Inicios de una gramática para el español en ALEP, un formalismo de unificación

Toni Badia
(Universitat Pompeu Fabra, Barcelona)

El objetivo de la presente comunicación es presentar la gramática del español que se está desarrollando en algunos programas LRE de la Comunidad Europea. Desde el final de la etapa central de EUROTRA (proyecto de traducción automática auspiciado por la Comunidad Europea) en diciembre de 1990, los responsables de la Comisión de las áreas de ingeniería lingüística y tratamiento de la información están impulsando el desarrollo de un prototipo de procesamiento en aras de la creación, en un plazo medio, de estándares gramaticales y léxicos para el procesamiento del lenguaje natural. Como resultado de estos esfuerzos surgió la definición del formalismo ET6 (Alshawi et al, 1991) y de una primera implementación experimental suya, ALEP. Es en esta implementación experimental que se han llevado a cabo los primeros experimentos de reutilización de recursos gramaticales existentes y de escritura de gramáticas (Arnold et al, 1992, 1993).

Uno de estos experimentos consiste en la elaboración de gramáticas iniciales en algunas lenguas, entre ellas el castellano, para poner a prueba las especificaciones formales del prototipo. Es, pues, esta gramática inicial (en curso de elaboración todavía) la que presentamos aquí. La comunicación se estructura en dos partes diferenciadas. En la primera, se describen brevemente las características expresivas básicas del formalismo; y, en la segunda, se exponen algunos de los aspectos centrales de la gramática del español que se está desarrollando.

1. Breve descripción del formalismo

Se trata de un formalismo que intenta compaginar los avances más recientes en el tratamiento de la información lingüística con criterios basados en la eficiencia en la computación. Así pues, por un lado, es un sistema que permite la definición de tipos, estructurados jerárquicamente mediante herencia simple, mientras que, por el otro, mantiene un esqueleto de reglas libres de contexto, el uso del sistema de tipos está restringido al tiempo de compilación (y no actúa en el de procesamiento), no permite rasgos cuyo valor sea un conjunto ni, consecuentemente, ninguna operación sobre conjuntos, no se puede formular la distinción entre los esquemas de dominio inmediato y los de precedencia lineal, etc.

El primer paso en el proceso de elaboración de una gramática en este formalismo es la definición del sistema de tipos. Para ello el lingüista dispone de los tipos primitivos incorporados en el sistema y de los mecanismos de estructuración de la jerarquía de los tipos. Los tipos primitivos, a partir de los cuales se pueden diseñar tipos complejos, son los siguientes:

(1) - &atom (para átomos),
- &list (para listas),
- &term (para términos), y
- &boolean (para valores atómicos con combinaciones booleanas).

Mediante estos tipos se pueden caracterizar valores correspondientes a atributos como los siguientes:

(2) lex => &atom _ lex => casa
subcat => &list &atom o/sn/sp/sa/sadv subcat => [sn, sp]

En el primer caso se indica que el valor propio del atributo "lex" corresponde a un átomo cualquiera (de ahí la variable "_"), un ejemplo de lo cual puede ser la palabra *casa*; por otra parte, en el segundo caso se especifica que el valor del atributo "subcat" es una lista de átomos a escoger entre los indicados, ejemplificada aquí mediante la lista "[sn,sp]".

La definición de los tipos booleanos implica la indicación del conjunto de valores que pueden tomar. Así, la definición de (3) indica que el atributo "concordancia" tiene como posibles valores cualquier combinación booleana de los valores de los dos conjuntos, como por ejemplo los de (4), donde "A ; B" indica disyunción, "A & B" conjunción, y "~A" negación.

(3) concordancia => &boolean {sing/pl, 1/2/3}

(4) concordancia => sing ; 3
concordancia => ~3
concordancia => ~(sing & 3)

El tipo "&term" es usado para indicar que el atributo en cuestión puede tener como valor cualquier término Prolog. Así, un atributo puede tomar una variable (indeterminada o compartida) o un término Prolog (simple o complejo):

(5) forma_logica => _
forma_logica => L
forma_logica => silla(X)
forma_logica => para_todo(X,silla(X))

En una definición de tipos, típicamente se caracterizan los diferentes atributos (y el tipo de sus valores) que corresponden a cada tipo dado. Así pues, un tipo se declara indicando cuales son los atributos que lo componen y el tipo de sus valores:

(6) <nombre_tipo> atts <nombre_atributo1> => <tipo_atributo1>,
<nombre_atributo2> => <tipo_atributo2>,

...
<nombre_atributoN> => <tipo_atributoN>.

Así un tipo "concordancia" declarado como en (7), permitiría la descripción lingüística de (8).

(7)
```
concordancia  atts    numero => &atom sing/pl,
                      persona => &atom 1/2/3.
```

(8)
```
concordancia:{numero => sing,
              persona => 3}
```

Naturalmente, este tipo complejo puede ser usado posteriormente en la definición de otros tipos. De esta forma puede empezarse a construir la jerarquía de tipos.

