TRADUCCIÓN AUTOMÁTICA: CONCEPTO Y APLICACIÓN PRÁCTICA EN EL SISTEMA LEKTAII.

José Francisco Quesada Moreno

CICA (Centro de Informática Científica de Andalucía), Avda. Reina Mercedes, s/n, Sevilla, Tlfno.: (95) 4623811, email: josefran@cica.es

José Gabriel Amores Carredano
María Teresa López Soto

Universidad de Sevilla, Departamento de Lengua Inglesa, Palos de la Frontera, s/n , 41004 Sevilla, Tlfno : (95) 4551549-4551588, email: gaby@fing.us es; teresa@fing us.es

(Recibido Mayo 1996; aceptado Julio 1996)

BIBLID [1133-682X (1995-1996) 3-4; 131-154.]

Resumen

En este articulo introducimos algunos conceptos generales sobre Traducción Automatica (TA) La TA se incluye en la disciplina conocida como Linguistica Computacional y también en Procesamiento del Lenguaje Natural Su fin primordial es el tratamiento automático de la información linguistica la TA presenta una doble dificultad. en primer lugar, existe un condicionamiento expresado en la estructura misma del lenguaje natural (gramática), en segundo lugar. la dificultad misma del proceso de traducir En TA. el estudio gramatical del lenguaje ha de conjugar dos estrategias la linguística y la computacional En TA se trata de elaborar una teoria del lenguaje formal que, sin perder la coherencia linguistica, mantenga el aspecto práctico exigido en computación LEKTAII es un sistema de TA, inspirado en la teoria linguistica LFG, basado en la gramática de unificación y con una estructura de transferencia con independencia del lenguaje Tambien presentamos algunos ejemplos de análisis y traduccion que prueban la eficiciencia de LEKTAII

Palabras clave: traducción automática, linguistica computacional, procesamiento del lenguaje natural

Abstract

In this paper. we introduce some general concepts on Machine Translation (MT) MT is included in the scientific field known as Computational Linguistics, or Natural Language Processing Its main objective is to process linguistic information automatically This process has to face two major difficulties: the restrictions imposed by the language structure itself (grammar rules) and the translation process MT is in between linguistics and computer science There is a necessity to design a formal language which is able t show natural science There is a necessity to design a formal language which is able t show natural language power of communication and, at the same time, not losing computational efficiency LEKTAII is an MT prototype based on unification-grammar and LFG-inspired. It is a transfer, language independent system We show some examples of analysis and translation done with LEKTAII

Key words: Machine translation, computational linguistics, natural language processing.

Résumé

Dans cet article on introduit quelques concepts généraux sur la Traduction Automatique (TA). La TA est inclue dans le champ commu comme Linguistique Computationale our Processement de l'information linguistique Son premier but c'est de processer l'information linguistique automatiquement Ce procès presente deux difficultes primordiales. les restrictions imposees par la structure du langage

(grammaire) et le procès de traduction luimême Dans la TA, l'étude grammatical du langage doit combiner deux stratégies celle linguistique et celle informatique Dans la TA, il s'agit d'élaborer une theorie de langage formel qui, sans jamais perdre la coherence linguistique, puisse mantenir l'aspect practique exige dans l'informatique LEKTAII est un système de AT inspiré de la théorie lingustique LFG, bassé sur la grammaire d'unification et avec une structure de transèrement indépendante du langage On présente aussi quelques examples d'analyse et de traduction qui prouvent l'efficacité de LEKTAII

Mots clés: traduction automatique, processement de l'information linguistique, linguistique computationale.

Sumario

0. Introducción

El lenguaje natural constituye el modo más rico y flexible de comunicación entre seres humanos. Esta frase, que pudiera parecer obvia, adquiere hoy día un sentido diferente. En un mundo en que la máquina ha sustituido al hombre en multitud de tareas, el lenguaje natural sigue siendo el único capaz de comunicar mensajes completos.

Es importante insistir en ello, pues, en la era de la tecnología informática, han aparecido múltiples lenguajes formales que, reduciendo la potencia expresiva característica de cualquier lenguaje natural, permiten una mayor flexibilidad y eficiencia en su uso computacional. Sin embargo, cuando, para ciertos dominios, la máquina tiene la función primordial de mantener esa expresividad, el lenguaje formal ha de ser diseñado manteniendo, en la medida de lo posible, las características gramaticales del lenguaje natural, como son la capacidad léxica, sintáctica y semántica.

En sistemas que requieren de una "comunicación" entre máquina y usuario, se plantea la necesidad de aprovechar la potencia expresiva del lenguaje natural junto a la eficacia y aspecto práctico que puede suponer un lenguaje formal para su aplicación computacional. Piénsese, por ejemplo, en un sistema telefónico que permita a personas discapacitadas marcar un número de teléfono simplemente diciéndolo de palabra; o en un sistema que permita la consulta a una base de datos expresando la petición mediante voz y en lenguaje natural, y que, una vez consultada la base, mediante traducción automática (TA) se devuelva al usuario la información traducida a la lengua de origen.

En este artículo pretendemos mostrar una visión general del procesamiento del lenguaje natural (PLN) y de la TA. Analizaremos los conceptos de lenguaje y traducción, en concreto en la relación que se crea para el campo de la TA. Por último, se estudiarán algunos problemas lingüísticos concretos, tal y como se resuelven en el sistema de TA LEKTAII.

1. Procesamiento del Lenguaje Natural

A grandes rasgos, se puede considerar que el **Procesamiento del Lenguaje Natural** es la disciplina que asume como objetivo básico el *tratamiento automático de información lingüística*. De hecho, a este campo de estudio también se le conoce con el nombre de **Lingüística Computacional.**

Desde un enfoque lingüístico, se podría entender que el objetivo básico de la Lingüística Computacional no es otro que el *estudio de los lenguajes naturales con el fin de obtener modelos e implementaciones computacionales de estos lenguajes.*

Es difícil obtener una definición adecuada para la noción de lenguaje natural. Normalmente se recurre a una enumeración de propiedades que diferenciarían a éstos por oposición a otros lenguajes formales como son la notación musical, matemática o los lenguajes de programación. Desde este enfoque se habla de la ambiguedad (en cualquiera de los niveles léxico, sintáctico, semántico y pragmático), y en relación con esta característica, aparecen la indeterminación y la consiguiente susceptibilidad de interpretación, la potencia expresiva, junto con la riqueza y la flexibilidad. Se dice que los lenguajes naturales son los efectivamente usados por los hombres para su comunicación interpersonal, usando para ello los órganos fisiológicos alojados en las cavidades bucal y torácica.

