved, the agreement between number, gender and animacy, and the weight of the rule.

3.1 Anaphora resolution for Spanish texts

There are some studies that have addressed the problem of identifying anaphoric references in Spanish texts. The focus of those studies, their resolution algorithms, and their limitations are briefly mentioned.

- The Open Domain Question Answering Systems [4] are tools capable of getting concrete answers directly from natural language documents over unrestricted domains, in response to precise information needs from users. At least, they try to find and extract those areas (of the documents) from whose contents an answer to a specific question can be obtained (by hand). Therefore, those systems try to reduce the time invested by users to find concrete information. They have three modules. The first module is a standard Information Retrieval system, collecting documents that may be relevant to the query. The second module, which uses SUPAR (Slot Unification Parser for Anaphora Resolution[2]), has as input the queries and the retrieved documents. The third module is a Question Answering system, which also interacts with SUPAR. It divides the sentences of the documents in order to find those where the correct answer appears. It is in this second module where the pronominal anaphora is resolved, using a lexical analyzer, a syntactic analyzer and a module that solves

[2] SUPAR also has three modules: lexical analysis, syntactic analysis, and linguistic problem resolution.

anaphors, extra-position, and ellipsis, when the result is a slot structure (refer to the description of Martinez' work at the beginning of Section 3) corresponding to the input sentence.

AnaPro does not use slot structures. Its resolution mechanism is detailed in next section.

- [18] presents an algorithm for resolution of the anaphora of a phrase (for third person pronouns, PP_R), demonstrative pronouns, DP_R, reflexive pronouns, RP_R and missing pronouns (cero pronouns, OP_R) in a prepositional phrase, PP, or if they also complement personal pronouns (clitic pronouns) in non-restricted Spanish texts. There are restrictions and preferences for the different types of nominal expressions. In the resolution of Spanish anaphora, different kinds of knowledge are considered: lexical, morphological, syntactic and linguistic. It also gives a partial parsing for pronouns that are not coreferential. The algorithm has been tested on a corpus of 1,677 pronouns with a success of 76.8%. The algorithm can be easily extended to other languages such as English, Portuguese, Italian, and Japanese.

The algorithm finds the different types of anaphora from left to right, as they occur in the sentence. The main steps of the algorithm are:

1. Identification of the type of pronoun.
2. Restrictions (a) morphological (person, gender and number); (b) syntactic conditions in NP –non-correferential pronouns.
3. Preferences. It is defined as a characteristic or feature that does not always satisfy the solution of an anaphor (it is sometimes unable to resolve it).

Unlike other works that resolve some types of anaphora using restrictions, and other types using preferences, this work uses them together, since the analysis flows from left to right in the sentence. For instance, for pronominal anaphora, the antecedents need to match person, gender and number. Otherwise, they are discarded.

This work is quite close to AnaPro; the main difference is that AnaPro uses somewhat simpler mechanisms.

- [20] uses as input the slot structure generated by the SUPP analyzer. This structure stores lexical, syntactic, morphologic and semantic information of each element of the grammar, and it also includes an anaphora detection mechanism, as well as another mechanism for its resolution. This last method uses the information from the topic detection system in order to determine the most relevant antecedent among the general topics list. The topics information joins the information obtained from other sources to get the best antecedent. In the same manner, once the anaphora has been resolved, the antecedent occupies the place left by the expression. This system works via dialogues.

Next, it is a simple example in which Martínez' algorithm [20] is applied to the resolution of anaphora. It is a small dialogue where the system of topics defines three spaces for anaphoric accessibility: a) the space of the actual intervention, b) the space of the previous intervention, and c) the relevant topic in the current dialogue space. The first and second spaces are directly determined thanks to the labeling of turns characteristic of dialogues. The third space is identified by two coefficients that compute the use or lack of use of an entity; that is to say, the importance or irrelevance of an entity occurring in the current intervention.

The algorithm uses three lists to store the entities that can be an antecedent (according to the previously defined spaces). The *list of current local entities,* Fa, contains the entities occurring in the current intervention. The *list of previous local entities,* Fa', contains the entities that occurred in the last intervention. The *list of general topics,* F, contains the entities relevant in the dialogue, evaluated according to frequency and distribution.

T01: <H1> Buenos días. {Good morning}
T02: <H2> Buenos días. ¿Qué deseas? {Good morning. What do you want?}
T03: <H1> Quiero *manzanas*. {I want *apples*}
T04: <H2> ¿De qué clase *las* quieres? {What kind do you want *them*?}
T05: <H1> No importa si son buenas. {It does not matter as long as they are good}
T06: <H2> *Éstas las* he recibido *esta mañana*. Son muy buenas. {I have received **these this** *morning***.** They are very good}

The algorithm starts by considering turns T01 and T02 as continuation turns since they do not contain any entity. Thus, the three lists remain empty until T03. When processing T03, *manzanas* {apples} is considered an entity, and is hence introduced in the list of current local entities Fa:

Fa=[manzanas] {apples}

T03 is therefore considered an intervention. When it finishes, the entities in Fa are incorporated into the list of general topics F with their corresponding weight, and they are also stored in the list of previous local entities Fa'. After these operations, the lists are:

Fa=[ ]
Fa'=[manzanas] {apples}
F=[(manzanas,10)] {apples, 10}

In T04, after the incorporation of *clase* {kind} as an occurring entity, the first anaphora is detected, as the pronoun *las* {them}. At this point the lists are:

Fa=[clase] {kind}
Fa'=[manzanas] {apples}
F=[(manzanas, 10)] {apples, 10}

To resolve the anaphora, the algorithm searches, first, an antecedent in the list of current local entities (Fa). It finds the entity *clase* {kind}; however, the intervention of the system of restrictions rejects it, since it does not meet the required morphological characteristics (it is looking for a plural antecedent). In this way, since Fa does not contain more candidates, the algorithm will look into the previous local entities list, Fa', finding manzanas {apples}. After restrictions are applied, manzanas is proposed as antecedent for the pronoun and it is included as an entity in Fa:

Fa=[clase, manzanas] {kind, apples}

After processing the first sentence of T04, Fa' is emptied:

Fa'=[ ]

And when T04 is finished, the lists are:

Fa=[ ]
Fa'=[clase, manzanas] {kind, apples}
F=[(manzanas, 20), (clase, 10)] {(apples, 20), (kind, 10)}

Since T05 is a continuation turn, the lists do not change. After T06, the weight of *manzanas* increases because of new appearances of this instance (introduced by the pronouns), while the entity *clase* {kind} loses weight due to the opposite:

Fa=[ ]
Fa'=[manzanas, esta mañana] {apples, this morning}
F=[(manzanas, 40),(esta mañana, 10), (clase, 9)] {(apples, 40), (this morning, 10), (kind, 9)}

We can see that the anaphora is resolved returning *manzanas* as a valid result, according to the coefficient that indicates the frequency of occurrence of entity *manzanas*.

A shortcoming is that it does not incorporate semantic information. Thus, it cannot rule out antecedent candidates, and it proposes the use of Spanish Wordnet (Internet reference 6) as lexical tool, to learn semantic patterns between subject and verb, and between verb and object. It does not solve any other kind of references in the dialogue.

The relevant feature of AnaPro (our work), when compared with the work of [20], is that AnaPro resolves anaphora in texts and not in short dialogues.

- [22] focuses on the anaphoric references, both direct and indirect. His resolution algorithm is based on the use of a thesaurus and on the detection and contextualization of coreferences, in such a way that not all the grammatical forms are processed by the algorithm, but through an entity exclusion process, irrelevant information is thrown away.

The algorithm discovers the existing interrelation between the coreference phenomena (direct and indirect) and the indirect anaphora, and that both use nominal referential expressions to manifest their presence.

It determines the evaluation order required to discriminate the phenomena that are used as basis to detect the presence of the indirect anaphora.

It develops a method based in the scenario model of the linguistic context that models the human reader, needed for the resolution of the indirect anaphora and the coreferences. But one of its shortcomings is that the additional information needed is not automatically obtained. When the system was tuned to use it with free text, its precision was 51%, while the human expert obtained a global average of 60%. With the linguistic context based on the scenario model, it achieves the automatic resolution of the nominal indirect anaphora, with a strong dependence on the additional information added.

- [11] proposes the development of a resolution system for anaphora generated by third person pronouns and adjective anaphors in the dialogues (this work is the closest to our algorithm). The system is based on a syntactic analyzer that uses an extended grammar, context independent, with unification elements. This program incorporates the results of the research to obtain handling patterns for verbs, adjectives, and nouns from Spanish. These results allow quantitative classification of the variations generated by the analyzer, using the weights assigned to the variations, following the values of the subcategorization combinations. The weights of the subcategorization combinations are the result of a syntactic analysis process and of an extraction following a statistical model that allows determining the complements of verbs, adjectives, and some nouns from Spanish, starting from a text corpus.

- [10] approaches the anaphora in Spanish and resolves it through a sequence of semantic criteria, generating an abductive theory (starting from facts, an hypothesis is generated) for the pronominal anaphora. They propose the following resolution procedure:

  1. Obtain the grammatical form of the sentence and determine the main verb.
  2. Incorporate the non-anaphoric referents that show up in the analysis into the context.
  3. Determine the anaphoric terms and their features.
  4. Determine, as a function of the type of argument that the anaphora occupies with respect to the main verb, its thematic role.
  5. Look for possible referents for the anaphora, among those occurring in the same and former sentences. Each one must pass three filters:
     a. Grammatical agreement (it is the syntactic consistency criterion).
     b. Agreement between the thematic roles (preferential criterion).
     c. Semantic coherence of the resulting sentence (a mixture of preferential and semantic consistency criteria).

Finally, the paper presents a solution, although it requires more experimentation to test the hypotheses presented.

As we can appreciate, anaphora resolution is a problem not yet fully solved. Hence, we have proposed and implemented a solution based in pattern identification and determiners enumeration, which we now show.

4. The architecture of AnaPro

AnaPro works in five stages.

Stage 1*:* It receives a text tagged with EAGLE tags [5]. The tagged text is stored in a structure in the main memory, and the nominal expressions are identified.

Stage 2*:* The pronominal syntagmas are identified, and the cases of nominal ellipsis are stored.

Stage 3*:* The direct anaphoric references are identified.

Stage 4*:* The anaphoric references go through a filter of grammatical restrictions. Here, candidates (for resolution) are identified. If ties exist, criteria of preferences in the assignment apply so that only one candidate is chosen for each anaphoric reference.

Stage 5*:* In the text, all pronouns solved in stage 4 are replaced by the chosen candidates.
We give a detailed description of the stages with the help of an example.

4.1 Stage 1

This stage reads the tagged text, storing each word and their tags in separate structures, which are numbered. Also, the nominal expressions are obtained, through the use of the module “To obtain the nominal expression” of AnaPro, which is explained below.

A portion of text “*La batalla de Puebla*” {Puebla’s battle} (Internet reference 3) shows AnaPro in action.

“La batalla de Puebla.
Aunque se considera en todo el país, ésta se celebra especialmente en Puebla, México, y en los estados del Sur de Estados Unidos” {Puebla’s battle. Although considered in all the country, it is celebrated especially in Puebla, Mexico, and in the southern states of USA}

See Tables 2 and 3 for the result of Stage 1 for this example.

Tagged and numbered text		
AD0FS00 = la {the}	AD0MS010 = el {the}	NCMS00020 = México
NCFS0001 = batalla {battle}	NCMS00011 = país {country}	Fx21 = ;
AC0MSP2 = de {of}	Fc12 = ,	CC22 = y {and}
NCFS0003 = puebla {Puebla}	PD0FS00013 = ésta {it, this}	SPS0023 = en {in}
Fp4 = .	P030000014 = se {is}	AD0MP024 = los {the}
CS5 = aunque {although}	VMSI3S015 = celebra {celebrate}	NCMP00025 = estados {states}
P00000006 = se {it is}	RG16 = especialmente {especially}	SPCMS26 = del {of the}
VMIP3S07 = considera {considered}	SPS0017 = en {in}	NCMS00027 = sur {South}
SPS008 = en {in}	NCFS00018 = Puebla	SPS0028 = de {of}
DI0MS09 = todo {all}	Fc19 = ,	NCMP00029 = Estados {States}
		AC0MPP30 = Unidos {United}

Table 2. Shows the tagged text, numbered and distributed in three columns

Table 3 shows the nominal expressions generated from the tagged and numbered text.