Además de este simple mecanismo de estructuración de tipos, el sistema permite la creación de una jerarquía de tipos mediante herencia simple. A un tipo determinado se le pueden definir subtipos, los cuales heredan la información definida en el tipo. Veamos por ejemplo el siguiente tipo:

(9)
```
head    atts    cat => &atom n/v/det/a,
                agr => agr
        subs    head_n/head_v/head_a/head_det.
```

En él se ha declarado que el tipo "head" tiene un par de atributos ("cat" y "agr") y que, además, tiene diferentes subtipos (head_n, head_v, etc.), a cada uno de los cuales pueden corresponder atributos específicos. Así, en los siguientes ejemplos se muestra que los atributos correspondientes a "head_n" y "head_v" son distintos:

(10)
```
head_n:head   vals    cat => n
              atts    nform => &atom norm/pro,
                      case => &atom nom/gen/acc/dat.
head_v:head   vals    cat => v
              atts    vform => &atom fin/infin/ger/pastp,
                      vtype => &atom main/cop/aux.
```

Cada uno de estos subtipos conlleva la especificación del supertipo al cual corresponden; además, se puede indicar (como en este caso) que al subtipo le corresponden unos valores determinados a alguno de sus atributos (en este caso, al atributo "cat"); finalmente, se indica cuáles son los atributos específicos del subtipo. Aún se podrían proponer subtipos de cada uno de ellos, de manera que se completara la jerarquía.

Una vez caracterizado el sistema de tipos el lingüista puede construir las entradas léxicas y las reglas gramaticales de su gramática. Naturalmente, tanto las unas como las otras deben ajustarse a la definición de tipos previamente establecida. Vamos a llamar

descripción lingüística (**dl**) a los tipos estructurados definidos según las reglas indicadas anteriormente.

Las entradas léxicas son simplemente descripciones lingüísticas, de complejidad variable, apareadas a un índice, como en esta de *por*, extraordinariamente simplificada:

(11) por ~

```
dl:{sintaxis:{nucleo => {cat => prep,
                         forma_prep => por},
               subcat => [ dl:{nucleo => {cat => sn}} ]}}.
```

Las reglas gramaticales tienen una estructura libre de contexto; de hecho, las categorías que relacionan son tipos complejos o descripciones lingüísticas. Así, pues, una regla del tipo de SP -> Prep SN podría tener una forma parecida a la siguiente:

(12)
```
sp1 = dl:{sintaxis:{nucleo => {cat => sp,
                               forma_prep => FP},
                               subcat => []}}
        ->
      dl:{sintaxis:{nucleo => {cat => prep,
                               forma_prep => FP},
                               subcat => [ SN ]}},
      SN @ dl:{sintaxis:{nucleo => {categoria => sn}}}.
```

En esta regla se puede observar el funcionamiento de las variables. Por una parte, "FP" unifica los valores de los dos atributos que aparecen en la regla, el de "forma_prep" de la descripción lingüística de la madre y el de la hija de categoría preposición. Por otra parte, "SN" unifica la descripción lingüística subcategorizada por la preposición con la hija de categoría sintagma nominal; como en la regla se quieren imponer restricciones adicionales a esta descripción lingüística, estas se añaden después del signo "@" a la variable (en este caso la restricción adicional es simplemente que la hija tenga la categoría de sintagma nominal).

2. Descripción de los módulos gramaticales existentes

En la gramática que presentamos se ha tratado de sacar el máximo partido del formalismo usado, con la finalidad de hacer más elegante y expresiva la gramática resultante. Esto ha supuesto, fundamentalmente, una opción a favor de la generalidad de las formulaciones gramaticales, para la cual se han usado las características del sistema de tipos con herencia simple y una distinción clara entre la información correspondiente a las entradas léxicas y la correspondiente a las reglas de reescritura.

Ciertamente, la falta de generalidad es uno de los problemas más agobiantes

presentes en las implementaciones gramaticales, en muchos de sus aspectos y, especialmente, en las reglas. Una gramática que pretenda generar las distintas formas que reviste la subcategorización del verbo español a partir de reglas distintas, necesitará un número forzosamente alto de reglas; por ejemplo, en la gramática del verbo español de Eurotra (descrita en una comunicación del anterior congreso de la SEPLN: Badia & Carulla, 1993) se mostraban más de veinte esquemas distintos de subcategorización que, en aquel formalismo, tenían que ser expresados en reglas distintas; nótese además, que a esta cifra cabría añadir las reglas adicionales necesarias para tratar la pasivización y la impersonalización de muchos de los esquemas de subcategorización listados.

A este tipo de planteamiento podemos oponer consideraciones tanto de tipo teórico como de tipo práctico. En primer lugar, en la mayoría de los casos la relación del verbo con sus complementos es siempre la misma, adopten estos la forma morfosintáctica que adopten; por lo tanto, parece razonable buscar un planteamiento de la subcategorización verbal que ponga de manifiesto esta generalización (recuérdese, por ejemplo, el avance que supuso el tratamiento de la subcategorización en las gramáticas de unificación propuesto en Shieber (1986) para PATR, en relación con el que era estándar en, por ejemplo, GPSG (Gazdar et al, 1985)). Desde este punto de vista, pues, es una característica del verbo la determinación de las características morfosintácticas de sus complementos, mientras que el mecanismo general de subcategorización es único para todos los verbos y todos sus complementos; es pues razonable distribuir estas dos informaciones en dos módulos distintos: la primera en las entradas léxicas de los verbos, y la segunda en las reglas generales de la gramática (o en sus principios generales, como por ejemplo en HPSG: Pollard & Sag, 1989).

Asimismo, un planteamiento de este tipo facilita enormemente la legibilidad y mantenimiento del sistema, justamente a causa de su modularidad. Los fenómenos generales, ligados a la relación de subcategorización (el movimiento de la información nuclear, el cálculo de las características semánticas de la madre, etc.) podrán ser formulados de una vez por todas en la regla de subcategorización. Pero los fenómenos que dependen del verbo concreto de que se trate serán planteados en la entrada léxica correspondiente. No hay duda de que un planteamiento como este permite una mejor diferenciación entre los distintos tipos de información lingüística relevantes para la subcategorización verbal.