2. Traducción Automática

A grandes rasgos, la traducción se puede concebir como el proceso de expresar en un lenguaje el contenido de una expresión formulada originalmente en un lenguaje diferente. Los conceptos que intervienen son básicamente los siguientes:

a) La noción de lenguaje, cuyo estudio exigirá tener en cuenta múltiples disciplinas, entre las que destacan la lingüística y la teoría de lenguajes formales.

b) La posibilidad de relacionar lenguajes, lo que nos llevará a plantear cuestiones como:

> *tipos de relaciones entre lenguajes (¿existen lenguajes homomorfos, isomorfos?, ¿es posible establecer aplicaciones inyectivas, sobreyectivas o biyectivas entre lenguajes?, y, si es posible, ¿a qué nivel: léxico, sintáctico o semántico?, o ¿es necesario involucrar nuevas fuentes de conocimiento para que las aplicaciones sean posibles?);
> * abordar las características de las relaciones entre lenguajes, tanto desde un punto de vista lingüístico (viabilidad y motivación lingüísticas de las relaciones) como computacional (complejidad de la tarea y requerimientos de una posible implementación);

* estudiar la adecuación de los formalismos matemáticos desarrolladospara los lenguajes formales como modelos para la especificación de las características de un lenguaje natural; etc.

2.1. Definición de Lenguaje

Un intento de definición de la noción de lenguaje no es una tarea trivial, debido fundamentalmente a la enorme cantidad de campos donde se usa el término, y a que no todos los usos son congruentes.

A este nivel resulta ilustrativo el análisis de Karttunen y Zwicky, [Karttunen & Zwicky-85], donde estudian el cambio de mentalidad que se ha producido en la concepción del análisis sintáctico o gramatical (*parsing*), fundamentalmente desde mediados de siglo. Los autores señalan cómo han aparecido serias diferencias entre las posiciones de la lingüística tradicional con respecto a las nuevas ideas, representadas por la lingüística formal, la teoría de lenguajes formales, la informática, la inteligencia artificial y la psicolingüística. E incluso analizan las diferencias entre estas disciplinas en su concepción del análisis gramatical.

El análisis de Karttunen y Zwicky para la noción de análisis sintáctico es, en gran medida, extrapolable al estudio del lenguaje, pues, en definitiva, lo que sustenta diferentes concepciones del análisis son formas distintas de entender qué es un lenguaje.

De entre todos los campos relacionados con el estudio del lenguaje, suelen usarse con más frecuencia las nociones desarrolladas en el ámbito de la teoría de lenguajes formales, una rama de las matemáticas, donde se han estudiado gran parte de los conceptos actualmente comunes en PLN, tales como análisis ascendente vs. descendente, determinista vs. no determinista, paralelo vs. secuencia con *"backtracking"*, etc. En el marco de esta teoría, suelen ser frecuentes las definiciones de lenguaje del tipo:

> Un lenguaje sobre un alfabeto **A** es un conjunto de cadenas de **A.**
> Donde un alfabeto es un conjunto finito de signos, y una cadena una secuencia de signos del alfabeto.

Según Aho y Ullman [Aho & Ullman-72], esta definición es capaz de aglutinar casi todas las definiciones de lenguaje. Y añaden: *FRORTAN, ALGOL, PL/I and even English are included in this definition* [Aho & Ullman-72].

Si, una vez equipados con esta noción, aún rudimentaria, de lenguaje, regresamos hasta la idea de traducción, podemos señalar con más acierto los problemas de la traducción:

a) A un nivel elemental podemos constatar que lenguajes diferentes tendrán alfabetos (signos) diferentes, lo que exigirá incorporar módulos morfológicos y léxicos, e incluso tratamiento de grafías especiales;

b) La traducción consistirá, básicamente, en un proceso tal que, dada una cadena de un lenguaje **O** (lenguaje origen), genere una cadena en un lenguaje **D** (lenguaje destino); concebido así, podríamos entonces abordar el estudio de la decibilidad de tal proceso;

c) La cuestión más espinosa surge al tener que decidir qué características de la cadena del lenguaje **O** deben ser mantenidas por la cadena del lenguaje **D** generada tras la traducción, y cómo se podrá medir la adecuación de la traducción; problemas todos ellos muy frecuentes en la traducción de textos literarios, y que ha sido foco de interés para múltiples disciplinas, desde la sociología hasta la filosofía, para corrientes tales como la hermenéutica, y para autores entre los que se puede citar a Merleau-Ponty o W.O. Quine.

Sin ánimo de entrar en una discusión filosófica acerca de la posibilidad de la traducción, la conclusión que se puede obtener es que el proceso de traducción se puede concebir como el traspaso de estructuras complejas representadas en un código (el propio del lenguaje **O**) hasta otras estructuras en otro código (del lenguaje **D**), las cuales deben mantener ciertos rasgos, como, por ejemplo, el número de signos, cierto cómputo matemático sobre propiedades de los signos de ambos lenguajes (en algún sentido, encriptar es traducir), o, normalmente, características no tan bien definidas como el significado.

2.2. Definición de Gramática

Tanto si se habla de traducción, como de TA, el análisis realizado nos permite concluir que la complejidad de la traducción va a estar directamente relacionada con la complejidad de los lenguajes que se pretenden traducir.

Para entender la clasificación de Chomsky únicamente hay que tener en cuenta que la noción de lenguaje, tal y como se presentó anteriormente, encaja perfectamente en la teoría de conjuntos, es decir, un lenguaje no es más que un conjunto (posiblemente infinito) cuyos elementos se obtienen mediante combinaciones de un conjunto (finito forzosamente) llamado alfabeto del lenguaje. Ahora bien, si en lugar de representar un lenguaje indicando todos sus elementos (definición extensiva), logramos establecer una o más propiedades de sus elementos (definición intensiva), habremos definido el mismo lenguaje, pero habremos obtenido una notación más útil; y ésta es la idea de gramática [Aho & Ullman-72], [Partee, Meulen & Wall-90].

2.3. Lenguajes y Traducción

Existe un condicionamiento completo entre la estructura de un lenguaje (su gramática) y la complejidad del proceso de traducción.

El proceso de compilación en informática y el proceso lingüístico de traducción comparten ciertas características. En teoría de compiladores es usual considerar un lenguaje de programación como un lenguaje generado por una gramática libre de contexto o independiente de contexto. Así por ejemplo lo expresan Aho y Ullman en uno de los textos básicos sobre teoría de compiladores [Aho & Ullman-77:145]:

> Chapter 4 showed how a context free grammar can be used to define the syntax of a programming language.

