Documentació de les modificacions al sistema de traducció automàtica Apertium per al projecte *Traducció Automàtica de Codi Obert per al Català*

Grup Transducens
Universitat d'Alacant

22 de desembre de 2006

Índex

Capítol 1

Mòdul de transferència estructural

[El contingut d'aquest capítol modifica la secció 3.4 de la documentació d'Apertium, pàg. 60.]

1.1 Introducció

La necessitat d'una nova arquitectura de transferència per a Apertium apareix amb el requeriment de la implementació de traductors entre llengües menys emparentades que les que fins ara s'han tractat.

L'arquitectura de transferència d'Apertium es basa en la manipulació automàtica de determinats patrons d'aparició de paraules mitjançant regles definides per l'usuari. A aquest model el denominem *transferència sintàctica superficial*. Una visió esquemàtica del sistema ens mostra una estructura que considera dos nivells des del punt de vista de la naturalesa de les dades: un nivell bàsic que denominem *nivell lèxic*, que s'ocupa de les paraules i de les operacions de consulta i modificació de les seues característiques (lema i etiquetes), a més de la traducció de lemes individuals mitjançant consultes al diccionari bilingüe; i un altre nivell que denominem *nivell de patrons de paraules*, que s'encarrega de fer, quan convé, reordenaments amb les paraules que constitueixen aquests patrons, així com canvis en les característiques de les paraules que depenguen de cada patró concret que haja sigut detectat. Tot aquest processament de detecció i manipulació lèxica i de patrons es du a terme en una única passada.

Com a contrast, la nova arquitectura de transferència, que anomenarem Apertium 2, es defineix com un sistema de transferència lèxica avançada en tres nivells i tres passades. Els dos primers nivells, el nivell lèxic i el de patrons, són homòlegs als del transfer d'Apertium. El nou nivell que apareix és un nivell de *patrons de patrons* de paraules. L'objectiu de la existència d'aquest nou nivell de processament és permetre la manipulació i la relació de patrons de patrons de paraules de manera similar a com es tracten les paraules en els patrons de paraules d'Apertium.

Amb aquesta nova identificació de nivells es pretén que hi haja un tractament més adequat de totes les transformacions que es requereixen per a traduir d'una llengua a una altra. Per això s'ha de posar de manifest que la identifica-

ció de patrons de paraules en Apertium no ha de coincidir amb la identificació de patrons en Apertium 2: es pretén que els patrons en Apertium 2 tinguen un *esperit* de sintagmes que no tenien en Apertium. Per aquesta raó utilitzarem el terme *segment* (*chunk*, en anglès), per a referir-nos a les seqüències de paraules d'Apertium 2.

L'arquitectura de transferència d'Apertium 2 s'organitza en tres passades. Seguint el model de processament d'Apertium, aquestes tres passades són realitzades per tres mòduls (programes) diferents:

- `transfer` (intrachunk): identifica els segments, realitza la traducció paraula per paraula, així com certes operacions de reordenament i propagació d'informació morfosintàctica dins del segment (per exemple, per a establir la concordança). A més, crea els segments perquè siguen tractats pel mòdul següent.

- `interchunk`: aquest mòdul rep els segments construïts per l'`intrachunk` i permet de reordenar els segments, modificar la "informació sintàctica" de cada segment i, finalment, imprimir els segments en l'ordre nou i amb les característiques noves en l'eixida.

- `postchunk`: aquest mòdul rep els segments modificats per l'interchunk i realitza tasques finals de modificació de les paraules contingudes en cada segment i d'impressió del text contingut en els segments en el format que accepta el generador.

L'especificació concreta de les tasques de cada mòdul es troba en les seccions corresponents.

1.2 Format dels segments

La comunicació entre els mòduls `intrachunk` i `interchunk`, al igual que la comunicació entre `interchunk` i `postchunk`, es du a terme mitjançant seqüències de segments. Definim C com una *seqüència de segments*, que té la forma:

$$C = b_0 c_1 b_1 c_2 b_2 \dots b_{k-1} c_k b_k$$

On cada b_i és un *superblanc*, i cada c és un *segment*. Un segment c es defineix com a una cadena $\hat{}\, F \{ W \} \$$ que conté la següent informació:

- F és el *pseudolema del segment*, i és una cadena que té la forma fE, on f és el *pseudolema* del segment, i $E = e_1 e_2 \dots$ es una seqüència de etiquetes morfològiques denominades *etiquetes del segment*. La modificació d'aquestes etiquetes provocarà la modificació de la informació morfològica de les paraules dins del segment que estiguen adscrites a aquests paràmetres.

- $W = b_0 w_1 b_1 w_2 b_2 \dots w_k b_k$ és la seqüència de paraules w_i enviada pel transfer, amb els *superblancs* b_i intercalats. Aquestes paraules es troben en el mateix format d'Apertium, és a dir, una paraula individual $w_i = \hat{}\, l_i E_i \$$ conté lema i etiquetes, algunes de les quals són *referències a les etiquetes* del

segment i es designen amb números naturals <1>, <2>, <3>, etc. Aquestes referències a etiquetes corresponen, en l'ordre especificat, amb les etiquetes d'E.

Un exemple d'ús d'aquest format és el següent:

```
^det_nom<SN><m><pl>{^el<det><def><2><3>$[
<a href="http://www.ua.es">]^gat<n><2><3>$}$[</a>]
```

Els caràcters { i }, quan estiguen presents en el text original, hauran de ser protegits amb una barra invertida \ al davant.

1.2.1 Tractament de majúscules i minúscules

Per a cada segment, la informació de majúscules i minúscules està relacionada amb la informació de majúscules i minúscules del pseudolema del segment, tenint en compte aquestes condicions.