Nominal expressions generated
NCFS0001 = la batalla de puebla {Puebla’s battle}
NCMS00011 = todo el país {all the country}
NCFS00018 = Puebla
NCMS00020 = México
NCMP00025 = los estados del sur {the southern states}
NCMP00029 = Estados Unidos {United States}

Table 3. Nominal expressions identified in the text. The first characters of the tag mean: N (noun), C (common), F (female), S (singular), M (masculine), P (plural). The table shows the order in which they occur. The tagging is done by the tagger TaggerFT [12] in a preprocessing step. The nominal expressions are generated by the tagger, and we only use several filters to identify and delimit them correctly. They are shown here

Module “To obtain the nominal expression”

1. *Each tagged word is numbered (by the tagger).*

2. *Using the tags, pronouns are identified as well as their determiners and auxiliary types, through the comparison between each tag and regular expressions, be they a pronoun, noun, adjective, etc.*

   For instance, to look for pronouns in the text, which will be tagged as pronominal anaphoric references, the expression shown in Figure 2 *is used.* Figure 2 *is expanded to* Figure 3*, showing that a pronoun must match with one of these expressions. It is required that a pronoun must match with the first two of one of the expressions of* Figure 3*. The remaining eight symbols in the regular expression will define with further detail each type of pronoun, such as its gender and number, which is also considered when searching for candidates.*

Regular expressions are used to identify pronouns, articles, nouns, adjectives and in some cases, to distinguish among each particular pronoun, to ascertain its gender and number in order to find pairs of pronouns and nouns. Of course, we could find the nominal expressions from direct analysis of the text strings, but we find it easier to analyze the tags.

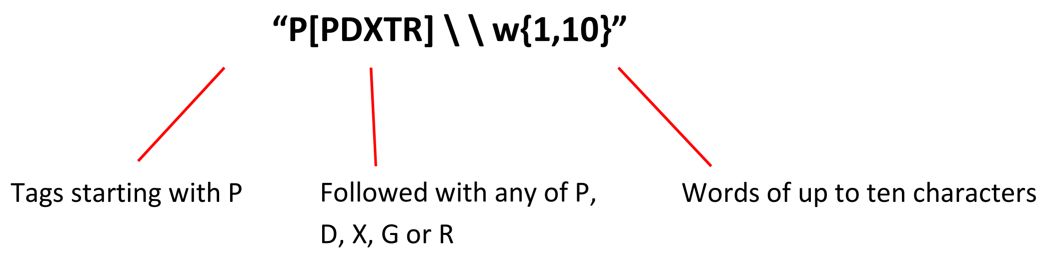

"P[PDXTR] \ \ w{1,10}"
Tags starting with P
Followed with any of P, D, X, G or R
Words of up to ten characters

Figure 2. Regular expression used to find pronouns in a text. It is divided in three parts: the start of the tags, the following letters, and the length of the words

Once the pronouns or anaphoric pronouns are identified, AnaPro locates the previous portion of text where the possible nouns or determiners may exist, those to which the pronoun refers. This area or range is known as *nominal expression*, which contains the anaphora and its possible determiners. The name *nominal expression* was changed to *sentence* in the implementation of AnaPro, for practical purposes.

```
PP********
PD********
PX********
PT********
PR********
```

Figure 3. Tag types, where asterisks indicate optional areas, and the first two characters indicate the pattern to look for. The standard requires that the tag of a pronoun should have at least eight elements, and then we assign an index, which could have from 1 to 10 digits. In this step, the algorithm does not take into account the last 6 or 8 digits, since they provide no information to our tool

Function to generate nominal expressions. This part of AnaPro, instead of dealing directly with nouns, starts by correcting labeling errors "on the fly", such as the exchange of a noun label by a quality adjective, in the case of a surname or a name. At the same time, the first step is always to find the nucleus. The process starts going through the text from start to finish in a single cycle. When the first noun nucleus is found, it stores the noun and its tag, then goes back looking for (jumping over phrases or sentences, moving backwards from the found noun) a determiner article, qualifying adjective, number and other modifiers.

These are sequentially added as they are found (using a window of at most five elements, which in our tests has never been exceeded). The cycle repeats going forward from the noun, but with a slightly modified order and nouns are sought. An important step in this part is to keep absolutely all the tags of the elements found going forward, in a structure called ACpre. It is used to verify the limits of the previous nominal expressions.

Let us assume that a nominal expression has two qualifying adjectives, and immediately after them there is a "new noun nucleus", that is, a different nominal expression. When the first nominal expression is built, both qualifying adjectives are added, but when the second nominal expression is built, AnaPro first verifies if the qualifying adjectives have been (or have not) previously been assigned. If that is the case, the search backwards for nominal elements is immediately interrupted. The same happens with specific punctuation marks.

An important addition has been the use of date tags as nuclei to construct nominal expressions with historical dates or references to some year, with all that this implies: use of prepositions, numbers, months, and years.

The search for lists and plural expressions is also added, looking for elements such as commas, nouns and conjunctions, in order to generate a single nominal expression, with the tag of the last noun modified to be plural. The search of plurals with conjunctions, unlike all the others, is not very reliable since errors have been detected when a conjunction is between two nouns. In this case its effectiveness depends on the correct use of the comma by the writer.

The ACpre structure. In the algorithm that constructs nominal expressions, the ACpre structure stores the names of all the nominal expressions at the same time they are being built, starting from the first word up to the last period. ACpre, a list, is used to avoid the construction of nominal expressions with "name nuclei" that have been previously used as part of a name of another nominal expression. Example: *el triunfo del movimiento revolucionario* {The success of the revolutionary movement}. A first step renders *triunfo* {success} as nucleus, the article *el* {the} is placed before it, and *del movimiento revolucionario* {of the revolutionary movement} is appended as qualifying adjectives. A second step will find *movimiento revolucionario* as nucleus and will try to create a new nominal expression. But, upon analyzing the ACpre structure, it will notice that *movimiento revolucionario* has already been used in a previous expression, therefore, it cannot be a new nucleus. Hence, it is ignored.

4.2 Stage 2

This stage detects the pronominal syntagmas containing pronouns as nucleus. At the same time, all the cases of nominal ellipsis are stored.

Identification of pronominal syntagmas. A simple analysis of the tag will identify the pronominals. Example:

PD0FS00013 = ésta {this} (See Table 3)

Here we find only a syntagma as direct anaphoric reference. Other examples of syntagmas are *el cual* {that which}*, la cual* {that which}*, el que* {that what}*,* which are also identified by AnaPro.

4.3 Stage 3

Candidates are identified, and direct anaphoric references (with gender and number defined) are resolved with a single candidate.
The module of AnaPro that does this task is the *Classifier.* It finds the coincidences of pronouns and nouns, classifying them as single or multiple matches (in the case of ties). Finally, it looks for the correct pairs.

Search for candidates. The tag found in Stage 2 is analyzed:

PD0FS00013 = ésta {this}

It is identified as a female singular pronoun. Thus, Stage 3 constructs the corresponding search pattern, which is then applied when searching for candidates:

Index: 13 Pattern: N\w{1}FS\w{3,10} Sentence: 2

indicating that it has to look for female singular nouns in the second or previous sentences. Therefore, the candidate found is (refer to Tables 2 and 3):

NCFS0001 = La batalla de puebla {Puebla's battle}
Therefore we have:

PD0FS00013 = NCFS0001

It means:

ésta {this} = La batalla de puebla {Puebla's battle}

In other words, the tags found in Stage 2 are converted by Stage 3 into regular expressions, which then look for the corresponding nominal expressions. Of course, more elaborate matching procedures are found in the literature, but this use of regular expressions is simpler and already produces good results (at least for Spanish texts).

4.4 Stage 4

It solves special cases of pronouns and candidates ties. This stage also uses the ***Classifier*.** Our example does not have special cases or ties among candidates. The following example has them:

Juan compró un boleto en la tienda. Éste estaba roto.

{John bought a ticket in the store. It was torn}

This is a tie, since there are several possible candidates: *Juan* {John}, *boleto* {ticket} and *tienda* {store}. *tienda* cannot be because *Éste* {this} is a masculine pronoun and *tienda* has female gender.

Solving the tie. To solve the tie, four important criteria are taken into account:

1. Less distance to the noun.
2. If it is determined.
3. Less distance to the main verb.
4. Higher number of occurrences in the text.

In case the first three criteria are not enough to break the tie, the numbers of occurrences are compared (fourth criteria). For this example, Table 4 shows the weights assigned to each candidate.

Criterion	Juan {John}	Boleto {ticket}
Less distance to the noun	0	1
Is it determined?	0	1
Less distance to the main verb	1	1
Higher number of occurrences in text	0	0
Total	1	3

Table 4. *Juan compró un boleto en la tienda. Éste estaba roto*. Comparison using weights among tied candidates

When comparing weights among candidates, we obtain *éste* {this} = *boleto* {ticket}, hence solving the sentence as follows:

*Juan compró un boleto en la tienda. [El boleto] estaba roto***.**

{John bought a ticket in the store. [The ticket] was torn}

Therefore, Stage 4 uses a rather simple method to solve ties. More complex methods could be used (for instance, the theory that considers the "center" of the discourse), but we are delighted to have found a rather simple and easy to implement procedure that, nevertheless, renders good results.

4.5 Stage 5

All the solved anaphors are replaced, yielding a new text that facilitates further processing by OM* (mentioned in Section 1). Pronouns are replaced by their respective candidates. The module that performs this task is the *Resolver***,** which replaces anaphora by their respective noun, generating an output file with the anaphors resolved. In general, this is a straightforward replacement module, with some cosmetic variations to its output in order to generate valid Spanish replacements.

Coming back to the previous example:

"La batalla de puebla. Aunque se considera en todo el país, ésta se celebra especialmente en Puebla, México; y en los estados del sur de Estados Unidos" {*Puebla's battle*. Although considered in all the country, **it** is celebrated especially in Puebla, Mexico, and in the southern states of USA}

As result of the *Resolver***,** the output file would contain:

"La batalla de puebla. Aunque se considera en todo el país, [la batalla] se celebra especialmente en Puebla, México; y en los estados del sur de Estados Unidos"

{Puebla's battle. Although considered in all the country, [*the battle*] is celebrated especially in Puebla, Mexico, and in the Southern states of USA}

Although AnaPro resolves *ésta* {this} as referring to *la batalla de puebla* {Puebla's battle}*,* when replacing is done, only the nucleus of the nominal expression is replaced. The expression *la batalla de puebla* has as nucleus *la batalla* {the battle}, which is used in the replacing algorithm.

5. Using AnaPro

AnaPro has two versions: a console and a graphical version. Before using any of them, a file in .txt format is needed in order to generate the tagged file Taller FT, which returns an extension *.txt.tag*. The Java virtual machine, version 1.2 or higher, must be already installed. We use the graphical version to show AnaPro in action.
For instance, in the document about the piano (Internet reference 1):

El piano está compuesto por una caja de resonancia, a la que se ha agregado un teclado, por donde se percuten las cuerdas de acero con macillos forrados de fieltro, por éstos se clasifica como instrumentos de percusión. Las primeras composiciones específicas para este instrumento surgieron alrededor del año 1732, entre ellas destacan las 12 sonatas para piano de Lodovico Giustini. {The piano is composed of a sounding board, to which a keyboard has been added, through which the steel chords are struck with hammers lined with felt; for this reason, it is classified as a percussion instrument. The first specific compositions for this instrument arose circa 1732. These include the 12 piano sonatas of Lodovico Giustini}

With the help of the example, the functions of the graphic interface (Figure 4) are explained

1. File menu, to open files with extension .txt.tag. They should be located in the same folders than the original text.

2. Tag showing original text, as found in the file.

3. Tag showing anaphoric references found by AnaPro, the word indices inside the text, and the noun corresponding to each reference; in other words, its solution.

4. Tagged text, used as a reference to compare it with the final tagged text.

5. Auxiliary view of tagged text, to compare it with the anaphoric references.

a. It shows the nominal expressions generated by AnaPro. Noun + adjectives, article.