De hecho, es fácil concebir la compilación como un proceso de traducción, y así lo entienden Dershem y Jipping al hablar de Language translation [Dershem & Jipping-90:25]:

> The purpose of a language translation program is to accept a set of instructions written in a programming language and to cause the activities specified by these instructions to be carried out by the receiving computer. Therefore, a language translation program accepts as input a program in a programming language and makes it possible for the activities specified by the input program to be carried out, either by translating the program into an equivalent program in a language that is already executable, or by carrying out the activities in the translation program itself.

La clave de los procesos de compilación es que los lenguajes de entrada y salida poseen unas características, fundamentalmente la ausencia de ambigüedad estructural y léxica, fuertes restricciones sobre las producciones (que en muchos casos no llegan ni incluso a cubrir todas las posibilidades de los lenguajes libres de contexto), etc., que los hacen especialmente adecuados para ser sometidos a procesos de análisis y generación por parte de ordenadores.

Quizás porque la traducción entre este tipo de lenguajes ha dejado de ser un problema serio, lo frecuente es que la TA se asocie con lenguajes naturales, lenguajes que se suelen situar entre los libres y los sensibles al contexto, y que incluyen construcciones cuyo análisis y generación presentan grandes problemas, tanto a nivel lingüístico, como computacional.

La TA se sitúa a medio camino entre la lingüística y la informática, en una zona conocida como PLN. Es aquí donde cobra sentido la definición de TA dada por Hutchins y Somers [Hutchins & Somers-92:2]:

> MT (Machine Translation) is part of a wider sphere of pure research in computer-based natural language processing in Computational Linguistics and Artificial Intelligence, which explore the basic mechanisms of language and mind by modelling and simulation in computer programs.

El siguiente problema que automáticamente aparece es que el procesamiento automático del lenguaje natural resulta bastante complejo, entre otras cosas porque como indica Amores [Amores-92:9]:

> MT is differente from other computer applications, though.
> It attempts to achieve a mechanical simulation of a human
> process of which we know very little ourselves.

Los problemas de la TA son prácticamente todos los del PLN, en la medida en que la TA requiere de casi todos los componentes usuales en el PLN, que van desde el reconocimiento del texto fuente hasta la generación del destino. Además, la TA, ya que se plantea la implementación real, debe abordar todos los fenómenos de los lenguajes naturales, y encontrar soluciones para ellos.

Aunque son muchos los problemas o fenómenos lingüísticos a los que se enfrenta la TA, la siguiente relación contiene algunos de los más significativos:

* La ambigüedad léxica;

* El conocimiento del dominio (conocimiento técnico) y el conocimiento de sentido común, que se incorporan implícitamente en los textos, y que deben ser tenidos en cuenta para la traducción,

* La complejidad sintáctica para la especificación de los lenguajes involucrados;

* Las diferencias léxicas entre lenguajes;

* El tratamiento de entradas gramaticalmente incorrectas, pero informativas;

* Elipsis, anáforas y otros fenómenos lingüísticos que dan lugar a ambigüedad estructural;

* El problema del significado, es decir, extraer el significado del texto de entrada y generar la cadena de salida con el mismo significado obtenido.

Este último problema citado se suelen considerar como uno de los puntos más espinosos, no sólo en la TA, sino en la Lingüística Computacional y en la Inteligencia Artificial en general. En TA, el problema reside en la naturaleza misma del fenómeno de la traducción: "traducir require comprender"

2.4. Estrategias para la TA

La mayoría de los estudios recientes acerca de la TA reconocen tres enfoques básicos a la hora de abordar el diseño de sistemas computacionales para la traducción:

2.4.1. Traducción Directa

Desde un punto de vista cronológico, los primeros sistemas de TA trataban el problema centrándose en el componente computacional, es decir, en la implementación.

Se partía de un lenguaje fuente y uno de destino, y, a continuación, se empezaba a escribir el código necesario para generar las traducciones de las distintas estructuras.

Como consecuencia, el sistema era totalmente dependiente de los lenguajes elegidos, y, por tanto, el conocimiento lingüístico (léxico, sintáctico, semántico, etc.) involucrado en la tarea, quedaba diluido en el seno de la estructura procedimental.

SYSTRAN [Toma-76] es uno de los primeros sistemas desarrollados con este enfoque.

2.4.2 Modelo Interlingua

El modelo interlingua asume una fuerte suposición teórica acorde con el innatismo chomskiano, como es la posibilidad de encontrar un modelo universal, válido para representar el contenido completo de una expresión en cualquier lenguaje. A ese lenguaje se le denomina interlingua.

Desde esta posición, la traducción se concibe como un proceso en dos momentos:

* desde el lenguaje fuente hasta el interlingua: análisis;
* desde éste al lenguaje destino: generación.

Este enfoque supone interesantes ventajas sobre el modelo de la traducción directa tanto a nivel lingüístico como computacional.

Lingüísticamente, el sistema prevé la existencia de determinados componentes, de índole declarativa, destinados a contener el conocimiento que describe cada lenguaje en cuestión.

Computacionalmente, se aumenta la modularidad del sistema y la independencia entre lenguajes. Añadir un lenguaje a un modelo interlingua supone desarrollar dos componentes para este nuevo lenguaje: uno de análisis que se encargará de traducir las entradas del lenguaje en cuestión hasta una representación correcta según la especificación del linterlingua, y un segundo módulo de síntesis o generación que modele un salida correcta en el lenguaje a partir de una representación correcta en interlingua. Por tanto, añadir un nuevo lenguaje en el modelo interlingua no debe implicar ningún cambio en los componentes de análisis y generación del resto de lenguajes.

Una de las implementaciones que sigue este modelo es PIVOT [Okumura, Muraki & Akamine-91].

2.4.3. Traducción Basada en Transfer

El tercer modelo tiene menores asumpciones teóricas y sus pretensiones a nivel de implementación son asimismo menos ambiciosas.

En este caso la traducción se estructura en tres fases: análisis, transferencia y generación. Las etapas de análisis y generación son totalmente autónomas para cada lenguaje, mientras que los componentes de transferencia son específicos para cada par de lenguajes.

Añadir un nuevo lenguaje a un sistema desarrollado según este modelo exigirá implementar sus componentes de análisis y generación, pero, además, el diseño de un módulo de transferencia entre este nuevo lenguaje y cada uno de los lenguajes ya implementados.

EUROTRA [Allegranza-91] es uno de los sitemas más conocidos que ha seguido este enfoque.

3. Análisis Lingüístico y Generación en TA

Como se ha puesto de manifiesto al estudiar las estrategias para la implementación de sistemas de TA, el análisis es uno de los componentes básicos e indispensables de todo sistema.

A grandes rasgos, el análisis se puede concebir como el estudio de la gramaticalidad de una entrada, es decir, el decidir si una entrada es o no correcta según la especificación de una gramática; y en caso de que la entrada sea gramaticalmente correcta, generar algún tipo de representación para el posterior tratamiento de esa información.