- Si totes les lletres del pseudolema estan en minúscules: l'estat de majúscules i minúscules no es modifica.
- Si la primera lletra del pseudolema està en majúscules i la resta en minúscules: en la generació en el mòdul `transfer-post`, es farà que la primera lletra que resulte de tots els possibles canvis d'ordre estiga en majúscules.
- Si totes les lletres del pseudolema estan en majúscules: totes les paraules romandran en majúscules.

Una tasca requerida és que les paraules de l'interior del segment no comencen per majúscules si no es tracta de noms propis, per tal d'evitar que l'últim mòdul busque quina és la paraula que ha de perdre la majúscula, si n'hi ha. Aquesta tasca li correspon al mòdul `transfer-intrachunk` i es fa mitjançant una macro o algun mecanisme similar.

1.3 Programa `transfer` (`intrachunk`)

Aquest programa és l'hereu més directe del mòdul de transferència d'Apertium. De fet, un dels objectius del disseny d'aquest programa és que siga plenament compatible amb l'actual mòdul de transferència.

1.3.1 Entrada/eixida

- Entrada: dades amb el format d'eixida del pretransfer, és a dir, amb les cues invariants dels multimots desplaçats a una posició anterior a qualsevol etiqueta.
- Eixida: segments (chunks), perquè el mòdul següent els detecte i realitze operacions amb ells.

1.3.2 Fitxers de dades

Aquest programa usa un únic fitxer de configuració i un fitxer precompilat de detecció de patrons calculat a partir del primer, de manera idèntica a Apertium. El fitxer de patrons tindrà en el seu nom l'extensió `.t1x`. Com que `intrachunk` es qui fa la consulta del diccionari bilingüe, aquest (en forma compilada) també s'ha d'aportar al programa.

La DTD d'aquest fitxer de dades s'especifica en l'apèndix A.1.

1.3.3 Detecció de patrons

El sistema de detecció de regles d'aquest mòdul serà el mateix que en el mòdul de transferència actual d'Apertium. El programa `apertium-pretransfer` és necessari per a adaptar el format d'eixida de l'etiquetador a l'entrada que reconeix el mòdul de transferència. En posteriors versions d'Apertium no es descarta que es modifique el *desambiguador categorial* perquè faça la feina del `apertium-pretransfer`.

1.3.4 Especificació de les accions associades

Serà bàsicament similar a l'actual mòdul de transferència d'Apertium, amb les següents incorporacions:

- Especificació de l'acció per defecte en l'element arrel `<transfer>`. Aquest element té l'atribut opcional `default` que pot tenir els valors `lu` (valor per defecte) o `chunk` depenent de si es vol que el transfer funcione a la manera d'Apertium o d'Apertium 2, respectivament.

- Atribut `comment` (opcional) per als elements `<rule>`. Les regles tindran a partir d'ara un comentari que permetrà d'incloure informació que s'usarà per a documentació.

- Generació de segments (chunks): una etiqueta d'un nivell més alt que `<lu>` (*lexical unit*) que permeta de generar segments amb les característiques que han sigut especificades a la secció anterior. Aquesta etiqueta és `<chunk>` i té els següents atributs:

  - `name` (opcional): pseudolema del segment. Una cadena que permeta d'identificar el pseudolema del segment.
  - `namefrom` (opcional): pseudolema del segment, obtingut d'una variable. És obligatori especificar bé `name` o bé `namefrom`.
  - `case` (opcional): variable que serveix per a obtenir la informació de majúscules o minúscules d'una variable i aplicar-la al lema especificat bé per `name`, bé per `namefrom`.

- Cada chunk comença per una instrucció `<tags>`, que no permet especificar cap atribut, la qual, per la seua part, conté una o més instruccions individuals `<tag>`.

- Les instruccions `<tag>` no tenen atributs. Dins d'elles es pot posar qualsevol instrucció que torne com a valor una cadena: `<lit>`, `<var>`.

- Les instruccions <clip> tenen un nou atribut opcional: link-to, que serveix per a especificar una etiqueta *<valor de link-to>* en comptes de la informació que continga el <clip> en la resta d'atributs. Aquesta informació és prescindible però pot servir com a documentació de l'origen de la decisió lingüística.

Veiem amb un exemple l'ús de l'element <chunk>:

```
<out>
  <chunk name="adj-nom" case="variableCase">
    <tags>
      <tag><lit-tag v="SN"/></tag>
      <tag><clip pos="2" side="tl" part="gen"/></tag>
      <tag><clip pos="2" side="tl" part="nbr"/></tag>
    </tags>
    <lu>
      <clip pos="2" side="tl" part="lemh"/>
      <clip pos="2" side="tl" part="a_nom"/>
      <clip pos="2" side="tl" part="gen" link-to="2"/>
      <clip pos="2" side="tl" part="nbr" link-to="3"/>
    </lu>
    <b pos="1"/>
    <lu>
      <var n="adjectiu"/>
      <clip pos="1" side="tl" part="lem"/>
      <clip pos="1" side="tl" part="a_adj"/>
      <clip pos="2" side="tl" part="gen" link-to="2"/>
      <clip pos="2" side="tl" part="nbr" link-to="3"/>
    </lu>
  </chunk>
</out>
```

1.3.5 Acció per defecte

Els *superblancs* aïllats, que no són detectats per cap patró d'aquest mòdul s'escriuen en el mateix ordre en què arriben.