6. It shows the new tagged text with the anaphors solved.

7. It shows the text with the words already replaced.

8. Erase all windows.

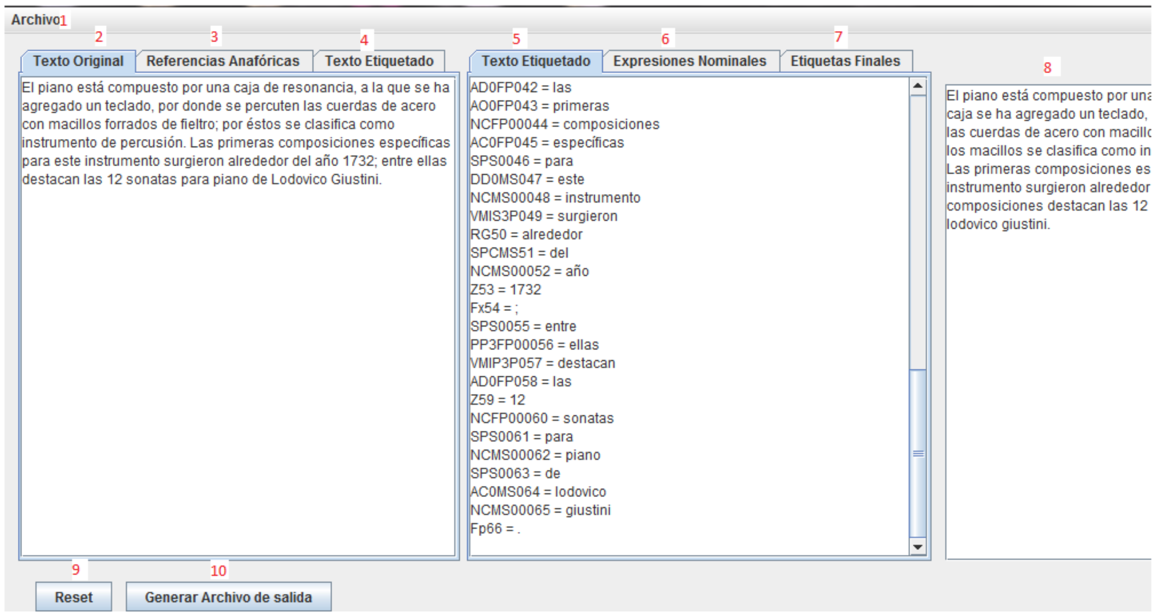
9. Button 10 stores the final file into the same folderscontaining the initial text file.

Figure 5 shows the tag showing the anaphoric references. The three sections marked A, B and C are:

Section A contains the anaphoric references found in the text, indicating the tag and then the word(s).

Section B shows the index of the references, followed by the search pattern (if it is a masculine pronoun, feminine, singular, etc.), and then it shows the sentence where the anaphora was found.

Section C shows the solution to each anaphora found by the resolver. When the *Generar Archivo* {generate file} button is clicked, a new file is produced, adding at the end of its name *_ANA.* Thus, if its name is *Mezcal.txt*, the file generated will be *Mezcal_ANA.txt.*

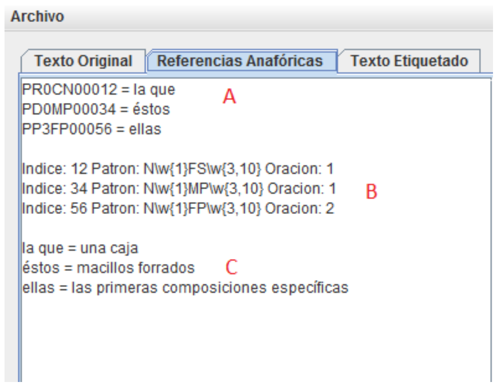


El piano está compuesto por una caja de resonancia, a la que se ha agregado un teclado, por donde se percuten las cuerdas de acero con macillos forrados de fieltro; por éstos se clasifica como instrumento de percusión. Las primeras composiciones específicas para este instrumento surgieron alrededor del año 1732; entre ellas destacan las 12 sonatas para piano de Lodovico Giustini.

```
AD0FP042 = las
AO0FP043 = primeras
NCFP00044 = composiciones
AC0FP045 = específicas
SPS0046 = para
DD0MS047 = este
NCMS00048 = instrumento
VMIS3P049 = surgieron
RG50 = alrededor
SPCMS51 = del
NCMS00052 = año
Z53 = 1732
Fx54 = ;
SPS0055 = entre
PP3FP00056 = ellas
VMIP3P057 = destacan
AD0FP058 = las
Z59 = 12
NCFP00060 = sonatas
SPS0061 = para
NCMS00062 = piano
SPS0063 = de
AC0MS064 = lodovico
NCMS00065 = giustini
Fp66 = .
```

El piano está compuesto por una
caja se ha agregado un teclado,
las cuerdas de acero con macillo
los macillos se clasifica como in
Las primeras composiciones es
instrumento surgieron alrededor
composiciones destacan las 12
lodovico giustini.

Figure 4. Graphical interface of the resolver



PR0CN00012 = la que
PD0MP00034 = éstos
PP3FP00056 = ellas

Indice: 12 Patron: N\w{1}FS\w{3,10} Oracion: 1
Indice: 34 Patron: N\w{1}MP\w{3,10} Oracion: 1
Indice: 56 Patron: N\w{1}FP\w{3,10} Oracion: 2

la que = una caja
éstos = macillos forrados
ellas = las primeras composiciones específicas

Figure 5. Tag: Anaphoric references

6. Tests

Testing has been done using the following texts taken from the Web:

1. "El piano" {the piano} (Internet reference 1),
2. "El burro flautista" {the flutist donkey} (Internet reference 2),
3. "Batalla de Puebla" {Puebla's battle} (Internet reference 3),
4. "Pancho Villa" (Internet reference 4) and
5. "Mario Almada" (Internet reference 5),

In all of them, AnaPro resolved automatically all the pronominal anaphors found. The same was done manually, in order to verify the effectiveness of the program. The formula is:

Accuracy = Percentage of correct responses = (number of anaphors correctly solved) / (total number of anaphors in the document).

We prefer to use accuracy instead of precision (which is the percentage of correct responses out of the anaphors considered; instead, accuracy considers *all* anaphors present in the text) or recall (percentage of the anaphors considered out of the anaphors in the text; instead, accuracy measures success against *all* anaphors present in the text). In other words, accuracy is the precision obtained in the presence of perfect recall. Table 5 shows the results obtained.

Text	Number of words	Anaphors solved by hand	Anaphors solved by AnaPro	Accuracy (assuming a perfect tagger)
El piano	61	3	3	100
El burro flautista	97	1	1	100
Batalla de Puebla	513	7	7	71.4
Pancho Villa	634	11	7	63.6
Mario Almada	601	9	5	55.5

Table 5. Where anaphors are solved, both by AnaPro and by hand

Failures, in most cases, were caused by final tagging. Thus, tagging has been manually corrected (to assume a perfect tagger). Retesting yields the results in Table 6, where failures are due to phrasing errors, or to very elaborate cases or very specific situations in a particular discourse (use of complex cohesion techniques), because the reference becomes incoherent. An example of these cases is the following fragment from Mario Almada (Internet reference 5):

"…fácilmente es expuesto a rodajes y en la Ciudad de México empieza a trabajar en un centro nocturno llamado Cabaret Señorial que era propiedad de su padre. Pero antes de esto, cuando su hermano Fernando empezó…" {… easily is exposed to filming and in Mexico City [he] starts to work in a night club called Cabaret Señorial that belonged to his father. But before this, when his brother Ferdinand started…}

The word esto {this} does not refer to a noun or a candidate, but to a whole context previously cited; hence, it is difficult to produce a computer tool that can resolve it.

We would like to perform larger tests, against standardized Spanish texts, where the anaphors were already found by hand. But, to our knowledge, these standard texts do not exist for Spanish. Our plan is to build these "standard texts" (containing the resolution of anaphora done by competent Spanish speakers) and then use them to assess the accuracy of AnaPro.

Annexs A, B and C shows the results of AnaPro on the texts “The Piano”, “Pancho Villa” and “Mario Almada”, respectively. For lack of space the results of the documents “El burro Flautista” {The Flutist Donkey} [internet reference 2] and “Batalla de Puebla” {Puebla’s battle} [internet reference 3] are not shown.

Text	Number of words	Anaphors solved by hand	Anaphors solved by AnaPro	Accuracy (assuming a perfect tagger)
El piano	61	3	3	100
El burro flautista	97	1	1	100
Batalla de Puebla	513	7	7	100
Pancho Villa	634	11	9	81.8
Mario Almada	601	9	8	88.8

Table 6. The columns represent: texts analyzed, number of words in each, manual solution, anaphoras solved by AnaPro, and the accuracy of Anapro assuming a perfect tagger (no mistakes in tagging)

6.1 Analyzing the failures of AnaPro

AnaPro has a high resolution success, except in some cases. Another example is taken from the same text (internet reference 5).

“Ha aparecido en más de 200 películas, muchas de las cuales eran drama clásico como su primer film Madre Querida (1935). En ésta apareció de niño con su hermano” {He has appeared in more than 200 films, many of which were classic drama, as his first film Beloved Mother (1935). In this he came out as a boy with his brother.}

Here we have a phrasing error, because the text uses a foreign word (film) that in Spanish is of masculine gender (el filme), but the text refers to it as ésta {this, feminine}. AnaPro generated the nominal expression (a candidate) *“film Madre Querida”* and assigned it the masculine gender. That fools AnaPro, which looks further backwards for a feminine candidate.

AnaPro heavily relies in the results of the tagger, since the anaphoric references fail if the tagger fails. If it is manually corrected (or by using a better tagger), the resolution algorithm works quite well.

7. Conclusions and future work

A method has been developed, using regular expressions, to separate, enumerate, and identify pronominal syntagmas and candidates to resolve the anaphoric references.

An algorithm has been designed and tested, able to solve ties using only information present in the text, statistical analysis of occurrences, text distance, and specific wording criteria, in order to assign weight to multiple candidates.

The method and algorithm are part of AnaPro, which resolves direct anaphora given by pronouns that occur in Spanish texts. The texts do not refer to a particular topic. AnaPro is part of Project OM* (a follow over to the Ph. D. thesis of one of the authors, Cuevas, 2006), that tries to build the ontology of the knowledge contained in a text, in order to answer complex queries.

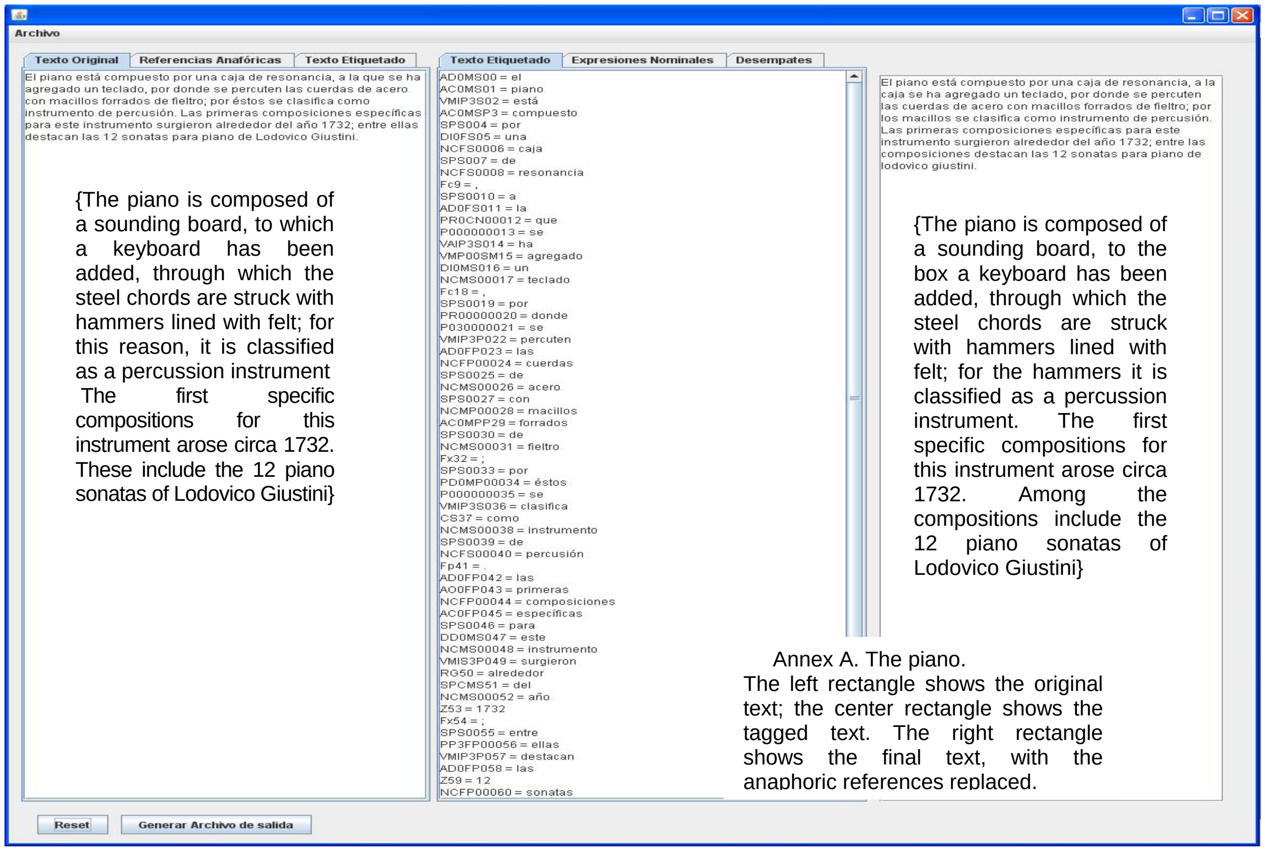
In order to minimize errors and to improve the efficiency of AnaPro, it is necessary to perfect the tagger, widening the scope of nouns handled, in order to solve quickly the simpler problems. Also, another tagger that uses EAGLE standard can be used.