Concebido así, el análisis para el PLN asume los mismos objetivos globales que el módulo de análisis de un compilador. No obstante, cuando se pasa desde la globalidad al detalle, los problemas a que se enfrentan los analizadores para lenguajes naturales y los correspondientes para lenguajes formales, como los de programación, son muy pronunciados.

Para los compiladores, uno de los problemas que debe resolver el analizador sintáctico lo constituye la evaluación de expresiones con operadores infijos [Aho & Ullman-77]. No obstante, la definición de una tabla de precedencia de operadores y la utilización de signos especiales para la separación, como los paréntesis, eliminan el problema.

Para los lenguajes naturales no se pueden ofrecer soluciones tan elegantes y rápidas; en primer lugar, porque no es sencillo encontrarlas, y, en segundo lugar, porque enriquecido con ese tipo de suposiciones para la eliminación automática de ambigüedades, tal vez

dejaría de ser un lenguaje natural. Como un ejemplo bastante rudimentario del tipo de fenómenos que pueden aparecer en el análisis de lenguajes naturales, considérese la oración (i):

(i) Juan vio a la niña con el telescopio.

(i) es un ejemplo clásico que ilustra la ambigüedad estructural de construcciones del tipo VERBO (V) + SINTAGMA NOMINAL (SN) + SINTAGMA PREPOSICIONAL (SP); donde el SP puede depender bien del verbo (*ver con el telescopio*), o, bien, del SN (*la niña llevaba el telescopio*).

Sin embargo, si se estudia el ejemplo, se puede concluir que la ambigüedad que presenta no se debe únicamente a razones sintácticas o estructurales, sino que intervienen factores léxicos y semánticos, que hacen que (ii) y (iii) no sean ambiguas a pesar de tener una estructura idéntica:

(ii) Juan vio a la niña con la muñeca
(iii) Juan levantó la caja con la palanca

e incluso factores propios del nivel de discurso o pragmáticas, como sucede en (iv) y (v):

(iv) El día de Navidad, Juan vio que el regalo de su sobrina era un telescopio.
(v) Mi amigo Juan es muy aficionado a observar a la gente. Tiene un telescopio y observa desde el balcón a todo el que pasa por la calle.

La conclusión que se puede obtener desde aquí es que el análisis gramatical podrá necesitar en muchos casos de información que va más allá de lo sintáctico para un funcionamiento adecuado. El análisis de elipsis y anáforas es asimismo una fuente nada trivial de dificultades para el tratamiento del lenguaje natural, como se desprende de la consideración del término *"ella"* en (vi) y (vii):

(vi) Marta dijo que ella llegó a las seis.
(vii) Luis dijo que Ana no regresó hasta tarde. Marta dijo que ella llegó a las seis.

Por último, hay que señalar que la TA se debe enfrentar a este tipo de problemas tanto a nivel teórico como práctico, es decir, si un sistema puede procesar entradas como (v)-(vii), debe ser capaz de resolver las ambigüedades, y esto quiere decir que el sistema debe estar equipado con la información necesaria para poder hacerlo.

De esta forma queda perfilado el análisis como un problema en nada trivial que exigirá para casos reales el tomar en cuenta múltiples niveles de conocimiento, entre los que podemos señalar:

* componentes morfológicos,

* léxicos,
* sintácticos,
* del dominio,
* de sentido común, etc.

Pero, si desde una perspectiva lingüística el análisis es complejo, también lo es cuando se aborda una implementación de lo que se denomina un analizador (*parser*). Los sistemas deben incorporar componentes capaces de manipular conocimiento morfológico, léxico, sintáctico, semántico y pragmático. En esta sección describiremos las características básicas de estos niveles de conocimiento, haciendo un especial hincapié en las denominadas gramáticas de unificación, uno de los paradigmas de la lingüística computacional actualmente más extendido.

3.1. Análisis Morfológico y Léxico

El campo de trabajo de la lexicografía computacional [Hartmann-84] está constituido por los denominados lexicones, "diccionarios en ordenador", o "diccionarios computerizados" [Russell et al.-86].

Desde el punto de vista morfológico, se deben almacenar los modelos de flexión asociados con cada palabra, o, más bien, con cada forma canónica o lema. Así, por ejemplo, se deben representar las reglas que rigen la conjungación de cada verbo, la formación del plural para nombres comunes, o los cambios de género y número para los adjetivos.

A partir de esta información, un analizador morfológico podría determinar que "*reads*" es la tercera persona del singular del presente simple del verbo "*to read*". Pero además de esta escueta información morfosintáctica (utilizada por el analizador sintáctico), existe mucha más información que el componente léxico debe extraer del diccionario y pasar a los módulos superiores (los analizadores semántico y pragmático). Desde el punto de vista del patrón de subcategorización, "*reads*" se rige por el modelo "*someone reads something*"; además, salvo en casos figurativos, "*someone*" se refiere normalmente a un ser humano, mientras que "*something*" debe tener la característica de "*poder ser leído*". Así, el sujeto ha de concordar en número y persona con el verbo.

Para recuperar toda esta información se han desarrollado múltiples formalismos, entre los que destaca el basado en estructuras de rasgos para gramáticas de unificación [Shieber-86].

Formalmente, una estructura de rasgos es un conjunto de pares atributo-valor, donde se permite que el valor asociado con un atributo sea a su vez una estructura de rasgos. De forma similar, una estructura de rasgos se puede definir como un grafo acíclico dirigido, cuyos arcos (atributos) y nodos (valores) están etiquetados. Asociada con cada definición existe un modo de representación: forma matricial y grafo. A continuación se muestran los dos tipos de representación para una estructura muy sencilla·

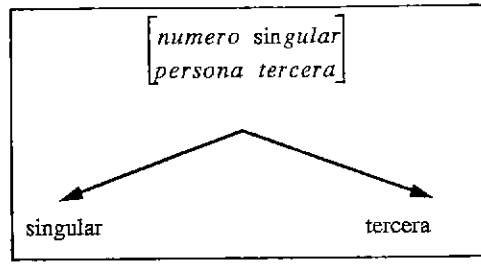

Figura1

Estas estructuras pueden entrar a formar parte de otras más complejas aprovechando la recursividad inherente de la definición dada para las estructuras de rasgos:

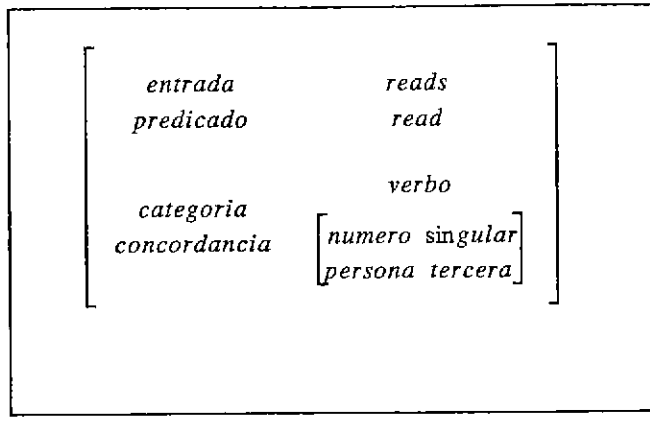

Figura2

Los formalismos para la especificación de léxicos basados en estructuras de rasgos incluyen operadores para la negación, opcionalidad, disyunción y co-referencia, mediante los cuales se aumenta considerablemente la potencia expresiva del modelo. Haciendo uso de esta última característica, la co-referencia, se puede expresar la concordancia de número y persona entre el verbo y su sujeto:

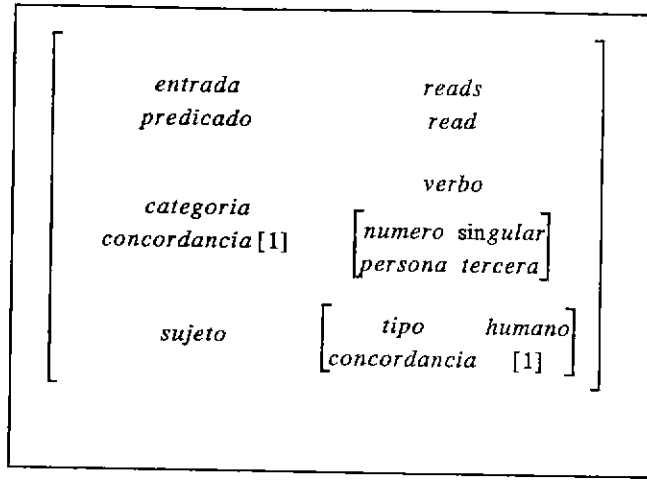

Figura3

3.2. Análisis Sintáctico (parsing)

Desde el punto de vista de su análisis, una expresión no es más que una cadena de signos del lenguaje en el que se está trabajando. Esta cadena de signos es analizada inicialmente por el componente morfológico y léxico, el cual, a partir del diccionario de lemas y las reglas de flexión asociadas, debe reconocer cada una de las palabras y devolver toda la información disponible.

De cara al uso que el analizador sintáctico (*parser*) hará de esta información, sólo es relevante la categoría sintáctica asociada con cada palabra, por lo que el analizador morfológico y léxico se ve desde el parser como un "*tokenizador*" que suministra una cadena de símbolos de entre el conjunto de símbolos terminales del lenguaje. El parser asumirá como objetivo la determinación de la corrección gramatical de la cadena de símbolos, y en caso de que ésta sea correcta, devolverá el árbol de análisis asociado.

Supongamos que disponemos de un analizador morfológico y léxico capaz de devolver la secuencia:

np v det n p n

El objetivo del parser será determinar si esta secuencia es correcta de acuerdo con la gramática, y, si es así, lo cual es cierto para este caso, construir el árbol de análisis:

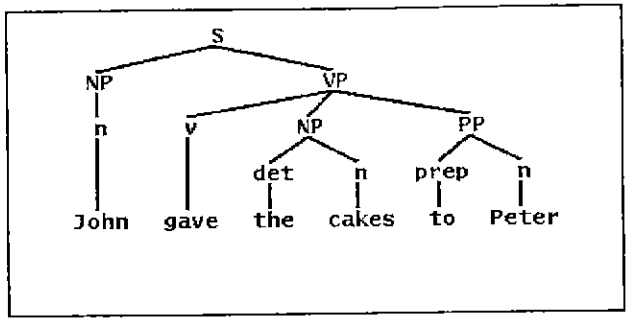

Figura4

De forma similar, el parser deberá rechazar como sintácticamente mal formada la secuencia:

John the cakes gave Peter

3.3. Análisis Semántico.

Además de la corrección gramatical que analiza el módulo sintáctico, una cadena de símbolos debe cumplir más requisitos para que se considere correcta en un lenguaje. Así, consideramos las expresiones (ix) y (x):

(viii) John opened the can with a hammer.
(ix) John opened the air with a hammer.

Se observa que tanto (ix) como (x) son sintácticamente correctas, aunque semánticamente (x) es incorrecta. Asimismo, considerando la sintaxis tal y como se ha definido en la sección anterior, es el módulo semántico el que deberá rechazar las siguientes secuencias de palabras:

(x) John gave two cake to Peter
(xi) John have given the cakes to Peter

Pero la semántica no es meramente un filtro que descarta las construcciones incorrectas, entre sus objetivos se encuentra también el obtener una representación del signicado para las expresiones correctas, problema para el cual se han desarrollado múltiples modelos y técnicas, entre las que pueden enumerarse:

* Las representaciones basadas en lógica [Sant-Dizier-94].
* Las redes semánticas [Quilliam-68].
* Las estructuras de rasgos.

Aquí nos centraremos en este tercer modelo, el cual ha servido de base a todo un paradigma, las denominadas gramáticas de unificación, las cuales combinan un módulo de parsing o análisis de constituyentes (que usa únicamente información categorial) junto con otro módulo basado en la operación de unificación sobre estructuras de rasgos [Knight-89].

Con estas herramientas formales se puede enriquecer el modelo gramatical incluyendo ecuaciones funcionales que detallen el comportamiento semántico del lenguaje, y que, usando la operación de unificación, permiten rechazar construcciones semánticamente incorrectas, a la vez que generan la estructura de rasgos (que pasa a funcionar como representación semántica) de la expresión completa.

Entre los formalismos y teorías gramaticales que siguen este enfoque se encuentran:

* **DCG** (*Definite Clause Grammar*) [Pereira & Warren-80];
* **LFG** (*Lexical-Functional Grammar*) [Bresnan-82];
* **FUG** (*Functional Unification Grammar*) [Kay-83];
* **GPSG** (*Generalized Phrase Structure Grammar*) [Gazdar et al.-85];
* **PATR-II** [Shieber-86] y
* **HPSG** (*Head-Driven Phrase Structure Grammar*) [Pollard & Sag-94].

3.4. Análisis Pragmático

Este módulo se encargaría de poner en relación la representación del significado de una frase con el contexto en el que aparece, con el fin de resolver ciertas ambigüedades o lagunas de información provocadas por fenómenos lingüísticos tales como la elipsis o la anáfora.

El análisis pragmático está íntimamente relacionado con la representación del discurso, y, para este problema, los modelos teóricos y formales son aún muy limitados. Las características de este problema lo hacen sumamente complejo, pues interviene conocimiento de sentido común compartido por los interlocutores que funciona como información implícita; además, el conocimiento de cada interlocutor evoluciona durante el diálogo, con lo cual son necesarios sistemas dinámicos o no monótonos de representación de conocimientos y creencias, etc.