Por otra parte, los complementos subcategorizados pueden aparecer en distintos órdenes relativos y entremezclados con modificadores. Las siguientes oraciones ejemplifican estos hechos, sin tener en cuenta la posible inversión del sujeto o las dislocaciones y topicalizaciones de los objetos (que no son tratadas por la gramática actual).

(13) a. La niña ha dado las golosinas a los niños
b. La niña ha dado a los niños las golosinas
c. La niña ha dado en la escuela las golosinas a los niños
d. La niña ha dado las golosinas a los niños en la escuela

Aunque se entremezclen entre sí, las relaciones que se establecen entre un núcleo y sus complementos subcategorizados, por una parte, y sus modificadores, por otra, son realmente distintas. Un tratamiento general adecuado debería permitir que estas dos relaciones fueran expresadas mediante reglas distintas (en las que, por ejemplo, el cálculo de las características semánticas de la madre procedería de manera distinta) y que pudieran aplicarse independientemente del orden en que se encuentren en la cadena a analizar.

Pero aún hay más. La relación de subcategorización no es exclusiva de los verbos; hay otras categorías gramaticales que pueden subcategorizar complementos (en castellano, algunos nombres, adjetivos y preposiciones). También aquí vemos características comunes a todas las relaciones de subcategorización, independientemente de la categoría del núcleo. Sería, por lo tanto, más general un tratamiento que comportara un solo tipo de regla para todos estos casos.

Estos planteamientos en favor de la generalidad han sido los que han dirigido la implementación concreta de esta gramática. Esto ha influido tanto en el diseño del sistema de tipos como en la distribución de las tareas entre las entradas léxicas y las reglas gramaticales. En los apartados siguientes se comentarán estas opciones oportunamente.

2.1 El sistema de tipos

El diseño implementado del sistema de tipos permite generalizaciones en las reglas de la gramática; por ejemplo, la inclusión de la distinción básica entre categorías mayores y menores, así como el conjunto de rasgos con listas como valores ("comps", "subj", "spr", "mod", y "spec") permiten la implementación de las relaciones de dominio mediante unas pocas reglas muy generales.

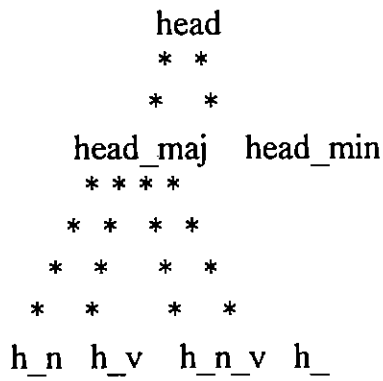
El sistema de tipos ha sido elaborado siguiendo, en gran parte, las propuestas de HPSG (que aparecen en los dos libros básicos de la teoría: Pollard & Sag, 1989; en prensa): la información sintáctica y semántica se halla reunida en un único atributo "synsem", hay diferentes listas para las distintas relaciones entre hermanas (siguiendo la propuesta del capítulo 9 de Pollard & Sag, en prensa), la organización interior de la información semántica sigue las propuestas de HPSG, etc. Aunque inspirada en HPSG, esta implementación no puede presentarse como una implementación suya, por cuanto que las posibilidades expresivas de ambos formalismos no son las mismas y, en caso de inadecuación, la solución adoptada aquí es la que mejor se adapta a las condiciones de ALEP.

En la breve presentación del sistema de tipos que sigue, conviene tener en cuanta que el nombre de los atributos y el del tipo de sus valores coincide siempre (a no ser, claro está, que se trate de tipos primitivos), con la simple finalidad de facilitar la lectura de los objetos resultantes del proceso de análisis. En lo que sigue los tipos se van a escribir en cursiva y los atributos en mayúsculas.

El tipo básico de las descripciones lingüísticas completas (léxicas o sintagmáticas) es *sign*, con tres atributos: ORTHO, que simplemente contiene información sobre la ortografía del signo; SYNSEM, que contiene toda la información lingüística (sintáctica y semántica) del signo; y LEX, que es un rasgo con valores atómicos ("lex", "nolex") que indican si el signo es léxico o no. Veamos con un poco detalle la estructuración de la información lingüística.

La estructura del tipo *catg*, que contiene toda la información sintáctica del signo, es fundamental para los principios de generalidad que hemos expuesto anteriormente. En él la información se halla dividida entre la que es propiamente nuclear (en el atributo HEAD) y la que se relaciona con las hermanas del signo (en la serie de atributos que tienen listas como valores: COMPS, SUBJ, SPR, MOD, y SPEC).

En el esquema siguiente se muestra la jerarquía de tipos correspondiente al tipo *head*.



La información sintáctica se completa mediante la serie de rasgos que expresan las relaciones que el signo puede contraer. Siguiendo la propuesta del último capítulo de Pollard & Sag (en prensa), se han distinguido hasta cinco distintos tipos de relaciones.

En primer lugar, la lista clásica de subcategorización ("subcat" en Pollard & Sag, 1989) ha sido dividida entre SUBJ y COMPS. La primera corresponde al sujeto, mientras que la otra incluye todos los otros complementos subcategorizados (en orden de oblicuidad). La razón fundamental para mantener esta división aquí está en que, en este momento tal como son tratados en la gramática, solo los verbos pueden tener sujeto, de manera que es adecuada la distribución en dos listas: los verbos tendrán, (casi) todos, un elemento en la lista de SUBJ, mientras que las otras categorías mayores van a tener esta lista vacía. La lista de COMPS va a ser vacía para los verbos intransitivos y para los adjetivos y nombres que no subcategorizan por ningún complemento, para los demás (verbos con por lo menos dos complementos subcategorizados, adjetivos y nombres con complemento) estará llena con algún elemento.