In order to go through larger texts, we plan to build first standardized Spanish texts, where we can locate the anaphors and tag them by hand.

Acknowledgements

Authors I.T. and E.V. would like to acknowledge ESCOM-IPN, where they defended their thesis, #**20110083**, which gives a more detailed description of AnaPro. Work herein reported was partially sponsored by CONACYT Grant #128163 (Project OM*), by IPN (A.G. as Resident Scientist), and by SNI.

Annex A. Example. The Piano.



El piano está compuesto por una caja de resonancia, a la que se ha agregado un teclado, por donde se percuten las cuerdas de acero con macillos forrados de fieltro; por éstos se clasifica como instrumento de percusión. Las primeras composiciones específicas para este instrumento surgieron alrededor del año 1732; entre ellas destacan las 12 sonatas para piano de Lodovico Giustini.

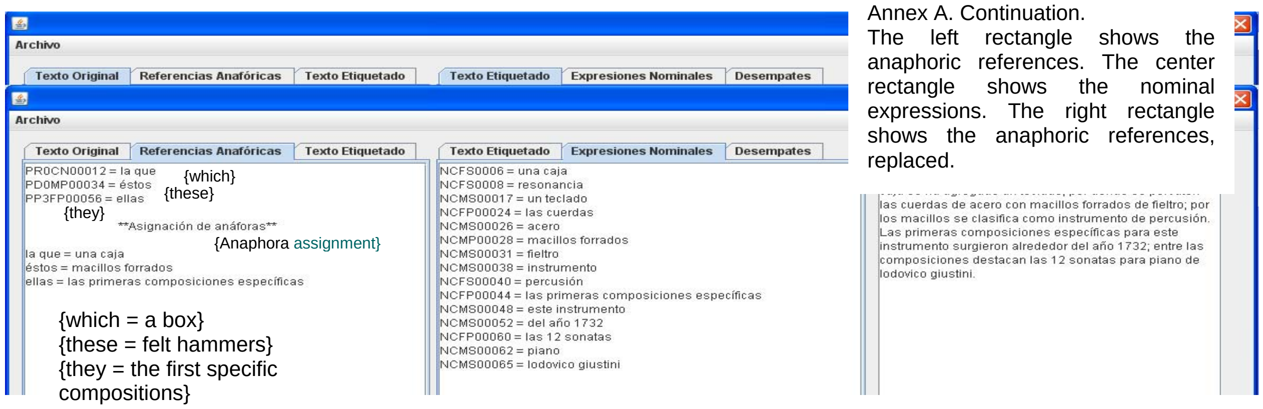
{The piano is composed of a sounding board, to which a keyboard has been added, through which the steel chords are struck with hammers lined with felt; for this reason, it is classified as a percussion instrument The first specific compositions for this instrument arose circa 1732. These include the 12 piano sonatas of Lodovico Giustini}

```
AD0MS00 = el
AC0MS01 = piano
VMIP3S02 = está
AC0MSP3 = compuesto
SPS004 = por
DI0FS05 = una
NCFS0006 = caja
SPS007 = de
NCFS0008 = resonancia
Fc9 = ,
SPS0010 = a
AD0FS011 = la
PR0CN00012 = que
P000000013 = se
VAIP3S014 = ha
VMP00SM15 = agregado
DI0MS016 = un
NCMS00017 = teclado
Fc18 = ,
SPS0019 = por
PR00000020 = donde
P030000021 = se
VMIP3P022 = percuten
AD0FP023 = las
NCFP00024 = cuerdas
SPS0025 = de
NCMS00026 = acero
SPS0027 = con
NCMP00028 = macillos
AC0MPP29 = forrados
SPS0030 = de
NCMS00031 = fieltro
Fx32 = ;
SPS0033 = por
PD0MP00034 = éstos
P000000035 = se
VMIP3S036 = clasifica
CS37 = como
NCMS00038 = instrumento
SPS0039 = de
NCFS00040 = percusión
Fp41 = .
AD0FP042 = las
AO0FP043 = primeras
NCFP00044 = composiciones
AC0FP045 = específicas
SPS0046 = para
DD0MS047 = este
NCMS00048 = instrumento
VMIS3P049 = surgieron
RG50 = alrededor
SPCMS51 = del
NCMS00052 = año
Z53 = 1732
Fx54 = ;
SPS0055 = entre
PP3FP00056 = ellas
VMIP3P057 = destacan
AD0FP058 = las
Z59 = 12
NCFP00060 = sonatas
```

El piano está compuesto por una caja de resonancia, a la caja se ha agregado un teclado, por donde se percuten las cuerdas de acero con macillos forrados de fieltro; por los macillos se clasifica como instrumento de percusión. Las primeras composiciones específicas para este instrumento surgieron alrededor del año 1732; entre las composiciones destacan las 12 sonatas para piano de lodovico giustini.

{The piano is composed of a sounding board, to the box a keyboard has been added, through which the steel chords are struck with hammers lined with felt; for the hammers it is classified as a percussion instrument. The first specific compositions for this instrument arose circa 1732. Among the compositions include the 12 piano sonatas of Lodovico Giustini}

Annex A. The piano.
The left rectangle shows the original text; the center rectangle shows the tagged text. The right rectangle shows the final text, with the anaphoric references replaced.



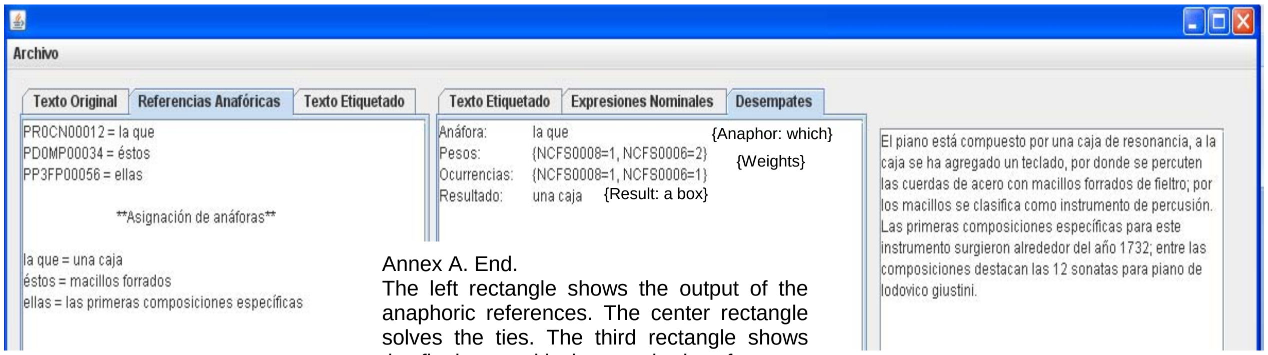
Annex A. Continuation.
The left rectangle shows the anaphoric references. The center rectangle shows the nominal expressions. The right rectangle shows the anaphoric references, replaced.

```
PR0CN00012 = la que      {which}
PD0MP00034 = éstos     {these}
PP3FP00056 = ellas
        {they}
              **Asignación de anáforas**
                              {Anaphora assignment}
la que = una caja
éstos = macillos forrados
ellas = las primeras composiciones específicas
```

{which = a box}
{these = felt hammers}
{they = the first specific compositions}

```
NCFS0006 = una caja
NCFS0008 = resonancia
NCMS00017 = un teclado
NCFP00024 = las cuerdas
NCMS00026 = acero
NCMP00028 = macillos forrados
NCMS00031 = fieltro
NCMS00038 = instrumento
NCFS00040 = percusión
NCFP00044 = las primeras composiciones específicas
NCMS00048 = este instrumento
NCMS00052 = del año 1732
NCFP00060 = las 12 sonatas
NCMS00062 = piano
NCMS00065 = lodovico giustini
```

las cuerdas de acero con macillos forrados de fieltro; por los macillos se clasifica como instrumento de percusión. Las primeras composiciones específicas para este instrumento surgieron alrededor del año 1732; entre las composiciones destacan las 12 sonatas para piano de lodovico giustini.



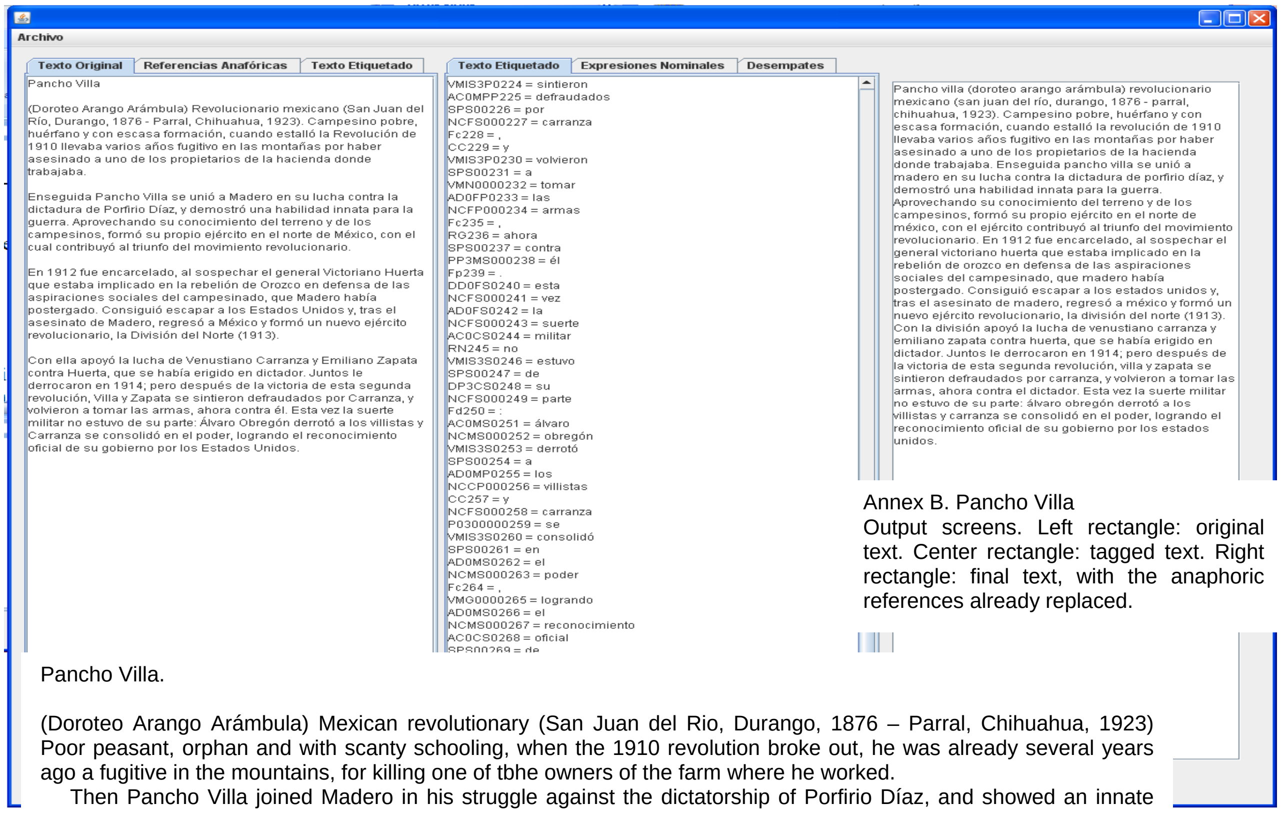
```
PR0CN00012 = la que
PD0MP00034 = éstos
PP3FP00056 = ellas

**Asignación de anáforas**

la que = una caja
éstos = macillos forrados
ellas = las primeras composiciones específicas
```

```
Anáfora:      la que                              {Anaphor: which}
Pesos:        {NCFS0008=1, NCFS0006=2}            {Weights}
Ocurrencias:  {NCFS0008=1, NCFS0006=1}
Resultado:    una caja         {Result: a box}
```

El piano está compuesto por una caja de resonancia, a la caja se ha agregado un teclado, por donde se percuten las cuerdas de acero con macillos forrados de fieltro; por los macillos se clasifica como instrumento de percusión. Las primeras composiciones específicas para este instrumento surgieron alrededor del año 1732; entre las composiciones destacan las 12 sonatas para piano de lodovico giustini.