Entre las teorías para las que se están desarrollando modelos computacionales merece destacarse la teoría de representación del discurso (**DRT: *Discourse Representation Theory***) [Kamp-81].

3.5. Generación

El problema de la generación consiste, a grandes rasgos, en un recorrido inverso de los módulos de análisis presentados. Partiendo de una representación situada en los niveles semántico/pragmático, un generador de lenguaje natural deberá obtener la estructura de constituyentes o árbol de análisis, y, a partir de éste, generar las palabras mismas que formarán la cadena de salida mediante un proceso de generación léxica y morfológica.

De nuevo, han sido muchos los modelos y formalismos desarrollados para abordar este problema. Centrándonos en el ámbito de las gramáticas de unificación, uno de los temas de investigación más interesantes lo constituye la obtención de formalismos reversibles, es decir, conseguir que la especificación de gramáticas pueda usarse tanto para el análisis como para la generación [Zajac-94].

4. Análisis Sintáctico y Generación en el Sistema de TA LEKTAII

A continuación mostraremos el diseño del sistema de TA LEKTAII. LEKTAII ha sido utilizado como prototipo en un proyecto conjunto entre la Unviersidad de Sevilla y Telefónica I+D. LEKTAII sigue el modelo de traductor de tercer tipo, es decir, basado en transfer. La teoría lingüística que inspira el modelo es la llamada Léxico-Funcional, o **LFG** (*Lexical-Functional Grammar*) [Bresnan, ed.-82]. La mayor ventaja en utilizar LFG en TA reside en la doble representación que LFG asigna a cada oración.

En primer lugar, la *estructura-c* o **estructura de constituyentes** ofrece la configuración más superficial de la oración, que se genera a partir de un parser libre de contexto. A continuación se obtiene una representación más abstracta, *estructura-f* o **estructura funcional**, que asigna las relaciones gramaticales expresadas en forma de matriz de pares atributo-valor. Nuestra teoría es que la *estructura-f* constituye el input ideal para incluir en el módulo de transferencia para cualquier sistema convencional de TA basado en transferencia.

Mientras que la *estructura-c* proporciona información necesariamente dependiente de la estructura lingüística de la frase de origen, ésta es descartada durante la fase de transferencia. Las relaciones gramaticales presentadas en la *estructura-f* ofrecen información independiente del lenguaje de origen. Este tipo de información es el más adecuado para ser utilizado en el módulo de transferencia entre la lengua de origen y la lengua de destino, pues incluye una representación que puede ser compatible para ambas lenguas.

La arquitectura de LFG nos permite manipular la información sintáctica y semántico-pragmática de manera separada. De esta manera, se hace más fácil diseñar sistemas bilingües y cambiar de un dominio semántico a otro sin demasiadas complicaciones.

A pesar de que existen problemas lingüísticos aún por resolver, la posibilidad de construir sistemas de TA con base lingüística, constituye un punto interesante de

conocimiento para la investigación en el campo. El mayor problema reside en proporcionar al mismo tiempo robustez y eficiencia al sistema informático sin por ello perder la calidad empírica de la información lingüística aportada. Es aquí cuando no nos podemos mostrar tan optimistas, aunque también es posible llegar a una postura de compromiso entre lo estrictamente correcto desde el punto de vista lingüístico y lo esencialmente práctico desde el punto de vista computacional. De esta manera, diversos enfoques han ido apareciendo para intentar paliar en cierta medida las limitaciones que existen cuando nos enfrentamos al duro campo de la TA. En este sentido, LEKTAII ha incorporado varios de estos enfoque para la resolución de problemas concretos. Entre ellos cabe destacar los buenos resultados obtenidos en la descripción morfológica y semántica a partir de estructuras semánticas e información estadística basada en el uso contextual de las unidades léxicas. Esto se verá con más detenimiento en el último apartado, donde incluimos algunos ejemplos de oraciones analizadas y traducidas por el sistema.

En cuanto al diseño base, LEKTAII es una herramienta informática escrita en lenguaje C. Esta programación ha aportado al sistema eficiencia y ha mantenido al mismo tiempo una cierta elegancia lingüística. Se trata de cuestiones de una relevancia capital, pues hablamos de un sistema que ha de generar traducción en tiempo real. (En el análisis de las oraciones incluimos el tiempo empleado por la máquina en realizar la traducción completa).

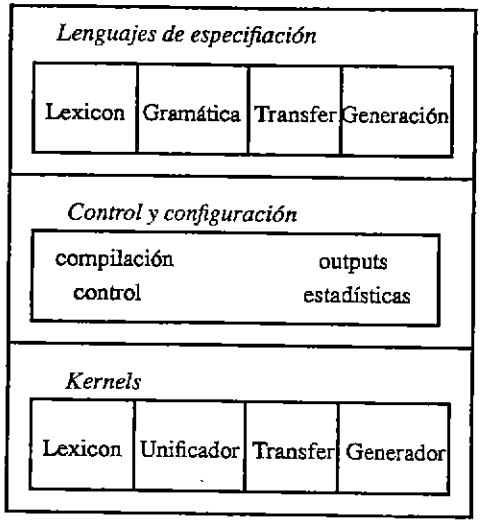

Figura5

4.1. Arquitectura General de LEKTAII

Existen tres pilares básicos que ayudan a mantener la potencia expresiva sin perder la eficiencia. El nivel inferior ("*traducción kernel*") contiene los siguientes submódulos: administrador de datos léxicos, parser, unificador, transfer y generador. El nivel intermedio o de control y configuración se dedica a la compilación de los lenguajes de especificación, "trace", "output", estadísticas y "setup". Por último, en el nivel más superior encontramos los denominados lenguajes de especificacón, en los que se definen el léxico y la gramática de análisis, la transferencia lingüística y las reglas de generación.

La especificación del lexicon permiten el uso de macros, listas, disyunción, negación, etc. Las reglas gramaticales siguen la clásica notación LFG. Las ecucaciones funcionales asociadas a cada regla se aumentan en un lenguaje de control que permite el uso de construcciones del tipo IF-THEN-ELSE, operadores lógicos, relacionales y matemáticos, funciones de tipo cadena y de tipo lista (MEMBER o CONCAT) y funciones específicas para controlar la definición de las *estructuras-f* (COMPLETENESS y COHERENCE). El módulo de transferencia (desde la *estructura-f* en lenguaje de origen a la *estructura-f* en el lenguaje de destino) permite una definición uniforme de las reglas de transferencia estructural y léxica. Cada regla puede ir asociada a diversas condiciones y/o acciones. Las condiciones se disparan según sea el tipo de *estructura-f* asociada. Las acciones se refieren a la manipulación de la *estructura-f* resultante mediante funciones específicas del tipo NOTRANSFER, TRANSFERAS y de sobreescritura. Las reglas de generación (desde la *estructura-f* de origen a la *estructura-c* de destino) asignan una *estructura-c* a la *estructura-f* entrante dependiendo de los atributos existentes.