L'element arrel del fitxer .t1x de regles del intrachunk, <transfer>, té un atribut opcional default que pot prendre els valors

- chunk: en el cas que es vulga que el mòdul genere segments trivials amb les paraules que no puga incloure en la formació de cap segment o que no hagen sigut especificats. Aquest serà el cas del nou sistema. El nou segment es crearà mitjançant la traducció de la paraula pel diccionari bilingüe. El lema per defecte dels chunks creats per defecte és default.
- lu (valor per defecte): el funcionament actual per defecte del mòdul de transferència d'Apertium, traduint la paraula amb el diccionari bilingüe.

A continuació s'ensenya el resultat del processament per defecte d'un segment que no ha sigut detectat per cap regla del transfer-intrachunk:

```
^default{^that<cnjsub>$}$
```

1.4 Programa `interchunk`

El programa `interchunk` tracta els segments, els reordena i fa els canvis pertinents en la informació morfosintàctica. Això es fa mitjançant la detecció de patrons de segments (seqüències de segments). La sintaxi de les instruccions que s'usen per a controlar el seu funcionament són, amb poques diferències, les mateixes que usa el programa `transfer` d'Apertium o el `intrachunk` d'Apertium 2, però ha de quedar clar que s'utilitza un arxiu diferent. Els segments són tractats de forma similar a com es tracten les paraules en el transfer d'Apertium.

1.4.1 Entrada/eixida

- Entrada: segments procedents d'`intrachunk`.
- Eixida: segments possiblement reordenats i amb la informació del pseudolema del segment possiblement modificada.

1.4.2 Fitxers de dades

Aquest mòdul usa dos fitxers de dades. Un fitxer d'especificació del comportament del programa `interchunk`, amb extensió `.t2x` perquè siga sistemàtica amb el mòdul anterior, i un fitxer de patrons precalculats per a accelerar l'anàlisi de l'entrada. No s'hi inclou el fitxer binari del diccionari bilingüe perquè no s'usa.

La sintaxi del fitxer d'especificació és molt similar a la del mòdul `intrachunk`, i s'especifica en l'apèndix A.2.

1.4.3 Detecció de regles

Les regles detecten patrons que estan definits mitjançant seqüències de pseudoparaules. Aquestes pseudoparaules tenen un format basat en el format de la forma lèxica de les paraules. Des del punt de vista pràctic, aquest pseudolema és vist de manera equivalent a com es veu la forma lèxica d'una paraula en el mòdul `intrachunk` per a realitzar la detecció. D'aquesta forma, s'usarà una detecció de patrons amb atributs definits sobre les pseudoparaules i no sobre les paraules del patró original.

1.4.4 Especificació de les accions associades

Les modificacions en el joc d'instruccions d'aquest mòdul —respecte del joc d'instruccions utilitzat en `intrachunk`— són les següents:

- L'element arrel es diu `<interchunk>` i no especifica cap atribut.
- Desaparició de l'atribut `side`: aquest mòdul no fa servir diccionaris bilingües; per tant l'atribut que indica si la part que es consulta és l'origen o la meta no té sentit. Aquest atribut és usat fonamentalment en les instruccions `<out>`.

- L'etiqueta <chunk> ara s'usa sense atributs, senzillament dins de <out> per a delimitar l'eixida de segments.

- L'atribut per defecte lem fa referència al pseudolema del segment. De la mateixa manera, l'atribut per defecte tags fa referència a les etiquetes del segment. El contingut del segment queda com una cua que es pot imprimir mitjançant l'atribut per defecte chcontent.

- Tots els valors de part, excepte chname, accedeixen a la informació del lema i les etiquetes del pseudolema de cada segment.

- Al contrari que en el transfer-interchunk, no es permet d'imprimir una altra cosa que chunks en les instruccions <out> en el fitxer d'especificació del comportament d'aquest mòdul, i en cap cas paraules soltes.

1.4.5 Acció per defecte

Al igual que en el mòdul anterior, està establerta una acció per defecte que deixa passar com entren els segments que no reconeix cap regla del fitxer d'especificació d'aquest mòdul. Aquesta regla per defecte escriu exactament el que llig, tant si es tracta de chunks com de blancs.

1.5 Programa postchunk

El programa postchunk detecta segments individuals, i per a cadascun d'ells, executa unes accions especificades. Aquestes deteccions es basen en els lemes del segment, i no en patrons (no en les etiquetes); això farà que la detecció en aquest mòdul siga específica per a cada "nom" de segment.

D'altra banda, el funcionament de la detecció i processament de les regles es basa en que les referències a paràmetres es resolen immediatament després de la detecció, és a dir, les etiquetes <1>, <2>, etc. són substituïdes automàticament pel valor dels paràmetres abans de començar el processament. Les posicions (atribut pos) a les quals es fa referència amb les instruccions <clip>, etc. corresponen a l'ordre de les paraules en l'interior del segment.

També automàticament aplica la política de majúscules (veure secció 1.2.1) dels lemes de les pseudoparaules cap a les paraules que són dins del segment.

1.5.1 Entrada/eixida

- Entrada: segments procedents d'interchunk.

- Eixida: entrada vàlida del mòdul de generació morfològica d'Apertium.

1.5.2 Fitxers de dades

Aquest programa té el seu propi arxiu d'especificació, que tindrà extensió .t3x. La sintaxi està basada igualment en el transfer d'Apertium i en el transfer-interchunk d'Apertium 2.

1.5.3 Detecció de regles

La detecció dels segments és individual i es basa en el nom dels segments. Els segments no detectats reben el processament per defecte.