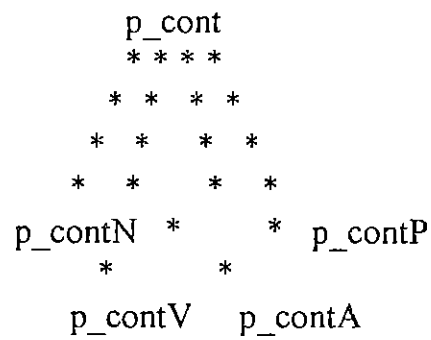
Las otras listas sirven para las siguientes caracterizaciones. Los modificadores imponen algunas condiciones sobre sus núcleos (de acuerdo con su tratamiento clásico en gramática categorial); estas son expresadas en la lista de MODS. Las categorías

menores exigen unas condiciones específicas de sus núcleos; en la gramática tal como está implementada esto afecta a los artículos y a las preposiciones fuertemente regidas; estas condiciones se expresan en la lista de SPEC. Finalmente los núcleos de los sintagmas también pueden restringir el tipo de especificador que pueden tener (por ejemplo, los nombres no contables no pueden tener un artículo indeterminado, en general un nombre contable en singular exige algún determinante, etc.); estas restricciones se especifican en la lista de SPR.

Esta estructuración del tipo *head* en subtipos y la distinción entre distintas listas para especificar distintas relaciones entre hermanas son esenciales para el objetivo planteado de conseguir reglas gramaticales lo más generales posibles. Por ejemplo, es posible pensar en una única regla de aplicación de los complementos subcategorizados no sujetos (sea cual sea la categoría del núcleo) gracias a que existe una manera precisa de especificar esta categoría en la jerarquía de tipos (o sea, *h_maj*) y un rasgo cuyo valor es una lista que no está vacía, única y exclusivamente en los casos en que aparecen estos complementos. Esto se puede ver claramente más abajo en la regla "comps". La otras reglas tienen restringida su aplicación de manera parecida.

La información semántica está contenida en el tipo *cont*, que tiene dos atributos: P_CONT y M_LIST. Conviene notar que esta distinción está motivada únicamente por la falta en el sistema de conjuntos como posibles valores de los atributos. No siendo esto posible, la única manera de poder incorporar el valor semántico de los modificadores de un signo es irlos añadiendo en una lista, que obviamente no puede unificar con la del signo sin el modificador. Por lo tanto, hay que separar en dos atributos distintos la información semántica que puede unificar de madre a hija (en el atributo P_CONT, es decir "proper content") y la que no puede hacerlo (en M_LIST, "modifier list").

El tipo *p_cont* está también estructurado jerárquicamente, como se muestra en el esquema siguiente, donde las iniciales en mayúsculas se deben leer como "nominal", "verbal", "adjetival" y "preposicional", respectivamente:



2.2 El tratamiento léxico de los fenómenos

De acuerdo con los principios mencionados anteriormente, muchos fenómenos han sido tratados en esta gramática de manera léxica. En esto sigue los pasos generales

que se han estado dando estos últimos años en la familia de teorías y formalismos de unificación. Por esto, en este apartado se van a mostrar algunas entradas léxicas características de la gramática desarrollada y se van a comentar los aspectos más relevantes de las mismas.

2.2.1 Artículos

Los artículos son tratados como categorías menores. Por lo tanto no son considerados los núcleos del sintagma nominal. No obstante, pueden imponer algunas restricciones sobre el nombre, que quedan especificadas en el valor del atributo SPEC. Por otra parte, naturalmente, la regla que construye sintagmas nominales a partir de un nombre y su artículo exige que se cumplan las condiciones que el nombre impone a sus determinantes (especificadas como valor del atributo SPR en las entradas léxicas nominales).

Al no ser el artículo el núcleo del sintagma, sus rasgos distintivos pueden aparecer de forma natural en la especificación del tipo de nombre con el cual pueden combinar (es decir, en el valor del atributo SPEC): así pues, tanto la propiedad de definitud, como las de concordancia vienen expresadas únicamente en este valor. Esto puede verse claramente en la entrada léxica siguiente, para el artículo *el*:

```
el ~

sign:{ortho => ortho:{string => [el|Rest], rest => Rest},
    lex => lex,
    synsem => synsem:{catg => catg:{head => h_min:{cat => min,
                                                   min_type => det},
                                    spec =>
[ sign:{synsem =>
          synsem:{catg => catg:{head => h_n},
                  cont => cont:{p_cont =>
                                  p_contN:{ind => ind:{gen => masc,
                                                       num => sing},
                                           restr => restr:{def => def}}}}}} ] },
                    cont => _}}}.
```

2.2.2 Nombres

En este momento los nombres implementados pertenecen todos al grupo de nombres simples, no predicativos y contables. En ellos, por lo tanto, no se encuentran elementos subcategorizadores. La información que contienen es muy elemental: en el rasgo SPR se determina el tipo de determinante que tienen que tener (si lo tienen que

tener) y en la información semántica se incluye especificación sobre los rasgos de concordancia y sobre unos pocos rasgos de semántica léxica (+/-animado, +/-humano +/-abstracto, +/- dinámico +/- contable).