Annex A. End.
The left rectangle shows the output of the anaphoric references. The center rectangle solves the ties. The third rectangle shows the final text, with the anaphoric references already replaced..

Annex B. Example. Pancho Villa.



Pancho Villa

(Doroteo Arango Arámbula) Revolucionario mexicano (San Juan del Río, Durango, 1876 - Parral, Chihuahua, 1923). Campesino pobre, huérfano y con escasa formación, cuando estalló la Revolución de 1910 llevaba varios años fugitivo en las montañas por haber asesinado a uno de los propietarios de la hacienda donde trabajaba.

Enseguida Pancho Villa se unió a Madero en su lucha contra la dictadura de Porfirio Díaz, y demostró una habilidad innata para la guerra. Aprovechando su conocimiento del terreno y de los campesinos, formó su propio ejército en el norte de México, con el cual contribuyó al triunfo del movimiento revolucionario.

En 1912 fue encarcelado, al sospechar el general Victoriano Huerta que estaba implicado en la rebelión de Orozco en defensa de las aspiraciones sociales del campesinado, que Madero había postergado. Consiguió escapar a los Estados Unidos y, tras el asesinato de Madero, regresó a México y formó un nuevo ejército revolucionario, la División del Norte (1913).

Con ella apoyó la lucha de Venustiano Carranza y Emiliano Zapata contra Huerta, que se había erigido en dictador. Juntos le derrocaron en 1914; pero después de la victoria de esta segunda revolución, Villa y Zapata se sintieron defraudados por Carranza, y volvieron a tomar las armas, ahora contra él. Esta vez la suerte militar no estuvo de su parte: Álvaro Obregón derrotó a los villistas y Carranza se consolidó en el poder, logrando el reconocimiento oficial de su gobierno por los Estados Unidos.

```
VMIS3P0224 = sintieron
AC0MPP225 = defraudados
SPS00226 = por
NCFS000227 = carranza
Fc228 = ,
CC229 = y
VMIS3P0230 = volvieron
SPS00231 = a
VMN0000232 = tomar
AD0FP0233 = las
NCFP000234 = armas
Fc235 = ,
RG236 = ahora
SPS00237 = contra
PP3MS000238 = él
Fp239 = .
DD0FS0240 = esta
NCFS000241 = vez
AD0FS0242 = la
NCFS000243 = suerte
AC0CS0244 = militar
RN245 = no
VMIS3S0246 = estuvo
SPS00247 = de
DP3CS0248 = su
NCFS000249 = parte
Fd250 = :
AC0MS0251 = álvaro
NCMS000252 = obregón
VMIS3S0253 = derrotó
SPS00254 = a
AD0MP0255 = los
NCCP000256 = villistas
CC257 = y
NCFS000258 = carranza
P0300000259 = se
VMIS3S0260 = consolidó
SPS00261 = en
AD0MS0262 = el
NCMS000263 = poder
Fc264 = ,
VMG0000265 = logrando
AD0MS0266 = el
NCMS000267 = reconocimiento
AC0CS0268 = oficial
SPS00269 = de
```

Pancho villa (doroteo arango arámbula) revolucionario mexicano (san juan del río, durango, 1876 - parral, chihuahua, 1923). Campesino pobre, huérfano y con escasa formación, cuando estalló la revolución de 1910 llevaba varios años fugitivo en las montañas por haber asesinado a uno de los propietarios de la hacienda donde trabajaba. Enseguida pancho villa se unió a madero en su lucha contra la dictadura de porfirio díaz, y demostró una habilidad innata para la guerra. Aprovechando su conocimiento del terreno y de los campesinos, formó su propio ejército en el norte de méxico, con el ejército contribuyó al triunfo del movimiento revolucionario. En 1912 fue encarcelado, al sospechar el general victoriano huerta que estaba implicado en la rebelión de orozco en defensa de las aspiraciones sociales del campesinado, que madero había postergado. Consiguió escapar a los estados unidos y, tras el asesinato de madero, regresó a méxico y formó un nuevo ejército revolucionario, la división del norte (1913). Con la división apoyó la lucha de venustiano carranza y emiliano zapata contra huerta, que se había erigido en dictador. Juntos le derrocaron en 1914; pero después de la victoria de esta segunda revolución, villa y zapata se sintieron defraudados por carranza, y volvieron a tomar las armas, ahora contra el dictador. Esta vez la suerte militar no estuvo de su parte: álvaro obregón derrotó a los villistas y carranza se consolidó en el poder, logrando el reconocimiento oficial de su gobierno por los estados unidos.

Annex B. Pancho Villa
Output screens. Left rectangle: original text. Center rectangle: tagged text. Right rectangle: final text, with the anaphoric references already replaced.

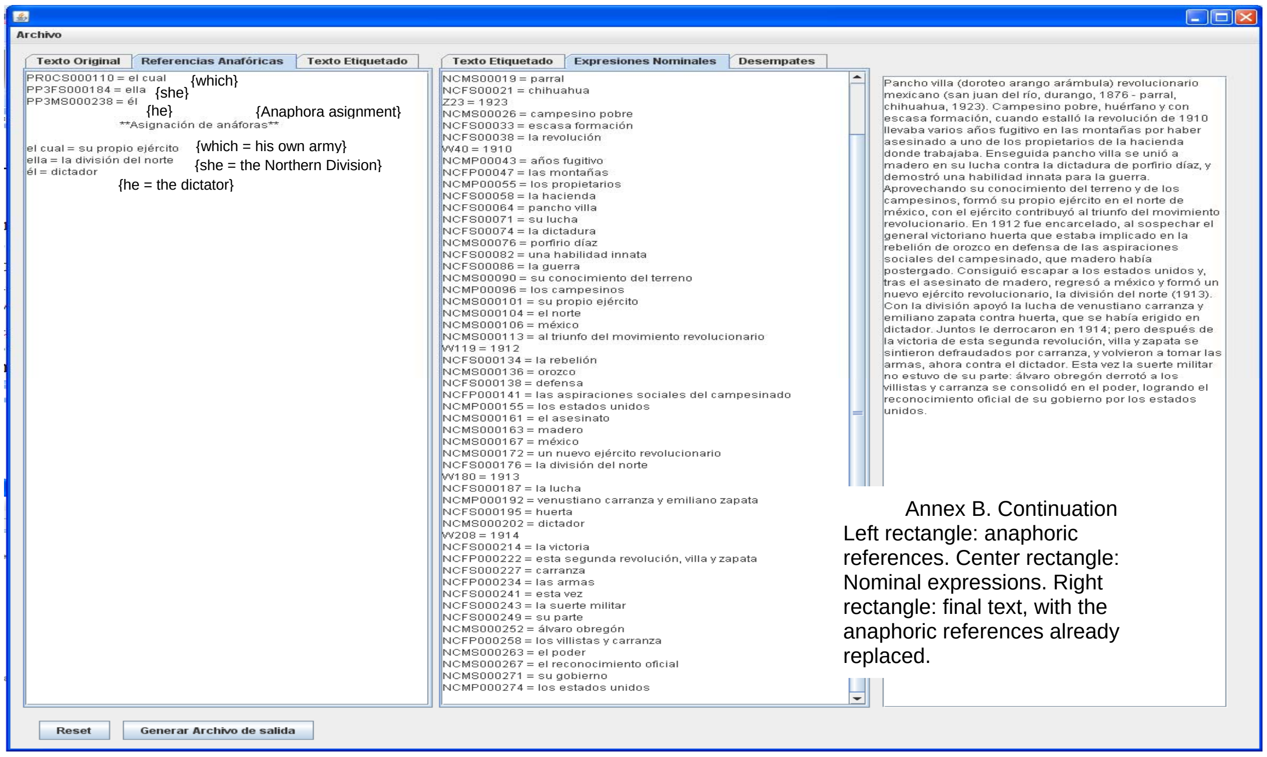
Pancho Villa.

(Doroteo Arango Arámbula) Mexican revolutionary (San Juan del Rio, Durango, 1876 – Parral, Chihuahua, 1923) Poor peasant, orphan and with scanty schooling, when the 1910 revolution broke out, he was already several years ago a fugitive in the mountains, for killing one of tbhe owners of the farm where he worked.

Then Pancho Villa joined Madero in his struggle against the dictatorship of Porfirio Díaz, and showed an innate ability for war. Leveraging his knowledge of the land and the peasants, he formed his own army in Northern Mexico, which contributed to the success of the revolutionary movement.

In 1912 he was jailed, since General Victoriano Huerta suspected he was involved in the revolution of Orozco in defense of the social aspirations of the peasantry, which Madero had postponed. He escaped to the United States and, after the assassination of Madero, returned to Mexico and formed a new revolutionary army, the Northern Division (1913).

With it he supported the struggle of Emiliano Zapata and Venustiano Carranza against Huerta, who had emerged as dictator. Together they overthrew him in 1914, but after the victory of the second revolution, Villa and Zapata felt disappointed by Carranza, and took up arms again, now against him. This time military luck was not on his side: Álvaro Obregón defeated Villa's army and Carranza consolidated his power, and obtained the official recognition by the U.S. government.