1.5.4 Especificació de les accions associades

Les diferències en les accions amb `transfer-interchunk` són les següents:

- Prohibició de l'escriptura de segments a l'eixida: només es poden
- Nou atribut de detecció de `<pattern-item>`, `name`, que s'usa en la part de `<pattern>` de les regles de manera aïllada, per tal de forçar la detecció dels patrons pel seu nom.
- Tampoc existeix l'atribut `side` als elements als quals és aplicable en el mòdul `transfer-interchunk`, per la mateixa raó que en `transfer--interchunk`: no s'accedeix al diccionari bilingüe.

1.5.5 Acció per defecte

En aquest mòdul l'acció per defecte és la d'escriure les paraules de dins del segment, substituint-hi les referències pels paràmetres del segment. Serà usada per la major part de segments, ja que es preveu que aquest mòdul s'utilitze per a casos excepcionals en els quals calga fer algun processament molt específic.

També s'aplica per defecte la política de majúscules (secció 1.2.1).

Els blancs externs als segments són copiats en el mateix ordre en el que són llegits en qualsevol cas, ja que la detecció de segments és unitària.

1.6 Elements dels formats de transferència

En aquesta secció s'explica el format en el qual s'escriuen les regles de transferència estructural. En l'apèndix, a les seccions A.1, A.2 i A.3 s'ofereix la descripció formal (DTD). Els arxius de regles de transferència estructural tenen dues parts ben diferenciades: una de declaració dels elements que s'utilitzaran en les regles i altra de regles pròpiament dites.

En la part de **declaració** trobem:

- Una sèrie de declaracions de *categories lèxiques* que especifiquen aquelles formes lèxiques que seran tractades com una categoria particular i que seran detectades en els patrons. Hem de destacar que el lingüista pot incloure qualsevol informació de la forma lèxica per a definir una categoria; les categories poden ser molt genèriques (p.e. tots els noms) o molt específiques (p.e. només aquells determinants que són demostratius femenins i plurals).
- Una sèrie de declaracions dels *atributs* que volem detectar en les formes lèxiques (com el *gènere*, el *nombre*, la *persona* o el *temps*), per a realitzar amb ells les operacions de transformació pertinents i enviar la informació resultant a la sortida de les regles. En la declaració dels atributs s'inclou el nom de l'atribut i els possibles valors que pot prendre aquest atribut en

una forma lèxica (en general es corresponen amb els símbols morfològics que la caracteritzen): per exemple, l'atribut de *nombre* pot prendre els valors de *singular*, *plural*, *singular-plural* (per a formes lèxiques invariables, com *crisis* en espanyol) i *nombre per determinar* (per a formes lèxiques de la LM amb distinció *singular–plural* el nombre de la qual no es pot determinar en la traducció perquè la forma lèxica de la LO siga invariable en gènere). Si dintre de la regla, fora del patró, s'ha de fer referència a alguna de les categories lèxiques definides en el punt anterior (per a sotmetre-les a accions o comprovacions), haurien de definir-se també atributs per a les mateixes.

- Una sèrie de declaracions de variables *globals*, que s'usen per a transferir valors d'atributs actius dintre de cada regla o d'una regla a les quals s'apliquen posteriorment.

- Una secció de *definició de llistes de cadenes*, generalment llistes de lemes, que seran utilitzats per a buscar en elles un element que correspongui per a realitzar una transformació determinada.

- Una sèrie de declaracions de *macroinstruccions*; les macroinstruccions contenen seqüències d'instruccions d'operacions freqüents i poden incloure's en diverses regles (per exemple, una macroinstrucció per a assegurar la concordança de gènere i nombre entre dues formes lèxiques d'un mateix patró).

En **les regles de transferència estructural** trobem:

- La definició del patró que serà detectat, especificat com una seqüència de categories lèxiques tal com s'han definit en la part de declaració. Cap ressaltar que, si la seqüència de formes lèxiques concorda amb dues regles diferents, primer, preval la més llarga i, segon, per a regles de la mateixa longitud, preval la regla definida en primer lloc.

- La part de processament de les regles, en la qual s'especifiquen les accions a realitzar sobre les FLLO i es construeix el patró en la LM.

A continuació s'expliquen detalladament les característiques de cadascun dels elements utilitzats.

1.6.1 Element `<transfer>`

Només transfer/intrachunk

És l'element arrel del mòdul transfer/intrachunk i conté tots els altres elements de l'arxiu de regles de transferència estructural d'aquest mòdul.

Té un atribut, `default`, que pot prendre els valors `lu` —que indica que les paraules per defecte han de deixar-se intactes, i que funcionarà com un transfer d'Apertium 1.0— o `chunk` —que indica que es crearà un segment per defecte que encapsule la paraula no reconeguda perquè puga ser tractada pels següents mòduls. El valor per defecte és `lu`.

1.6.2 Element <interchunk>

Només interchunk

És l'element arrel del mòdul interchunk i conté tots els altres elements de l'arxiu de regles de transferència estructural d'aquest mòdul.

1.6.3 Element <postchunk>

Només postchunk

És l'element arrel del mòdul interchunk i conté tots els altres elements de l'arxiu de regles de transferència estructural d'aquest mòdul.

1.6.4 Element de secció de definició de categories <section-def-cats>

En aquesta secció es defineixen les categories lèxiques que s'usaran per a crear els patrons utilitzats en les regles. Cada definició es realitza amb un <**def-cat**>.

1.6.5 Element de definició de categories <def-cat>

Cada definició de categoria té un nom `n` obligatori com per exemple `det`, `adv`, `prep`, etc. i una llista de categories (<**cat-item**>) que la defineixen. El nom de la categoria no pot contenir accents.

1.6.6 Element de categoria <cat-item>

Aquest element té dos usos diferenciats segons en el mòdul que s'use.