En la siguiente entrada léxica de *niña* puede verse fácilmente como queda estructurada toda esta información:

niña ~

```
sign:{ortho => ortho:{string => [niña|Rest], rest => Rest},
    lex => lex,
    synsem => synsem:{catg => catg:{head => h_n:{nform => norm},
                                    spr =>
[sign:{synsem => synsem:{catg => catg:{head =>
                                              h_min:{min_type => det}}}}} ]},
          cont => cont:{p_cont => p_contN:{ind => @ X
                                                ind:{gen => fem,
                                                     num => sing,
                                                     prs => '3'},
                                    restr => restr:{pred =>
                                                pred:{rel => niña},
                                                     semf => hum:{hum => hum,
                                                                  abs => noabs},
                                                inst => @ X ind}}}}}}.
```

2.2.3 Verbos

Los verbos implementados ejemplifican distintos fenómenos:

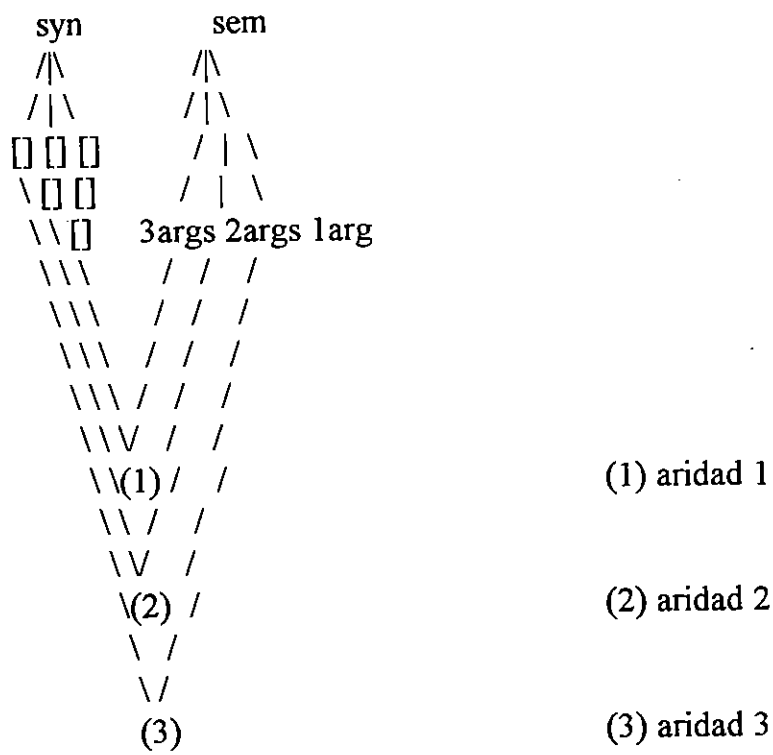
- la subcategorización de sintagmas nominales y preposicionales, y de sintagmas verbales en infinitivo y participio,
- los verbos auxiliares
- los verbos de elevación (*parecer*) y de control (*prometer*),
- la pasiva morfológica (con y sin complemente agente), y
- variaciones en el orden de los complementos (postverbales).

En todas las entradas léxicas se establece una distinción entre el sujeto (en el rasgo SUBJ) y el resto de los complementos subcategorizados por el verbo (en el rasgo COMPS).

Todos los fenómenos señalados anteriormente son tratados léxicamente, de manera que las diferentes maneras que tienen los verbos de relacionarse con sus complementos vienen determinadas por la especificación que de estas relaciones se hace en las entradas léxicas. Así, por ejemplo, en la entrada léxica de *promete* (que exige un sintagma verbal en infinitivo) se expresa el hecho de que el contenido del sujeto

corresponde al argumento primero de *prometer* y, también, al del verbo en infinitivo que actúa de complemento de *prometer*.

En todas las entradas léxicas se establece una distinción entre el esquema de subcategorización del verbo (que se expresa mediante los atributos SUBJ y COMPS) y su estructura argumental (contenida en los rasgos semánticos). La correspondencia natural entre estos dos tipos de información no puede ser expresada completamente en el sistema a causa de la falta de herencia múltiple en el sistema de tipos. En caso de haber tenido herencia múltiple, habríamos podido establecer que, a no ser que se establezca de otra manera, la valencia de un predicado establecida en los rasgos semánticos está en correspondencia con la aridad que se determina en el conjunto de rasgos sintácticos; evidentemente, en tal caso, esta correspondencia se podría haber establecido en el sistema de tipos, y no en el léxico, como debe hacerse ahora. Si esto hubiera sido posible, podríamos haber definido subtipos de predicados de acuerdo con su valencia, tanto sintáctica como semántica. Una representación esquemática de esta relación podría ser la siguiente:



En la entrada léxica siguiente de la forma verbal *piensa* se pueden ver algunas de las características fundamentales de las entradas léxicas verbales. Básicamente la información se estructura en tres grandes grupos. Por una parte, la información sintáctica de tipo nuclear, en la que se especifica, por ejemplo, el tipo de verbo y la forma verbal concreta de que se trata; esta es la información que se hereda de hijas a madres hasta llegar a la proyección máxima del núcleo. En segundo lugar, hallamos la información de subcategorización, dividida entre los rasgos SUBJ y COMPS; en las listas que tienen ambos como valores se especifican las características que la forma verbal *piensa* impone a sus complementos; en este caso concreto, ambas listas tienen un solo elemento, pero