Archivo

Texto Original | Referencias Anafóricas | Texto Etiquetado

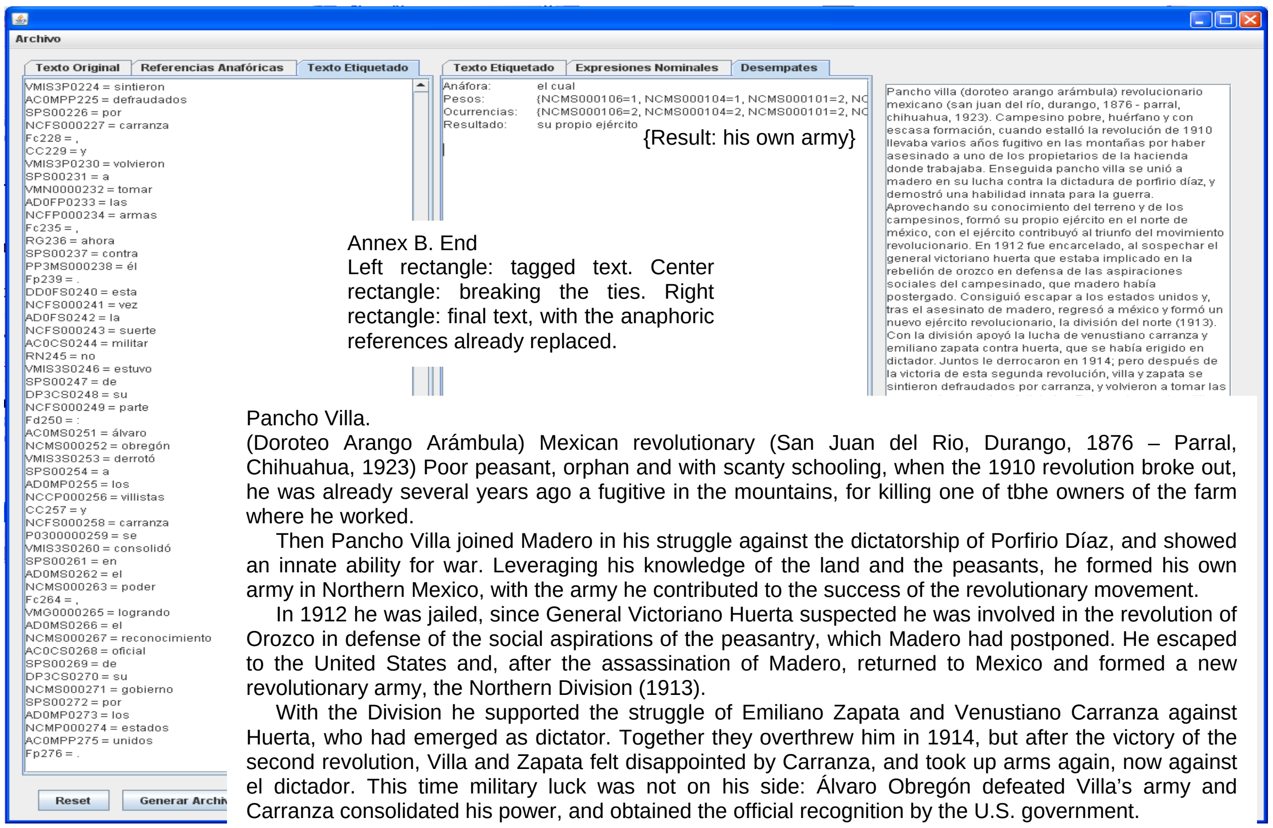
```
PR0CS000110 = el cual    {which}
PP3FS000184 = ella   {she}
PP3MS000238 = él
      {he}                    {Anaphora asignment}
         **Asignación de anáforas**

el cual = su propio ejército    {which = his own army}
ella = la división del norte    {she = the Northern Division}
él = dictador
         {he = the dictator}
```

Texto Etiquetado | Expresiones Nominales | Desempates

```
NCMS00019 = parral
NCFS00021 = chihuahua
Z23 = 1923
NCMS00026 = campesino pobre
NCFS00033 = escasa formación
NCFS00038 = la revolución
W40 = 1910
NCMP00043 = años fugitivo
NCFP00047 = las montañas
NCMP00055 = los propietarios
NCFS00058 = la hacienda
NCFS00064 = pancho villa
NCFS00071 = su lucha
NCFS00074 = la dictadura
NCMS00076 = porfirio díaz
NCFS00082 = una habilidad innata
NCFS00086 = la guerra
NCMS00090 = su conocimiento del terreno
NCMP00096 = los campesinos
NCMS000101 = su propio ejército
NCMS000104 = el norte
NCMS000106 = méxico
NCMS000113 = al triunfo del movimiento revolucionario
W119 = 1912
NCFS000134 = la rebelión
NCMS000136 = orozco
NCFS000138 = defensa
NCFP000141 = las aspiraciones sociales del campesinado
NCMP000155 = los estados unidos
NCMS000161 = el asesinato
NCMS000163 = madero
NCMS000167 = méxico
NCMS000172 = un nuevo ejército revolucionario
NCFS000176 = la división del norte
W180 = 1913
NCFS000187 = la lucha
NCMP000192 = venustiano carranza y emiliano zapata
NCFS000195 = huerta
NCMS000202 = dictador
W208 = 1914
NCFS000214 = la victoria
NCFP000222 = esta segunda revolución, villa y zapata
NCFS000227 = carranza
NCFP000234 = las armas
NCFS000241 = esta vez
NCFS000243 = la suerte militar
NCFS000249 = su parte
NCMS000252 = álvaro obregón
NCFP000258 = los villistas y carranza
NCMS000263 = el poder
NCMS000267 = el reconocimiento oficial
NCMS000271 = su gobierno
NCMP000274 = los estados unidos
```

Pancho villa (doroteo arango arámbula) revolucionario mexicano (san juan del río, durango, 1876 - parral, chihuahua, 1923). Campesino pobre, huérfano y con escasa formación, cuando estalló la revolución de 1910 llevaba varios años fugitivo en las montañas por haber asesinado a uno de los propietarios de la hacienda donde trabajaba. Enseguida pancho villa se unió a madero en su lucha contra la dictadura de porfirio díaz, y demostró una habilidad innata para la guerra. Aprovechando su conocimiento del terreno y de los campesinos, formó su propio ejército en el norte de méxico, con el ejército contribuyó al triunfo del movimiento revolucionario. En 1912 fue encarcelado, al sospechar el general victoriano huerta que estaba implicado en la rebelión de orozco en defensa de las aspiraciones sociales del campesinado, que madero había postergado. Consiguió escapar a los estados unidos y, tras el asesinato de madero, regresó a méxico y formó un nuevo ejército revolucionario, la división del norte (1913). Con la división apoyó la lucha de venustiano carranza y emiliano zapata contra huerta, que se había erigido en dictador. Juntos le derrocaron en 1914; pero después de la victoria de esta segunda revolución, villa y zapata se sintieron defraudados por carranza, y volvieron a tomar las armas, ahora contra el dictador. Esta vez la suerte militar no estuvo de su parte: álvaro obregón derrotó a los villistas y carranza se consolidó en el poder, logrando el reconocimiento oficial de su gobierno por los estados unidos.

Annex B. Continuation
Left rectangle: anaphoric references. Center rectangle: Nominal expressions. Right rectangle: final text, with the anaphoric references already replaced.


Archivo

Texto Original | Referencias Anafóricas | Texto Etiquetado

```
VMIS3P0224 = sintieron
AC0MPP225 = defraudados
SPS00226 = por
NCFS000227 = carranza
Fc228 = ,
CC229 = y
VMIS3P0230 = volvieron
SPS00231 = a
VMN0000232 = tomar
AD0FP0233 = las
NCFP000234 = armas
Fc235 = ,
RG236 = ahora
SPS00237 = contra
PP3MS000238 = él
Fp239 = .
DD0FS0240 = esta
NCFS000241 = vez
AD0FS0242 = la
NCFS000243 = suerte
AC0CS0244 = militar
RN245 = no
VMIS3S0246 = estuvo
SPS00247 = de
DP3CS0248 = su
NCFS000249 = parte
Fd250 = :
AC0MS0251 = álvaro
NCMS000252 = obregón
VMIS3S0253 = derrotó
SPS00254 = a
AD0MP0255 = los
NCCP000256 = villistas
CC257 = y
NCFS000258 = carranza
P0300000259 = se
VMIS3S0260 = consolidó
SPS00261 = en
AD0MS0262 = el
NCMS000263 = poder
Fc264 = ,
VMG0000265 = logrando
AD0MS0266 = el
NCMS000267 = reconocimiento
AC0CS0268 = oficial
SPS00269 = de
DP3CS0270 = su
NCMS000271 = gobierno
SPS00272 = por
AD0MP0273 = los
NCMP000274 = estados
AC0MPP275 = unidos
Fp276 = .
```

Texto Etiquetado | Expresiones Nominales | Desempates

```
Anáfora:        el cual
Pesos:          {NCMS000106=1, NCMS000104=1, NCMS000101=2, NC
Ocurrencias:    {NCMS000106=2, NCMS000104=2, NCMS000101=2, NC
Resultado:      su propio ejército
                                  {Result: his own army}
```

Pancho villa (doroteo arango arámbula) revolucionario mexicano (san juan del río, durango, 1876 - parral, chihuahua, 1923). Campesino pobre, huérfano y con escasa formación, cuando estalló la revolución de 1910 llevaba varios años fugitivo en las montañas por haber asesinado a uno de los propietarios de la hacienda donde trabajaba. Enseguida pancho villa se unió a madero en su lucha contra la dictadura de porfirio díaz, y demostró una habilidad innata para la guerra. Aprovechando su conocimiento del terreno y de los campesinos, formó su propio ejército en el norte de méxico, con el ejército contribuyó al triunfo del movimiento revolucionario. En 1912 fue encarcelado, al sospechar el general victoriano huerta que estaba implicado en la rebelión de orozco en defensa de las aspiraciones sociales del campesinado, que madero había postergado. Consiguió escapar a los estados unidos y, tras el asesinato de madero, regresó a méxico y formó un nuevo ejército revolucionario, la división del norte (1913). Con la división apoyó la lucha de venustiano carranza y emiliano zapata contra huerta, que se había erigido en dictador. Juntos le derrocaron en 1914; pero después de la victoria de esta segunda revolución, villa y zapata se sintieron defraudados por carranza, y volvieron a tomar las

Annex B. End
Left rectangle: tagged text. Center rectangle: breaking the ties. Right rectangle: final text, with the anaphoric references already replaced.

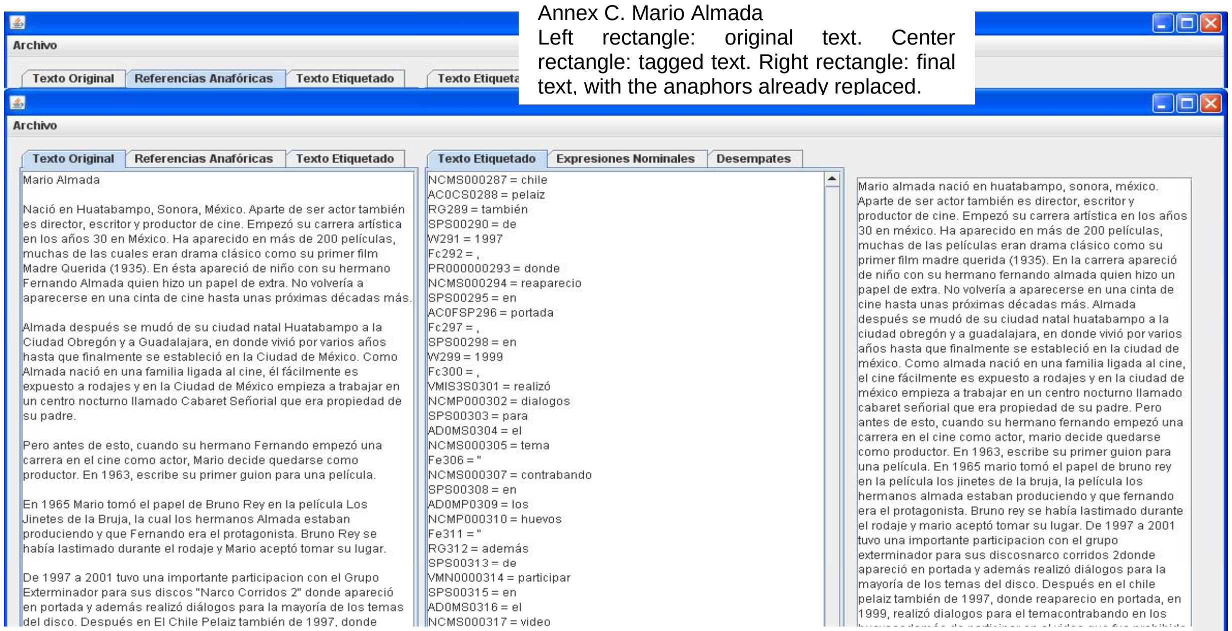
Pancho Villa.
(Doroteo Arango Arámbula) Mexican revolutionary (San Juan del Rio, Durango, 1876 – Parral, Chihuahua, 1923) Poor peasant, orphan and with scanty schooling, when the 1910 revolution broke out, he was already several years ago a fugitive in the mountains, for killing one of tbhe owners of the farm where he worked.

Then Pancho Villa joined Madero in his struggle against the dictatorship of Porfirio Díaz, and showed an innate ability for war. Leveraging his knowledge of the land and the peasants, he formed his own army in Northern Mexico, with the army he contributed to the success of the revolutionary movement.

In 1912 he was jailed, since General Victoriano Huerta suspected he was involved in the revolution of Orozco in defense of the social aspirations of the peasantry, which Madero had postponed. He escaped to the United States and, after the assassination of Madero, returned to Mexico and formed a new revolutionary army, the Northern Division (1913).