Ús en transfer/intrachunk i en interchunk

Mitjançant aquest element es defineixen les categories lèxiques que es faran servir en els patrons, és a dir, que es vol detectar en el text origen. Aquestes categories es defineixen a partir d'una subseqüència de les etiquetes que donen tant l'analitzador morfològic com el desambiguador. Cal tenir en compte que al llarg dels diferents mòduls de processament lingüístic s'utilitzen diferents categoritzacions lèxiques: en els diccionaris morfològics, els lemes s'acompanyen d'una etiqueta fina (per exemple, *<n><m><pl>* per als noms masculins en plural); el desambiguador categorial agrupa aquestes etiquetes fines en etiquetes més generals (per exemple, la categoria `NOM` per a tots els noms), encara que a la seva sortida torna a lliurar l'etiqueta fina completa de cada FL; finalment, en el mòdul de transferència, les etiquetes fines de les FL s'agrupen altra vegada en categories més generals (encara que poden definir-se també categories detallades) segons el tipus de formes lèxiques que es vulgui detectar en els patrons.

Cada element <**cat-item**> té un atribut obligatori `tags` el valor del qual és una seqüència de símbols gramaticals separats per punts; aquesta seqüència és una subseqüència de la etiqueta fina, és a dir, de la seqüència de símbols gramaticals que defineix cada possible forma lèxica lliurada pel desambiguador lèxic categorial. D'aquesta manera, una categoria representa un determinat

```
<def-cat n="nom"/>
  <cat-item tags="n.*"/>
</def-cat>
<def-cat n="que"/>
  <cat-item lemma="que" tags="cnjsub"/>
  <cat-item lemma="que" tags="rel.an.mf.sp"/>
</def-cat>
```

Figura 1.1: Ús de l'element <**cat-item**> per a definir dues categories, una per a noms sense especificar cap lema (*nom*) en la qual s'inclouen totes les formes lèxiques el primer símbol de les quals siga *n*, i altra amb lema associat (*que*), que té més d'un símbol gramatical associat, per a incloure *que* com a *conjunció* i *que* com a *relatiu* en castellà.

```
<def-cat n="det-nom"/>
  <cat-item name="det-nom"/>
</def-cat>
```

Figura 1.2: Ús de l'element <**cat-item**> en el mòdul postchunk per a detectar chunks de determinant-nom.

conjunt de formes lèxiques. Hem de definir tantes categories diferents com tipus de formes lèxiques vulguem detectar en els patrons. Així, si ens interessa detectar tots els noms (cas de l'intrachunk) o tots els sintagmes nominals de determinat tipus (cas de l'interchunk) per a realitzar operacions amb ells, crearem una categoria definida a partir del símbol gramatical `n`. D'altra banda, si ens interessa detectar tots els noms femenins i plurals, hauríem de definir una categoria mitjançant els símbols `n f` i `pl`.

Quan un símbol gramatical utilitzat per a definir una categoria ve seguit per altres símbols gramaticals en el grup de lemes que se volen incloure, s'utilitza el caràcter `"*"`. Per exemple, `tags="n.*"` abasta totes les formes lèxiques que continguen aquest símbol, com ara `casa<n><f><pl>` o `cotxe<n><m><sg>` per al cas de l'intrachunk; l'interchunk seria equivalent però amb pseudoparaules. En canvi, quan darrere del símbol utilitzat no pot venir cap altre símbol, no s'inclou l'asterisc: per exemple, `tags="adv"` abastarà tots els adverbis. També pot utilitzar-se l'asterisc per a indicar l'existència de símbols precedents: `tags="*.f.*"` inclou a totes les formes lèxiques femenines, siguin de la categoria que siguin. A més, pot utilitzar-se un atribut opcional, `lemma`, per a definir formes lèxiques a partir del seu lema (veure figura 1.1).

Ús en postchunk

Aquest element només preveu l'atribut obligatori `name`, que fa referència al nom de la regla, sense etiquetes, ja que al mòdul `postchunk` només s'usa el pseudolema per a la detecció. En la detecció no distingeix entre majúscules i minúscules, perquè la informació sobre les majúscules i les minúscules del segment va en el pseudolema.

```
<def-attr n="nbr"/>
  <attr-item tags="sg"/>
  <attr-item tags="pl"/>
  <attr-item tags="sp"/>
  <attr-item tags="ND"/>
</def-attr>

<def-attr n="anom"/>
  <attr-item tags="n"/>
  <attr-item tags="n.acr"/>
</def-attr>
```

Figura 1.3: Definició de l'atribut de categoria *nombre* `nbr`, que pot prendre els valors *singular*, *plural*, *singular-plural* o *nombre per determinar*, així com de l'atribut de categoria *nom*, que pot prendre els valors dels símbols *n* o *n.acr*.

1.6.7 Element de secció de definició d'atributs de categoria <section-def-attrs>

En aquesta secció es defineixen els atributs que s'extrauran de les categories detectades pel patró i que s'utilitzaran en la part d'acció de les regles. Cada atribut es defineix mitjançant una etiqueta<**def-attr**>.

1.6.8 Element de definició d'atributs de categoria <def-attr>

Cada <**def-attr**> defineix un atribut amb informació morfològica (tant informació flexiva –gènere, nom, persona, etc.–, com categorial –verb, adjectiu, etc–) mitjançant una llista d'elements d'atribut de categoria (<**attr-item**>) i té un nom únic obligatori `n`. Un atribut es defineix, per tant, a partir dels símbols morfològics que poden trobar-se en una forma lèxica donada. Cada atribut extreu, de les formes lèxiques del patró, els símbols que aquestes contenen d'entre els possibles valors donats.