resulta claro que los verbos que tienen un objeto directo y un objeto indirecto (o un complemento preposicional) tendrán dos elementos en la lista del rasgo COMPS; naturalmente las otras listas están todas vacías. Finalmente, el tercer apartado informativo corresponde a la semántica, en el rasgo CONT; aquí se consigna información relativa a la diátesis, al tiempo y el aspecto, y a la estructura predicativa; naturalmente, la entrada léxica contempla la unificación del contenido de los distintos complementos del verbo (en las listas de subcategorización) con el valor de los atributos correspondientes en su estructura argumental (el contenido del sujeto unifica con el argumento primero de la estructura argumental, y el del objeto preposicional, con el argumento segundo).

```
piensa ~

sign:{ortho => ortho:{string => [piensa|Rest], rest => Rest},
      lex => lex,
      synsem => synsem:{catg => catg:{head =>  h_v:{vform => fin,
                                                    vtype => main},
                                      mod => [ ],
                                      subj =>
[sign:{lex => nolex,
      synsem => synsem:{catg => catg:{head => h_n},
                                cont => @ SUBJ
                                cont:{p_cont =>
                                      p_contN:{ind => ind:{num => sing,
                                                           prs => '3'},
                                               restr =>
                                               restr:{semf =>
                                                      semf:{anim => anim}}}}}}],
                               comps =>
[sign:{lex => nolex,
       synsem =>
       synsem:{catg => catg:{head => h_n:{pform => en}},
                    cont => OBJ }}} ],
                               spr => [ ],
                               mod => [ ],
                               spec => [ ]},
          cont => cont:{p_cont =>
                               p_contV:{dia => activ,
                                        tense => pres,
                                        aspect => noperf,
                                        pred => pred3:{rel => pensar,
                                                       arg1 => SUBJ,
                                                       arg2 => OBJ }}}}}.
```

2.2.4 Adverbios

Los adverbios ejemplifican perfectamente el tratamiento dado a los modificadores. A nivel sintáctico se caracterizan por ser de categoría -n & -v y por no subcategorizar ningún complemento (en esto se diferencian de las preposiciones semánticamente llenas, como veremos más abajo). Como modificadores, los adverbios imponen algunas condiciones a los núcleos de los sintagmas que modifican; por ejemplo, los adverbios *ahora* y *aquí* exigen de su núcleo que sea verbal. Las restricciones de este tipo se encuentran especificadas en el rasgo MOD, que comparten los adverbios con todas las categorías que tienen una relación de modificación con sus núcleos (como, por ejemplo, algunas preposiciones).

Veámoslo en la entrada léxica de *ahora*.

```
ahora ~

sign:{ortho => ortho:{string => [ahora|Rest], rest => Rest},
      lex => lex,
      synsem => synsem:{catg => catg:{head => h_:{vadv => time},
                                      mod =>
[ sign:{lex => nolex,
      synsem => synsem:{catg =>
                  ( catg:{head => h_v:{vtype => main}};
                    catg:{head => h_v:{vtype => aux}, subj => [ ]}),
                    cont => cont:{p_cont =>
                                    p_contV:{tense => pres}}}}}],
                        subj => [ ],
                        comps => [ ]},
                        cont => cont:{p_cont =>
                                        p_contPtemp:{r_type => time,
                                                     pred =>
                                                     pred:{rel => ahora},
                                                          tense => pres}}}}}.
```

2.2.5 Preposiciones

En el estado de implementación en que se encuentra la gramática, aparecen dos tipos de preposiciones: las que son de categoría menor y las que tienen la misma categoría que los adverbios (es decir, -n & -v). Las primeras son las llamadas a veces preposiciones fuertemente regidas y actúan generalmente como marcadoras de caso. Las segundas son preposiciones que acarrean un significado léxico determinado.

Como es natural las entradas léxicas de unas y otras tienen poco en común. La siguiente entrada léxica de la preposición *en* puede servir de ejemplo de las preposiciones de categoría menor. Estas preposiciones no tienen ningún contenido semántico; su entrada léxica, pues, contiene sólo un mínimo de información sintáctica y la indicación

del tipo de sintagma del que ellas pueden ser marcadores (en el rasgo SPEC). En este caso, simplemente se indica que este sintagma tiene que ser nominal y que llevará la indicación de que su PFORM es *en*.

```
en ~

sign:{ortho => ortho:{string => [en|Rest], rest => Rest},
      lex => lex,
      synsem => synsem:{catg => catg:{head => h_min:{min_type => marker},
                                      mod => [ ],
                                      comps => [ ],
                                      spr => [ ],
                                      subj => [ ],
                                      spec =>
[ sign:{lex => nolex,
      synsem =>
      synsem:{catg => catg:{head => h_n:{pform => en}}}}}]}}}}.
```

Por otra parte, la preposición *en* puede aparecer en contextos en los cuales es una preposición semánticamente llena. En este caso, la información que contendrá la entrada léxica será la misma que contienen las entradas de los adverbios, más la que hace referencia al complemento subcategorizado por la preposición, de manera que la diferencia más importante entre el adverbio *aquí* y la preposición locativa *en* será el rasgo COMPS, que en el primero tiene como valor la lista vacía, y en el segundo se especifican las características que tiene que tener el complemento (por ejemplo, que tiene que ser nominal y que no puede ser animado). Veamos esta entrada léxica como ejemplo:

```
en ~

sign:{ortho => ortho:{string => [en|Rest], rest => Rest},
      lex => lex,
      synsem => synsem:{catg => catg:{head => h_:{vadv => loc},
                                      mod =>
[ sign:{lex => nolex,
      synsem => synsem:{catg => catg:{head => h_v:{vtype => main}} }}} ],
                                     comps =>
[ sign:{lex => nolex,
      synsem => synsem:{catg => catg:{head => h_n},
                        cont => @ ARG
                        cont:{p_cont =>
                              p_contN:{restr =>
                                       restr:{semf =>
                                              semf:{anim => noanim}}}}}}]},
```

```
                    cont => cont:{p_cont =>
    p_contP:{r_type => space,
             pred => pred1:{rel => en, arg1 => ARG}  }}}}}.
```

2.3 Las reglas básicas de reescritura

En su estado actual, la gramática contiene seis reglas, que realizan operaciones muy generales. De hecho hay una regla para cada relación entre hermanas. A ellas se añade una regla nueva para poder tratar por un igual todos los complementos subcategorizados por un núcleo (es, pues, una regla que expande los signos léxicos en no léxicos; Shieber, 1986).