With the Division he supported the struggle of Emiliano Zapata and Venustiano Carranza against Huerta, who had emerged as dictator. Together they overthrew him in 1914, but after the victory of the second revolution, Villa and Zapata felt disappointed by Carranza, and took up arms again, now against el dictador. This time military luck was not on his side: Álvaro Obregón defeated Villa's army and Carranza consolidated his power, and obtained the official recognition by the U.S. government.

Annex C. Example. Mario Almada.



Annex C. Mario Almada
Left rectangle: original text. Center rectangle: tagged text. Right rectangle: final text, with the anaphors already replaced.

Mario Almada

Nació en Huatabampo, Sonora, México. Aparte de ser actor también es director, escritor y productor de cine. Empezó su carrera artística en los años 30 en México. Ha aparecido en más de 200 películas, muchas de las cuales eran drama clásico como su primer film Madre Querida (1935). En ésta apareció de niño con su hermano Fernando Almada quien hizo un papel de extra. No volvería a aparecerse en una cinta de cine hasta unas próximas décadas más.

Almada después se mudó de su ciudad natal Huatabampo a la Ciudad Obregón y a Guadalajara, en donde vivió por varios años hasta que finalmente se estableció en la Ciudad de México. Como Almada nació en una familia ligada al cine, él fácilmente es expuesto a rodajes y en la Ciudad de México empieza a trabajar en un centro nocturno llamado Cabaret Señorial que era propiedad de su padre.

Pero antes de esto, cuando su hermano Fernando empezó una carrera en el cine como actor, Mario decide quedarse como productor. En 1963, escribe su primer guion para una película.

En 1965 Mario tomó el papel de Bruno Rey en la película Los Jinetes de la Bruja, la cual los hermanos Almada estaban produciendo y que Fernando era el protagonista. Bruno Rey se había lastimado durante el rodaje y Mario aceptó tomar su lugar.

De 1997 a 2001 tuvo una importante participacion con el Grupo Exterminador para sus discos "Narco Corridos 2" donde apareció en portada y además realizó diálogos para la mayoría de los temas del disco. Después en El Chile Pelaiz también de 1997, donde

```
NCMS000287 = chile
AC0CS0288 = pelaiz
RG289 = también
SPS00290 = de
W291 = 1997
Fc292 = ,
PR000000293 = donde
NCMS000294 = reaparecio
SPS00295 = en
AC0FSP296 = portada
Fc297 = ,
SPS00298 = en
W299 = 1999
Fc300 = ,
VMIS3S0301 = realizó
NCMP000302 = dialogos
SPS00303 = para
AD0MS0304 = el
NCMS000305 = tema
Fe306 = "
NCMS000307 = contrabando
SPS00308 = en
AD0MP0309 = los
NCMP000310 = huevos
Fe311 = "
RG312 = además
SPS00313 = de
VMN0000314 = participar
SPS00315 = en
AD0MS0316 = el
NCMS000317 = video
```

Mario almada nació en huatabampo, sonora, méxico. Aparte de ser actor también es director, escritor y productor de cine. Empezó su carrera artística en los años 30 en méxico. Ha aparecido en más de 200 películas, muchas de las películas eran drama clásico como su primer film madre querida (1935). En la carrera apareció de niño con su hermano fernando almada quien hizo un papel de extra. No volvería a aparecerse en una cinta de cine hasta unas próximas décadas más. Almada después se mudó de su ciudad natal huatabampo a la ciudad obregón y a guadalajara, en donde vivió por varios años hasta que finalmente se estableció en la ciudad de méxico. Como almada nació en una familia ligada al cine, el cine fácilmente es expuesto a rodajes y en la ciudad de méxico empieza a trabajar en un centro nocturno llamado cabaret señorial que era propiedad de su padre. Pero antes de esto, cuando su hermano fernando empezó una carrera en el cine como actor, mario decide quedarse como productor. En 1963, escribe su primer guion para una película. En 1965 mario tomó el papel de bruno rey en la película los jinetes de la bruja, la película los hermanos almada estaban produciendo y que fernando era el protagonista. Bruno rey se había lastimado durante el rodaje y mario aceptó tomar su lugar. De 1997 a 2001 tuvo una importante participacion con el grupo exterminador para sus discosnarco corridos 2donde apareció en portada y además realizó diálogos para la mayoría de los temas del disco. Después en el chile pelaiz también de 1997, donde reaparecio en portada, en 1999, realizó dialogos para el temacontrabando en los

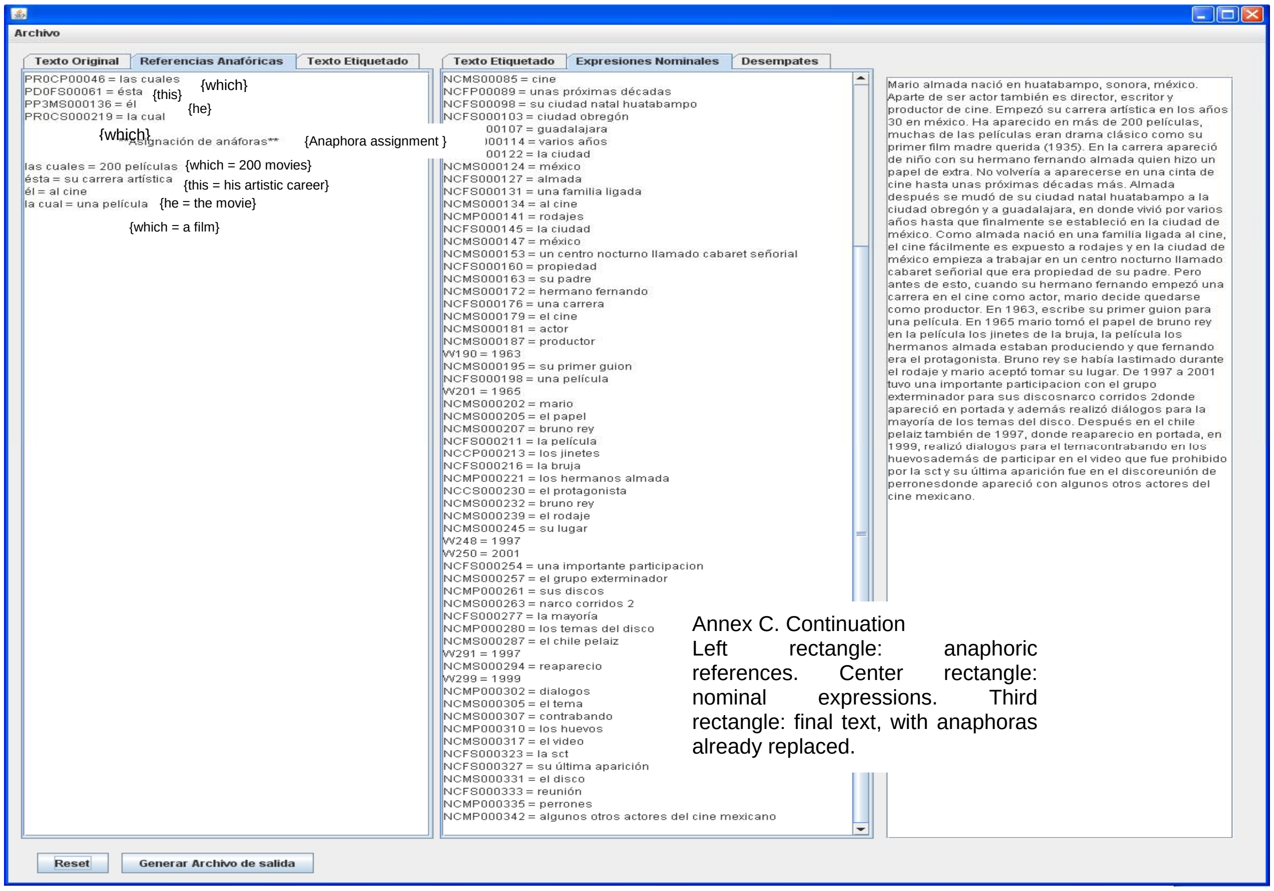
Mario Almada.- Born in Huatabampo, Sonora, Mexico. In addition of being a great actor, he is also director, writer and movie producer. He started his artistic career in the 1930s in Mexico. He has appeared in more than 200 movies, many of which were classic drama, like his first film *Madre Querida* (1935). In this he appeared as a child with his brother Fernando Almada, who performed as an extra. He would not appear again in a movie until several decades later.

Almada moved from his native Huatabampo to Ciudad Obregón and Guadalajara, where he lived for several years until he finally settled in Mexico City. Since Almada was born in a family linked to films, he is easily exposed to filming and in Mexico City he starts working in a night club called *Cabaret Señorial* which belonged to his father.

But before this, when his brother Ferdinand started a career in the movies as an actor, Mario decides to stay as producer. In 1963, he writes his first screenplay for a movie.

In 1965 Mario performed the role of Bruno the King in the movie Riders of the Witch, which the Almada brothers were producing, and where Fernando was the first actor. Bruno the King became injured during the filming and Mario accepted to take his place.

From 1997 to 2001 he had an important participation with the group The Exterminator for their records “Narco Corridos 2” where he appeared in the cover, and he also conducted dialogues for most tracks in the album. After that, in *El Chile Peláiz* also from 1997, where he reappeared in the cover, in 1999 he performed dialogues for the theme “Smuggling in the eggs” in addition to taking part in the video that was prohibited by SCT, and his last appearance was in the album “Gathering of dogs” where he appeared with some other Mexican actors.

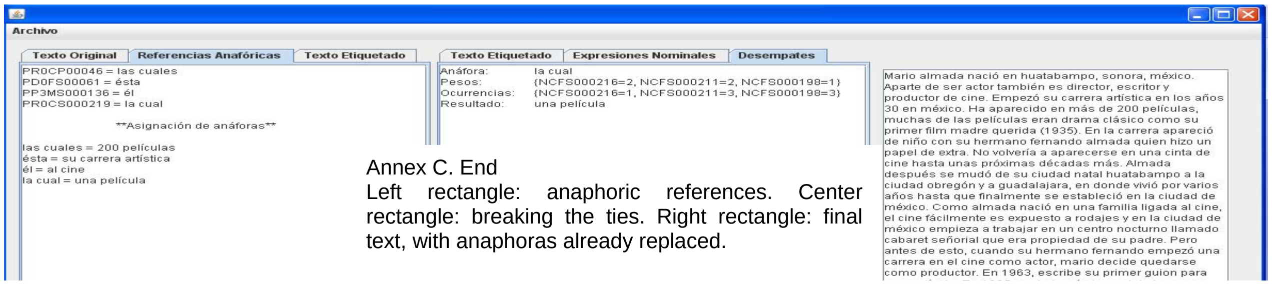


```
PR0CP00046 = las cuales
PD0FS00061 = ésta
PP3MS000136 = él
PR0CS000219 = la cual

                **Asignación de anáforas**

las cuales = 200 películas
ésta = su carrera artística
él = al cine
la cual = una película
```

```
NCMS00085 = cine
NCFP00089 = unas próximas décadas
NCFS00098 = su ciudad natal huatabampo
NCFS000103 = ciudad obregón
00107 = guadalajara
00114 = varios años
00122 = la ciudad
NCMS000124 = méxico
NCFS000127 = almada
NCFS000131 = una familia ligada
NCMS000134 = al cine
NCMP000141 = rodajes
NCFS000145 = la ciudad
NCMS000147 = méxico
NCMS000153 = un centro nocturno llamado cabaret señorial
NCFS000160 = propiedad
NCMS000163 = su padre
NCMS000172 = hermano fernando
NCFS000176 = una carrera
NCMS000179 = el cine
NCMS000181 = actor
NCMS000187 = productor
W190 = 1963
NCMS000195 = su primer guion
NCFS000198 = una película
W201 = 1965
NCMS000202 = mario
NCMS000205 = el papel
NCMS000207 = bruno rey
NCFS000211 = la película
NCCP000213 = los jinetes
NCFS000216 = la bruja
NCMP000221 = los hermanos almada
NCCS000230 = el protagonista
NCMS000232 = bruno rey
NCMS000239 = el rodaje
NCMS000245 = su lugar
W248 = 1997
W250 = 2001
NCFS000254 = una importante participacion
NCMS000257 = el grupo exterminador
NCMP000261 = sus discos
NCMS000263 = narco corridos 2
NCFS000277 = la mayoría
NCMP000280 = los temas del disco
NCMS000287 = el chile pelaiz
W291 = 1997
NCMS000294 = reaparecio
W299 = 1999
NCMP000302 = dialogos
NCMS000305 = el tema
NCMS000307 = contrabando
NCMP000310 = los huevos
NCMS000317 = el video
NCFS000323 = la sct
NCFS000327 = su última aparición
NCMS000331 = el disco
NCFS000333 = reunión
NCMP000335 = perrones
NCMP000342 = algunos otros actores del cine mexicano
```