4.2. Proceso de Traducción: Algunos Ejemplos

En esta sección incluimos varios ejemplos de análisis sintáctico-semántico y traducción hechos en LEKTAII. En las figuras 6 y 7 se muestran los árboles de análisis sintáctico para las oraciones:

(xii) Show me the flights from Boston to San Francisco
(xiii) Activa el modo no molesten

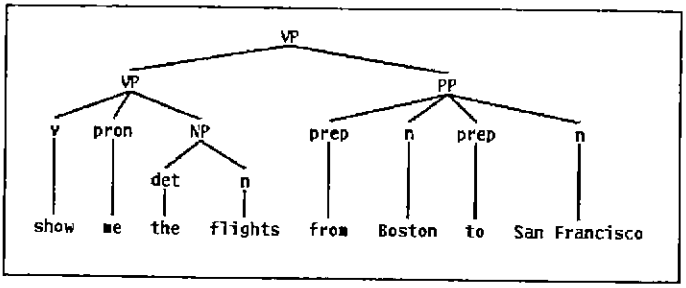

Figura6

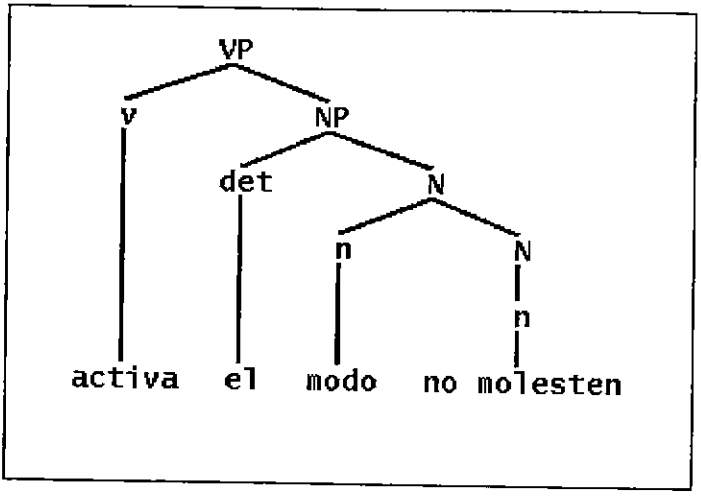

Figura 7

Es decir, lo que en terminología LFG se conoce como *estructura-c*. En las figuras 8 y 9 se muestra el análisis de *estructura-f* para esas mismas oraciones.

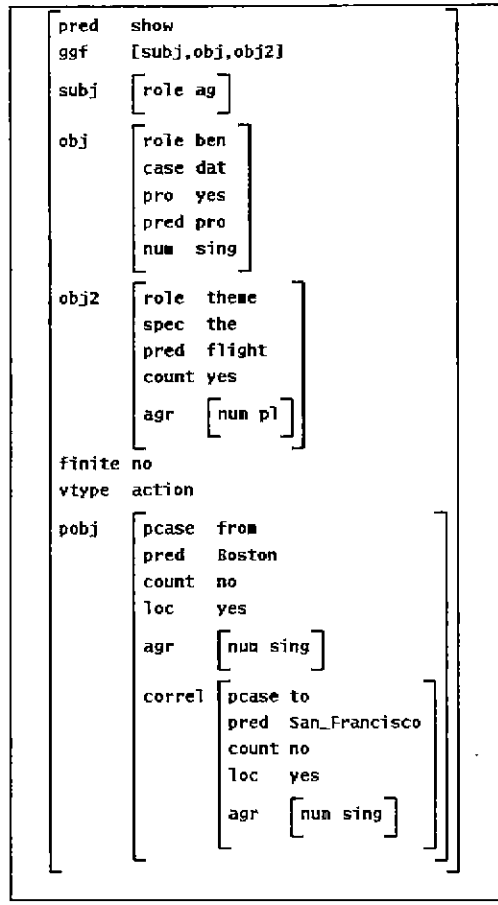
pred show
ggf [subj,obj,obj2]
subj [role ag]
obj [role ben
case dat
pro yes
pred pro
num sing]
obj2 [role theme
spec the
pred flight
count yes
agr [num pl]]
finite no
vtype action
pobj [pcase from
pred Boston
count no
loc yes
agr [num sing]
correl [pcase to
pred San_Francisco
count no
loc yes
agr [num sing]]]