Son los siguientes:

a/ para sujetos:

```
subj = sign:{ortho => ortho:{string => A, rest => C},
             lex => nolex,
             synsem =>
             synsem:{catg => catg:{head => HEAD,
                                   comps => [ ],
                                   subj => [ ]},
                                   cont => CONTENT}}}
->
[  @ SUBJ sign:{ortho => ortho:{string => A, rest => B},
                lex => nolex,
                synsem => synsem:{catg => catg:{subj => [ ],
                                                spec => [ ],
                                                comps => [ ],
                                                mod => [ ],
                                                spr => [ ] }}}},
   sign:{ortho => ortho:{string => B, rest => C},
         lex => nolex,
         synsem => synsem:{catg => catg:{head => HEAD,
                                         comps => [ ],
                                         subj => [ SUBJ ]},
                           cont => CONTENT}}} ] head 1.
```

Esta es la regla que cancela el sujeto de un sintagma, especificado en la lista de SUBJ. Por el momento solo se trata el sujeto en posición preverbal.

b/ para complementos:

```
comps = sign:{ortho => ortho:{string => A, rest => C},
              lex => nolex,
              synsem => synsem:{catg => catg:{head => @ HEAD h_maj,
                                              subj => SUBJ,
                                              spec => [ ],
                                              spr => SPR,
                                              mod => MOD,
                                              comps => REST },
                                cont => CONTENT}}}
->
[   sign:{ortho => ortho:{string => A, rest => B}, lex => nolex,
          synsem => synsem:{catg => catg:{head => HEAD,
                                          spec => [ ],
                                          spr => SPR,
                                          mod => MOD,
                                          subj => SUBJ,
                                          comps => [COMPS|REST]},
                            cont => CONTENT}}} ,
@ COMPS sign:{ortho => ortho:{string => B, rest => C}, lex => nolex,
                synsem => synsem:{catg => catg:{spec => [ ],
                                                comps => [ ],
                                                mod => [ ],
                                                spr => [ ] }}}}] head 1.
```

Esta es la regla que construye SSVV, SSPP, etc. Es una regla binaria que cancela solo un complemento en cada aplicación. Para poder usar esta misma regla para el primero y para el resto de los complementos de la lista de COMPS el signo léxico se tiene que convertir en no léxico mediante una aplicación de la regla "xBAR".

Esta regla toma el siguiente constituyente de la cadena de entrada y lo unifica con el primer elemento de la lista de COMPS del núcleo. Todos sus otros rasgos (HEAD, SUBJ, SPR, MOD, y CONTENT) simplemente se elevan a la madre.

c/ para especificadores:

```
det = sign:{ortho => ortho:{string => A, rest => C},
            lex => nolex,
            synsem => synsem:{catg => catg:{head => HEAD,
                                            spr => [ ],
```

```
                                                    subj => [ ],
                                                    spec => [ ],
                                                    comps => [ ],
                                                    mod => [ ]},
                                    cont => CONTENT}}}
    ->
[   sign:{ortho => ortho:{string => A, rest => B}, lex => lex,
          synsem => @ SPR
          synsem:{catg => catg:{head => h_min:{min_type => det},
                                        spec =>
           [ sign:{synsem => synsem:{catg => catg:{head => HEAD },
                                            cont => CONTENT}}} ]}}}},
  sign:{ortho => ortho:{string => B, rest => C}, lex => lex,
        synsem => synsem:{catg => catg:{head => @ HEAD h_n,
                                                    subj => [ ],
                                                    spec => [ ],
                                                    comps => [ ],
                                                    mod => [ ],
                                                    spr =>
        [ sign:{synsem => SPR } ] },
                cont => @ CONTENT
                cont:{p_cont => p_contN:{c_type => nominal}}}}} ] head 2.
```

Esta regla trata los especificadores. Es una característica de los especificadores que imponen condiciones a los contituyentes con los que se combinan, mientras que estos, a su vez, también restringen el tipo de los especificadores con que pueden ir. Esta doble interacción se ha resuelto en esta gramática de la siguiente manera (siguiendo la propuesta de Pollard & Sag, en prensa:cap.9): los especificadores no son el núcleo de los constituyentes; las restricciones que ellos les imponen están especificadas en la lista de SPEC; finalmente, las condiciones del núcleo sobre sus especificadores están expresadas en la lista de SPR. Todas estas condiciones están expresadas en la regla "det".