1.6.9 Element d'atribut de categoria <attr-item>

Cada element d'atribut de categoria representa un dels valors possibles que pot prendre l'atribut. Per exemple, l'atribut de número `nbr` pot prendre els valors singular (`sg`), plural (`pl`), singular–plural (`sp`) i nombre per determinar (`ND`). Aquests valors són una subseqüència de les etiquetes morfològiques que caracteritzen a les formes lèxiques i s'indiquen en l'atribut `tags` de l'element, separades per punts si hi ha més d'una. En la figura 1.3 pot veure's un exemple d'ús d'aquest element per a l'atribut de *nombre* i l'atribut de *nom*. Compareu la definició de l'atribut de nom en aquesta figura (amb tots els valors possibles i sense asteriscos) amb la definició de la categoria de nom de la figura 1.1.

1.6.10 Element de secció de definició de variables <section-def-vars>

En aquesta secció es defineixen (mitjançant etiquetes <**def-var**>) les variables globals de tipus cadena que s'utilitzaran per a transferir informació dins d'una

```
<def-list n="verbsestat">
  <list-item v="actuar"/>
  <list-item v="buscar"/>
  <list-item v="estudiar"/>
  <list-item v="existir"/>
  <list-item v="ingressar"/>
  <list-item v="introduir"/>
  <list-item v="penetrar"/>
  <list-item v="publicar"/>
  <list-item v="treballar"/>
  <list-item v="viure"/>
</def-list>
```

Figura 1.4: Definició d'una llista de lemes en català. Aquests lemes s'utilitzen en una regla que es pot veure en la figura 1.10.

regla i d'una regla a altra (per exemple, per a poder transmetre informació de gènere o nombre entre dos patrons).

1.6.11 Element de definició de variables <def-var>

La definició d'una variable global de tipus cadena té un nom únic obligatori`n` que s'utilitzarà per a referir-se a la mateixa dintre de les regles. Les variables transmeten cadenes que descriuen informació d'estat, com la existència de concordança entre dos elements, la detecció de un signe d'interrogació que calga eliminar en LM, etc.

1.6.12 Element de secció de definició de llistes de cadenes <section-def-lists>

En aquesta secció es defineixen (mitjançant etiquetes <**def-list**>) les llistes que s'utilitzaran per a realitzar recerques de cadenes. Aquestes llistes es poden utilitzar per a agrupar lemes de paraules que tinguen alguna característica comuna (per exemple, verbs que expressen moviment, adjectius que expressen estats d'ànim, etc.). Aquesta secció és opcional.

1.6.13 Element de definició de llistes de cadenes <def-list>

Aquest element serveix per a posar-li nom a la llista de cadenes mitjançant l'atribut `n` i per a encapsular la llista que es defineix mitjançant un o més elements <**list-item**>. Pot veure's un exemple d'ús en la figura 1.4.

1.6.14 Element de membre de llista de cadenes <list-item>

Aquest element defineix, mitjançant el valor de l'atribut `v`, la cadena concreta que s'inclou en la definició de la llista que encapsula aquest element. Pot veure's un exemple d'ús en la figura 1.4.

1.6.15 Element de secció de definició de macroinstruccions <section-def-macros>

En aquesta secció es defineixen les macroinstruccions que contenen fragments de codi utilitzat amb freqüència en la part d'acció de les regles.

1.6.16 Element de definició de macroinstruccions <def-macro>

Cada definició d'una macroinstrucció té un nom obligatori (el valor de l'atribut `n`), el nombre d'arguments que se li passen com a referència (atribut `npar`) i un cos amb instruccions.

1.6.17 Element de secció de regles <section-rules>

Aquesta secció conté les regles de transferència estructural, cadascuna en un element <**rule**>.

1.6.18 Element de regla <rule>

Cada regla té un patró (<**pattern**>) i l'acció (<**action**>) associada a ell que s'executa quan és detectat.

La regla pot especificar en un atribut opcional `comment` un comentari que explica quina és la funció de la regla.

1.6.19 Element de patró <pattern>

Un patró s'especifica utilitzant components de patró (<**pattern-item**>), cadascun dels quals correspon a una forma lèxica en el patró detectat, en ordre d'aparició.

1.6.20 Element de component de patró <pattern-item>

En cada component d'un patró s'indica, mitjançant un atribut de nom obligatori `n`, quin tipus de forma lèxica es vol detectar. Per a això s'utilitza una de les categories definides en<**section-def-cats**> (es pot veure un exemple de com es detecta el patró determinant–nom en la figura 1.12).

1.6.21 Element d'acció <action>

En aquest element s'inclouen les "instruccions" que s'han d'executar per a portar a terme el processament que es desitgi per a cada detecció de patrons. La part de processament del patró detectat és un bloc de zero o més instruccions de tipus: <**choose**> (processament condicional), <**let**> (assignació de valors), <**out**> (escriptura de formes lèxiques de la LM),<**modify-case**> (modificació de l'estat de les majúscules i minúscules d'una forma lèxica), <**call-macro**> (cridades a macroinstruccions) i <**append** (concatenació de cadenes).

Durant el processament, segons si es compleixen o no una sèrie de opcions condicionals, es realitzen diferents operacions, com ara la concordança dels elements d'un patró, ja que aquests estan condicionats a possibles canvis de

```
<append n="temporal">
  <clip pos="3" part="gen" side="tl"/>
</append>
```

Figura 1.5: En aquest exemple es concatena a la variable de nom `temporal` el valor de gènere de la part de LM de la tercera paraula detectada per la regla.

gènere o nombre durant el procés de transferència lèxica. Per això, a pesar de treballar amb FL en LM també es té en compte la informació de la LO ja que si, per exemple, en LO els elements d'un patró no concorden, potser tampoc han de fer-ho en LM. Durant l'aplicació de les distintes operacions realitzades dintre un patró, s'assignen valors a atributs del mateix, si escau, a variables globals o d'estat i s'envia la informació del patró en LM resultant al següent mòdul (el generador morfològic).