Figura8

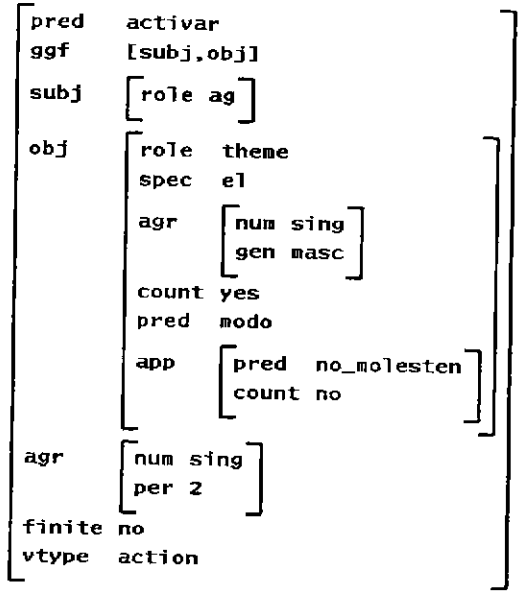

Figura 9

En cuanto a la traducción, aquí incluimos algunos ejemplos de frases traducidas del español al inglés. Se incluye el tiempo real empleado por el sistema en conseguir la traducción.

```
LektaII> Leyendo fichero de configuracion: lekta.ini
@LktRe> (Parsing:COR)  (T=0.030) (P=6) (T/P=0.0050) (E-f:1)
@LktRe> (Transfer)  (T=0.010) (E=29) (T/E=0.0003)
@LktRe> (Generac.)  (T=0.000) (E=29) (T/E=0.0000)
@LktRe> (Transl)
(dónde está el banco más cercano)
     ====>
(where is the closest bank)


@LktRe> (Parsing:COR)  (T=0.040) (P=9) (T/P=0.0044) (E-f:1)
@LktRe> (Transfer)  (T=0.000) (E=38) (T/E=0.0000)
@LktRe> (Generac.)  (T=0.010) (E=38) (T/E=0.0003)
@LktRe> (Transl)
(aquí tiene el pasaporte y mi tarjeta American Express)
     ====>
(here is the passport and my American Express card)


@LktRe> (Parsing:COR)  (T=0.010) (P=2) (T/P=0.0050) (E-f:1)
@LktRe> (Transfer)  (T=0.010) (E=18) (T/E=0.0006)
@LktRe> (Generac.)  (T=0.000) (E=18) (T/E=0.0000)
@LktRe> (Transl)
(cuándo abren)
     ====>
(when do you open)


@LktRe> (Parsing:COR)  (T=0.030) (P=6) (T/P=0.0050) (E-f:1)
@LktRe> (Transfer)  (T=0.010) (E=20) (T/E=0.0005)
@LktRe> (Generac.)  (T=0.000) (E=19) (T/E=0.0000)
@LktRe> (Transl)
(abrimos a las cinco menos diez)
     ====>
(we open at 4.50)


@LktRe> (Parsing:COR)  (T=0.000) (P=3) (T/P=0.0000) (E-f:1)
@LktRe> (Transfer)  (T=0.000) (E=1) (T/E=0.0000)
@LktRe> (Generac.)  (T=0.010) (E=1) (T/E=0.0100)
@LktRe> (Transl)
(no le entiendo)
     ====>
(I don't understand)


@LktRe> (Parsing:COR)  (T=0.030) (P=6) (T/P=0.0050) (E-f:1)
@LktRe> (Transfer)  (T=0.010) (E=29) (T/E=0.0003)
@LktRe> (Generac.)  (T=0.000) (E=29) (T/E=0.0000)
@LktRe> (Transl)
(dónde está el banco más cercano)
```

De los ejemplos mostrados podemos concluir que LEKTAII puede ser considerado como un sistema eficaz de TA en tiempo real.

Referencias

[Aho & Ullman-72] AHO, Alfred V. & Jeffrey D. ULLMAN. 1972. The Theory of Parsing, Translation and Compiling. Volume I: Parsing. Englewood Cliffs, NJ: Prentice-Hall.

[Aho & Ullman-77] AHO, Alfred V. & Jeffrey D. ULLMAN. 1977. *Principles of Compiler Design. Readings* MA: Addison-Wesley.

[Allegranza-91] ALLEGRANZA, V. P. BENNETT, F. DURAND, F. van EYNDE, L. HUMPHREYS, P. SCHMIDT & E. STEINER. 1991. *Linguistics for Machine Translation: The Eurotra Linguistic Specifications.* En Copeland (ed): Studies in Machine Translation and Natural Language Processing. Bruselas: Commission of the European Communities.

[Amores-92] AMORES, Gabriel. 1992. A Lexical-Functional Grammar-Based Machine Translation System for Medical Abstracts. Universidad de Sevilla, tesis doctoral.

[Bresnan-82] BRESNAN, J. (ed). 1982. The Mental Representation of Grammatical Relations. Cambridge, Mass. MIT Press.

[Dershem & Jipping-90] DERSHEM, Herbert L. & Michael J. JIPPING. 1990. Programming Languages: Structures and Models. Belmont: Wadsworth Publishing Company.

[Gazdar et al.-85] GAZDAR, G. E. KLEIN, G. K. PULLUM & I. A. SAG. 1985. Generalized Phrase Structure Grammar. Cambridge, Mass.: Harvard University Press.

[Hartmann-84] HARTMANN, R. K. K. 1984. *LEXeter '83 Proceedings: papers from the International Conference on Lexicography.* Tübingen: Niemeyer

[Hutchings & Somers-92] HUTCHINS, W. John & Harold L. SOMERS. 1992. An Introduction to Machine Translation. San Diego, CA: Academic Press.

[Kamp-81] KAMP, H. 1981. *A Theory of Truth and Semantic Representation.* En Groenendijk, J., T. Janssen & M. Stokhof (eds). Formal Methods in the Study of Language. Amsterdam: Mathematical Centre Tracks, **135**:277-320.

[Karttunen & Zwicky-85] KARTTUNEN, Lauri & Arnold M. ZWICKY. 1985. *Introduction.* En Dowty, David R., Lauri Karttunen & Arnold M. Zwicky. Natural Language Parsing, Psychological, Computational and Theoretical Perspectives. Cambridge: Cambridge University Press.

[Kay-83] KAY, M. 1983. *Unification Grammar* Technical Report, Xerox Palo Alto Research Center.

[Knight-89] KNIGHT, K. 1989. "Unification: Multidiscipinary Survey". *ACM Computing Surveys,* **21(1)**:93-124.

[Okumura, Muraki & Akamine-91] OKUMURA, A., K. MURAKI & S. AKAMINE. 1991. "Multilingual Sentence Generation from the PIVOT Interlingua". En *Proceedings of MT Summit III.* Washington.

[Partee, Meulen & Wall-90] PARTEE, Barbara H., Alice ter MEULEN & Robert E. WALL. 1990. Mathematical Methods in Linguistics. Dordrecht: Kluwer Academic Publishers.

[Pereira & Warren-80] PEREIRA, F.C.N & WARREN. 1980 "Definite Clause Grammars for Language Analysis--A Survey of the Formalism and a Comparison with Augmented Transition Networds". *Artificial Intelligence,* **13**:231-78.

[Pollard & Sag-94] POLLARD, C. & I. SAG. 1994. Head-Driven Phrase Structure Grammar. Chicago: Chicago University Press.

[Quilliam-68] QUILLIAM, M.4. 1968 *Semantic Memory* En Minsky, M. (ed). Semantic Information Processing. Cambridge, Mass.: MIT Press, p.227-70.

[Russell et al.-86] RUSSELL, G., S. PULMAN, G. RITCHIE & A. BLACK. 1986. "A Dictionary and Morphological Analyser for English". *Proceedings of the 11th International Conference on Computational Linguistics,* **COLING-86**:277-9.

[Sant-Dizier-94] SANT-DIZIER, P. 1994. Advanced Logic Programming for Language Processing. Academic Press.

[Shieber-86] SHIEBER, S.M. 1986. *An Introduction to Unification-Based Approaches to Grammar.* Stanford, CA: Center for the Study of Language and Information. CSLI. Lecture Notes Series.

[Toma-76] TOMA, P. 1976. *An Operational Machine Translation System.* En Briwlin, R. W. (ed). Translation and Research. New York: John Wiley and Sons.

[Zajac-94] ZAJAC, R. 1994. *A Uniform Architecture for Parsing, Generation and Transfer.* En Strzalkowski, T. (ed). Reversible Grammar in Natural Language Processing. Kluwer Academic Publishers, p.97-112.