Mario almada nació en huatabampo, sonora, méxico. Aparte de ser actor también es director, escritor y productor de cine. Empezó su carrera artística en los años 30 en méxico. Ha aparecido en más de 200 películas, muchas de las películas eran drama clásico como su primer film madre querida (1935). En la carrera apareció de niño con su hermano fernando almada quien hizo un papel de extra. No volvería a aparecerse en una cinta de cine hasta unas próximas décadas más. Almada después se mudó de su ciudad natal huatabampo a la ciudad obregón y a guadalajara, en donde vivió por varios años hasta que finalmente se estableció en la ciudad de méxico. Como almada nació en una familia ligada al cine, el cine fácilmente es expuesto a rodajes y en la ciudad de méxico empieza a trabajar en un centro nocturno llamado cabaret señorial que era propiedad de su padre. Pero antes de esto, cuando su hermano fernando empezó una carrera en el cine como actor, mario decide quedarse como productor. En 1963, escribe su primer guion para una película. En 1965 mario tomó el papel de bruno rey en la película los jinetes de la bruja, la película los hermanos almada estaban produciendo y que fernando era el protagonista. Bruno rey se había lastimado durante el rodaje y mario aceptó tomar su lugar. De 1997 a 2001 tuvo una importante participacion con el grupo exterminador para sus discosnarco corridos 2donde apareció en portada y además realizó diálogos para la mayoría de los temas del disco. Después en el chile pelaiz también de 1997, donde reaparecio en portada, en 1999, realizó dialogos para el temacontrabando en los huevosademás de participar en el video que fue prohibido por la sct y su última aparición fue en el discoreunión de perronesdonde apareció con algunos otros actores del cine mexicano.

Annex C. Continuation
Left rectangle: anaphoric references. Center rectangle: nominal expressions. Third rectangle: final text, with anaphoras already replaced.



```
PR0CP00046 = las cuales
PD0FS00061 = ésta
PP3MS000136 = él
PR0CS000219 = la cual

                **Asignación de anáforas**

las cuales = 200 películas
ésta = su carrera artística
él = al cine
la cual = una película
```

```
Anáfora:        la cual
Pesos:          {NCFS000216=2, NCFS000211=2, NCFS000198=1}
Ocurrencias:    {NCFS000216=1, NCFS000211=3, NCFS000198=3}
Resultado:      una película
```

Mario almada nació en huatabampo, sonora, méxico. Aparte de ser actor también es director, escritor y productor de cine. Empezó su carrera artística en los años 30 en méxico. Ha aparecido en más de 200 películas, muchas de las películas eran drama clásico como su primer film madre querida (1935). En la carrera apareció de niño con su hermano fernando almada quien hizo un papel de extra. No volvería a aparecerse en una cinta de cine hasta unas próximas décadas más. Almada después se mudó de su ciudad natal huatabampo a la ciudad obregón y a guadalajara, en donde vivió por varios años hasta que finalmente se estableció en la ciudad de méxico. Como almada nació en una familia ligada al cine, el cine fácilmente es expuesto a rodajes y en la ciudad de méxico empieza a trabajar en un centro nocturno llamado cabaret señorial que era propiedad de su padre. Pero antes de esto, cuando su hermano fernando empezó una carrera en el cine como actor, mario decide quedarse como productor. En 1963, escribe su primer guion para

Annex C. End
Left rectangle: anaphoric references. Center rectangle: breaking the ties. Right rectangle: final text, with anaphoras already replaced.

Mario Almada.- Born in Huatabampo, Sonora, Mexico. In addition of being a great actor, he is also director, writer and movie producer. He started his artistic career in the 1930s in Mexico. He has appeared in more than 200 movies, many of which were classic drama, like his first film *Madre Querida* (1935). In this he appeared as a child with his brother Fernando Almada, who performed as an extra. He would not appear again in a movie until several decades later.

Almada moved from his native Huatabampo to Ciudad Obregón and Guadalajara, where he lived for several years until he finally settled in Mexico City. Since Almada was born in a family linked to films, he is easily exposed to filming and in Mexico City he starts working in a night club called *Cabaret Señorial* which belonged to his father.

But before this, when his brother Ferdinand started a career in the movies as an actor, Mario decides to stay as producer. In 1963, he writes his first screenplay for a movie.

In 1965 Mario performed the role of Bruno the King in the movie Riders of the Witch, which the Almada brothers were producing, and where Fernando was the first actor. Bruno the King became injured during the filming and Mario accepted to take his place.

From 1997 to 2001 he had an important participation with the group The Exterminator for their records “Narco Corridos 2” where he appeared in the cover, and he also conducted dialogues for most tracks in the album. After that, in *El Chile Peláiz* also from 1997, where he reappeared in the cover, in 1999 he performed dialogues for the theme “Smuggling in the eggs” in addition to taking part in the video that was prohibited by SCT, and his last appearance was in the album “Gathering of dogs” where he appeared with some other Mexican actors.

References

[1] Chistopher Kenedy and Branimir Boguraev, 1996. Anaphora for Everyone: Pronominal Anaphora Resolution without a Parser. Proceeding COLING '96 Proceedings of the 16th conference on Computational linguistics – Vol. 1 Pages 113-118. Copenhagen, Denmark.

[2] A. Cuevas “Merging of Ontologies Using Semantic Properties”. Ph. D. Thesis, CIC-IPN. In Spanish, 2006.

[3] A. Cuevas, and A. Guzmán. “Knowledge accumulation through automatic merging of ontologies”, Expert Systems with Applications, Vol. 37, Elsevier Editorial System, USA. ISSN: 0957-4174, 2010, pp 1991-2005.

[4] A. Ferrández et al. “¿Cómo influye la resolución de la anáfora pronominal en los sistemas de búsqueda de respuestas?” RUA. Repositorio Institucional de la Universidad de Alicante. Revistas. Procesamiento del Lenguaje Natural Vol. 26, ISSN. 1135-5948, septiembre 2000, INV-PLN-Artículos de Revistas, España. Web site http://hdl.handle.net/10045/1907 (Last consulted August 21, 2012) In Spanish. pp. 231-238

[5] A. Toral et al. “EAGLES compliant tagset for the morphosyntactic tagging of Esperanto”. In Proceedings of the 5th International Conference on Recent Advances in Natural Language Processing (RANLP). Borovets, Bulgaria, 2005.

[6] D. Jurafsky and J. H. Martin. “Speech and language processing”. Second edition. Pearson-Prentice Hall, 2008.

[7] D. Villanueva et. al. “Using frames to disambiguate prepositions”, Journal Expert Systems with Applications Vol. 40. Elsevier Editorial System, USA, ISSN: 0957-4174. 2013. pp. 598-610.

[8] D. Villanueva and R. Nava, “Herramienta para identificar las raíces de un verbo en una oración y un desambiguador de preposiciones”. B. Sc. thesis No. 2010-0069 ESCOM-IPN. May 2011. In Spanish.

[9] F. Colorado “Mapping words to concepts: disambiguation”. M. Sc. Thesis, CIC-IPN, Mexico. In Spanish, 2008.

[10] F. Salguero and F. Soler “Resolución abductiva de anáforas pronominales”. In David Fernández, Emilio Gómez-Caminero, Ignacio Hernández (Eds.), Estudios de Lógica, Lenguaje y Epistemología. IV Jornadas Ibéricas, Fénix Editora. In Spanish, 2010.

[11] G. Sidorov and O. Olivas, “Resolución de anáfora pronominal para el español usando el método de conocimiento limitado”. Avances en la Ciencia de la computación, 7° congreso internacional ENC-2006, México, In Spanish, 2006, pp. 276–281.

[12] J. Meneses and M. García, "Construction of an analyzer of colloquial sentences and a syntactic tagger of sentences of a document”. B. Sc. thesis No. 2010-0075 ESCOM-IPN. May 2011. In Spanish.

[13] G. López-yebra et. al. “SERCDD. Knowledge extraction and representation from descriptive Spanish documents”. Submitted to Computational Intelligence. 2013.

[14] K. Chistopher and B. Branimir, “Anaphora for Everyone: Pronominal Anaphora Resolution without a Parser”. COLING '96 Proceedings of the 16th conference on Computational linguistics, Copenhagen, Denmark. Vol. 1, 1996. pp 113-118.

[15] L. Shalom and J. H. Leass. “An algorithm for pronominal anaphora resolution”. Computational Linguistics 20, 4, 1994, pp. 535-561.

[16] M. Haliday, and H. Ruqaiya “Cohesion in English Longman”, Group United Kingdom, ISBN-10: 0582550416, ISBN-13: 978-0582550414, July 1976, London

[17] M. Minsky “A Framework for Representating Knowledge”. MIT-AI Laboratory Memo 306, June 1974 Reprinted in The Psychology of Computer Vision, P. Winston, ed. McGrawHill, 1975. Link http://web.media.mit.edu/~minsky/papers/Frames/frames.html (Last consulted August 21, 2012)

[18] M. Palomar et. al. “An Algorithm for Anaphora Resolution in Spanish Texts”. Computational Linguistics - Special issue on computational anaphora resolution Vol. 27 Issue 4, Pages 545-567 MIT Press Cambridge, MA, USA, 2001.

[19] M. Poesio and H. Rieser, “Completions, coordination, and alignment in dialogue”, Dialogue and Discourse 1, 1-89, 2010.

[20 P. Martínez “Resolución Computacional de la anáfora en diálogos: Estructura del discurso y conocimiento lingüístico”, RUA. Repositorio Institucional de la Universidad de Alicante. Revistas. Procesamiento del Lenguaje Natural Vol. 28, ISSN. 1135-5948, mayo 2002 INV-PLN-Artículos de Revistas. España. Web site http://rua.ua.es/dspace/handle/10045/1851 (Last consult ed August 21, 2012) In Spanish.

[21] T. Liang, and D. S. Wu, “Automatic Pronominal Anaphora Resolution in English Texts”. In Computational Linguistics and Chinese Language Proceedings Vol. 9 No. 1, 2004.

[22] R. Morales, “Resolución automática de la anáfora indirecta en español”, Ph. D. Thesis. CIC-IPN. Mexico. In Spanish, 2004.

[23] R. Mitkov, “Anaphora resolution: the state of the art”, Working paper (Based on the COLING'98/ACL'98 tutorial on anaphora resolution), University of Wolverhampton, U.K. 1999.

[24] X. Carreras, et al. “FreeLing: An Open-Source Suite of Language Analyzers”, LREC’04 Proceedings of the 4th International Conference on Language Resources and Evaluation. Vol. 4, Lisbon, Portugal, 2004.

[25] Online source 1 Piano description Archive. Available from: (online): http://es.wikipedia.org/wiki/Piano

[26] Online source 2 Burro flautista description Archive. Available from: (online): http://www.juegosyeducacion.com/fabulas/el_burro_flautista.html

[27] Online source 3 Batalla de Puebla description Archive. Available from: (online): http://www.educar.org/comun/efemerides/Mexico/5dem ayomexico

[28] Online source 4 Francisco Villa description Archive. Available from: (online): http://www.biografiasyvidas.com/biografia/v/villa.htm

[29] Online source 5 Mario Almada description Archive. Available from: (online): http://es.wikipedia.org/wiki/Mario_Almada

[30] Online source 6 Spanish WordNet 3.0 description Archive. Available from: (online): http://sinai.ujaen.es/timm/wiki/index.php/Spanish_WordNet_3.0

[31] Online source 7 TnT -- Statistical Part-of-Speech Tagging. Thorsten Brants. Available from: (online): ww.coli.uni-saarland.de/~thorsten/tnt/