1.6.22 Element de concatenació de cadenes <append>

La instrucció <**append**> serveix per emmagatzemar l'eixida abans d'imprimir-la per conveniència del dissenyador de les regles de transferència.

L'atribut obligatori `n` especifica el nom de variable que té l'expressió. Al final de la instrucció, el contingut antic de la variable referenciada serà prefix del nou contingut.

El contingut d'aquesta instrucció pot ser una o més de les següents etiquetes: <**b**>, <**clip**>, <**lit**>, <**lit-tag**>, <**var**>, <**get-case-from**>, <**case-of**> o <**concat**>. Se pot veure un exemple a la figura 1.5.

1.6.23 Element de cridada a macroinstrucció <call-macro>

Dintre d'una regla pot invocar-se qualsevol de les macroinstruccions definides en <**section-def-macros**>. Per a això, cal especificar el seu nom en l'atribut `n` i un o més arguments en l'element de paràmetres <**with-param**> (veure a continuació).

1.6.24 Element de paràmetres <with-param>

Aquest element s'utilitza dintre de les cridades a macroinstruccions <**call-macro**>. L'atribut `pos` de cada argument s'utilitza per a referir-se a una forma lèxica de la regla que invoca la macroinstrucció. Per exemple, si s'ha definit una macroinstrucció de 2 paràmetres que realitza operacions de concordança nom–adjectiu, pot utilitzar-se amb els arguments 1 i 2 en una regla de nom–adjectiu, amb els arguments 2 i 3 en una regla de determinant–nom–adjectiu, amb els arguments 1 i 3 en una regla de nom–adverbi–adjectiu i amb els arguments 2 i 1 en una regla d'adjectiu–nom. Es pot veure un exemple de cridada a macroinstrucció en la figura 1.6.

```
<call-macro n="fconcord2">
  <with-param pos="3"/>
  <with-param pos="1"/>
</call-macro>
```

Figura 1.6: Cridada a la macroinstrucció `f-concord2` per a fer concordar els elements d'un patró com per exemple determinant–adverbi–nom. La propagació de gènere i nombre es fa a partir d'un dels elements, en aquest cas el nom que apareix com a tercer element del patró (3). Per això, la posició del nom es passa com primer paràmetre i després es passen la resta d'elements. Com que l'adverbi (en posició 2) no necessita informació de concordança, només es passa la posició del determinant (1).

1.6.25 Element de selecció <choose>

La instrucció de selecció es compon d'una o més opcions condicionals (<**when**>) i una opció alternativa<**otherwise**>, d'inclusió opcional.

1.6.26 Element de condició <when>

Amb aquest element es descriu una opció condicional (veure pàgina 22). Es compon de la condició a verificar<**test**> i d'un bloc de zero o més instruccions de tipus <**choose**>, <**let**>,<**out**>, <**modify-case**> i<**call-macro**>, que s'executaran si es compleix la citada condició.

1.6.27 Element d'opció alternativa <otherwise>

L'element <**otherwise**> conté un bloc d'una o més instruccions (de tipus <**choose**>, <**let**>, <**out**>, <**modify-case**>, <**call-macro**> i **append**>) que han de realitzar-se si no és certa la condició descrita en cap dels <**when**> d'un<**choose**>.

1.6.28 Element d'avaluació <test>

L'element d'avaluació <**test**> dintre d'un element de condició <**when**> pot estar compost per una conjunció (<**and**>), una disjunció (<**or**>) o una negació (<**not**>) de condicions a avaluar, així com per una senzilla condició d'igualtat(<**equal**>).

1.6.29 Elements d'operadors condicionals o booleans: <equal>, <and>, <or>, <not>, <in>

- L'element de conjunció <**and**> representa una condició, composta de dues o més condicions, que es compleix quan totes les condicions en ell incloses són certes. Es pot veure un exemple del seu ús en la figura 1.12

- L'element de disjunció <**or**> representa una condició, composta de dues o més condicions, que es compleix quan almenys una d'elles és certa. En la figura 1.9 es presenta un exemple d'aquest tipus de condició per a avaluar la concordança de gènere d'un patró en LO.

- L'element de negació <**not**> representa una condició que es compleix quan la condició inclosa no es compleix i viceversa. Es pot veure un exemple de negació d'una igualtat en la figura 1.9.

- L'operador condicional més simple és el d'igualtat (<**equal**>). És una instrucció que comprova si dos arguments (dues cadenes) són o no idèntics. Es pot veure un exemple de l'ús d'aquest element en les figures 1.7 i 1.8. En aquest operador, a més, es pot especificar l'atribut `caseless`, el qual, si té el valor `yes`, fa que la comparació entre cadenes es realitzi sense tenir en compte les diferències de majúscules i minúscules.

- L'operador de recerca en llistes, <**in**>, que serveix per a buscar qualsevol valor (que correspon al primer paràmetre de la condició en una llista definida en la secció corresponent (<**section-def-lists**), a la qual es fa referència mitjançant l'atribut `n` de l'element <**list**>. La recerca és certa si el valor es troba en la llista. En aquesta comparació es pot utilitzar també l'atribut `caseless`: si se li assigna el valor `yes`, es realitza la recerca en la llista sense tenir en compte les diferències de majúscules i minúscules. En la figura 1.10 pot veure un exemple del seu ús.