d/ para marcadores:

```
pMIN = sign:{ortho => ortho:{string => A, rest => C},
             lex => nolex,
             synsem => synsem:{catg => catg:{head => HEAD,
                                                    subj => [ ],
                                                    spec => [ ],
                                                    comps => [ ],
                                                    spr => [ ],
                                                    mod => [ ]},
                                    cont => CONTENT}}}
```

```
->
[ sign:{ortho => ortho:{string => A, rest => B}, lex => lex,
        synsem =>
        synsem:{catg => catg:{head => h_min:{min_type => marker},
                              spec => [ SPEC ] }}}} ,
 @ SPEC sign:{ortho => ortho:{string => B, rest => C},          lex => nolex,
              synsem => synsem:{catg => catg:{head => @ HEAD h_n,
                                              subj => [ ],
                                              spec => [ ],
                                              comps => [ ],
                                              mod => [ ],
                                              spr => [ ] },
                                cont => @ CONTENT
                                cont:{p_cont =>
                    p_contN:{c_type => nominal}}}}} ] head 2.
```

Esta regla trata la relación entre un marcador y el sintagma que acompaña. Difiere de la regla "det" en que en este caso es solo el marcador el que impone restricciones (expresadas en SPEC) al sintagma que acompaña.

e/ para modificadores:

```
mod = sign:{ortho => ortho:{string => A, rest => C},        lex => nolex,
            synsem => synsem:{catg => catg:{head => HEAD,
                                            spec => [ ],
                                            spr => SPR,
                                            subj => SUBJ,
                                            comps => COMP},
                              cont => cont:{ p_cont => CONTENT,
                                             m_list => [CONT_MOD | REST] }}}}
->
 [ @ MODS sign:{ortho => ortho:{string => A, rest => B}, lex => nolex,
                synsem => synsem:{catg => catg:{head => HEAD,
                                                spec => [ ],
                                                spr => SPR,
                                                subj => SUBJ,
                                                comps => COMP},
                                  cont => cont:{p_cont => CONTENT,
                                                m_list => REST }}}} ,
   sign:{ortho => ortho:{string => B, rest => C}, lex => nolex,
         synsem => synsem:{catg => catg:{subj => [ ],
                                         spec => [ ],
```

```
                                comps => [ ],
                                mod => [ MODS ],
                                spr => [ ] },
                cont => CONT_MOD }}} ] head 1.
```

Esta regla permite la construcción de una estructura formada por un núcleo con un complemento suyo no subcategorizado y semánticamente lleno (es decir, con un adjunto o modificador). Tal como está escrita esta regla permite solo modificadores posteriores a su núcleo. Una regla similar podría ser escrita para los modificadores prenucleares (nótese que esta duplicación será innecesaria en el momento en que se implemente el formato de reglas ID/LP en el formalismo).

Por supuesto la regla solo admite los modificadores que imponen las restricciones adecuadas a sus núcleos (expresadas en la lista de MODS). Puesto que la mayoría de los adjuntos no son léxicos, se exige que no lo sean; por lo tanto, adverbios simples (como *ahora*) deberán ser convertidos en signos sintagmáticos mediante la regla "xBAR".

El contenido del modificador es puesto en este momento en una lista de contenidos de modificadores, que es independiente del contenido propiamente dicho del signo (es decir, sin sus modificadores); esta distinción es necesaria para poder elevar el contenido del signo del núcleo a la madre.

f/ regla "xBAR":

```
xBAR = sign:{ortho => ortho:{string => A, rest => B}, lex => nolex,
                synsem => SYNSEM }
       ->
  [sign:{ortho => ortho:{string => A, rest => B}, lex => lex,
         synsem => @ SYNSEM
         synsem:{catg => catg:{head => h_maj:{cat => maj}}}}}] head 1.
```

Esta regla permite la conversión de un signo léxico en sintagmático. Como ya hemos mencionado es necesario para mantener la generalidad de las reglas.

3. Conclusión

En esta comunicación hemos presentado los primeros pasos de la implementación de una gramática del español en un nuevo formalismo, ALEP. El sistema permite un gran nivel de generalidad en los planteamientos, al disponer de un sistema de tipos estructurados por herencia simple y de una clara distinción entre la información consignada en las entradas léxicas y el tratamiento de la misma en las reglas de

reescritura de frase. Se ha mostrado el diseño básico del sistema de tipos, así como algunos de los fenómenos tratados en esta primera etapa.

NOTA: Parte del trabajo presentado en esta comunicación ha podido ser realizado gracias al proyecto ET10/52 de la CE, a través de la Fundació Bosch Gimpera, de la Universitat de Barcelona.

Referencias bibliográficas

Alshawi H, Arnold D, Backofen R, Carter D, Lindop J, Netter K, Pulman S, Tsujii J & Uszkoreit H, 1991, *Eurotra ET6/1: Rule Formalism and Virtual Machine Design Study* (Final Report). CEC.

Arnold D, Badia T, van Genabith J, Kohl D, Markantonatou S, Pulman S, Sadler L & Schmidt P, 1992, *ET10/52 Final Report: Reusability of Grammatical Resources and Grammar Development in ALEP*. CEC.

Arnold D, Badia T, van Genabith J, Markantonatou S, Momma S, Sadler L & Schmidt P, 1993, Experiments in Reusability of Grammatical Resources, en *EACL 1993*, Utrecht.

Badia T & Carulla M, 1993, El tratamiento de la subcategorización verbal en las gramáticas de Eurotra (1987-92), en *Boletín de la SEPLN*.

Shieber S, 1986, *Introduction to Unification-Based Grammatical Formalisms*. CSLI, Stanford, Ca, EUA.

Gazdar G et al, 1985, *Generalized Phrase Structure Grammar*. Blackwell, Oxford, UK.

Pollard & Sag, 1987, *Information-Based Syntax and Semantics, I*. CSLI, Stanford, Ca, EUA.

Pollard & Sag, en prensa, *Head-driven Phrase Structure Grammar*. CSLI, Stanford, Ca, EUA.