1.6.30 Element <clip>

L'element <**clip**> representa una subcadena d'una forma lèxica de la LO o de la LM, definida pel valor de diversos atributs (es pot veure un exemple d'ús en la figura 1.7):

- `pos` és un índex (1, 2, 3, etc.) utilitzat per a seleccionar una forma lèxica dintre de la regla: es correspon amb el lloc que ocupa la forma lèxica en el patró. En el mòdul *postchunk*, també existeix l'índex "0" i fa referència al pseudolema del chunk que és tractat com una paraula més, per tal d'accedir a la seua informació i prendre decisions a partir d'ella.

- `side` *(només transfer/intrachunk)* especifica si se selecciona un *clip* de la LO (`sl`, per *source language*) o de la LM (`tl`, per *target language*).

- en `part` s'indica quina part de la forma lèxica es processa. Generalment el seu valor és un dels atributs definits en <**section-def-attrs**> (`gen`, `nbr`, etc.), encara que també pot prendre quatre valors predefinits que són: `lem` (per a fer referència al lema de la forma lèxica), `lemh` (per a fer referència a la primera part d'un lema partit), `lemq` (la cua del lema partit), i `whole` (la forma lèxica completa, amb el lema i tots els símbols gramaticals després d'aplicar o no les possibles transformacions en la part precedent de la regla).

- `link-to` *(només transfer/intrachunk)* substitueix el valor que es derivaria de la consulta de la resta d'atributs pel valor especificat en l'atribut, que ha de ser un número natural (> 0), i que serveix per a indicar dins dels elements <**chunk**> a quina etiqueta del segment es fa referència des d'una unitat lèxica <**lu**>. La resta d'atributs queden com a documentació que enllaça la informació de processament amb la informació lingüística.

```
<test>
  <not>
    <equal>
      <clip pos="2" side="tl" part="gen"/>
      <clip pos="2" side="sl" part="gen"/>      </equal>
  </not>
</test>
```

Figura 1.7: Fragment d'una regla en la qual es comprova si el gènere (`gen`) en LM (`tl`) de la segona unitat lèxica identificada en un patró és distint del gènere de la mateixa unitat lèxica en LO (`sl`)

```
<equal>
  <clip pos="2" side="tl" part="nbr"/>
  <lit-tag v="ND"/>
</equal>
```

Figura 1.8: Ús de l'element <**lit-tag**>: es comprova si l'etiqueta (o símbol) de nombre (`nbr`) de la segona unitat lèxica de la LM (`tl`) és `ND` (nombre per determinar)

1.6.31 Element de cadena literal <lit>

Aquest element serveix per a especificar el valor d'una cadena literal mitjançant l'atribut `v`. Per exemple, <**lit** `v="caminar"/>` representa la cadena *caminar*.

1.6.32 Element de valor d'etiqueta <lit-tag>

És similar a l'element <**lit**> però no especifica el valor d'una cadena literal sinó el d'una etiqueta morfològica, mitjançant l'atribut `v`. Es pot veure un exemple del seu ús en la figura 1.8.

1.6.33 Element de concatenació de cadenes <concat>

Aquest element s'usa per a concatenar cadenes per a assignar-les a una variable. S'usa de manera combinada amb <**let**>, i el valor anterior de la variable que es modifica amb l'assignació de `</textbfconcat>`, es perd.

No té cap atribut. Pot contenir qualsevol instrucció que retorna una cadena, com ara <**lit**>, <**lit-tag**> o <**clip**>.

1.6.34 Element de variable <var>

Cada <**var**> és un identificador de variable: l'atribut obligatori `n` indica el seu nom tal com ha sigut definit en <**section-def-vars**>. Quan es troba en un <**out**>, un <**test**>, o en la part dreta d'un <**let**>, representa el valor de la variable; quan es troba en la part esquerra d'un <**let**>, dins d'un <**append**> o en `un` <**modify-case**>, representa la referència de la variable i es pot canviar el seu valor.

```
<test>
  <or>
    <not>
      <equal>
        <clip pos="1" side="sl" part="gen"/>
        <clip pos="3" side="sl" part="gen"/>
      </equal>
    </not>
    <not>
      <equal>
        <clip pos="2" side="sl" part="gen"/>
        <clip pos="3" side="sl" part="gen"/>
      </equal>
    </not>
  </or>
</test>
```

Figura 1.9: Fragment d'una regla en la qual es comprova si el gènere en LO de la primera o de la segona unitat lèxica identificada en un patró (podria ser determinant–adjectiu–nom) és distint del gènere de la tercera unitat lèxica també en LO.

1.6.35 Element de referència a llista de cadenes <list>

Aquest element només s'usa com a segon paràmetre d'una recerca <**in**>. L'atribut `n` fa referència a la llista concreta definida en la secció de definició de llistes de cadenes <**section-def-lists**>. Es pot veure un exemple d'ús la figura 1.10.

1.6.36 Element d'informació de majúscules/minúscules <get-case-from>

L'element <**get-case-from**> representa la cadena resultant d'aplicar l'esquema de majúscules que presenta el lema d'una unitat lèxica en LO a una cadena (*clip*, *lit* o *var*). Per a fer referència a la unitat lèxica de la qual es prendrá la informació, s'utilitza l'atribut `pos` el qual indica la posició d'aquesta unitat lèxica en LO. S'utilitza quan es reordenen les unitats lèxiques d'un patró o quan s'afegeix o es suprimeix una unitat lèxica. Es pot veure un exemple d'ús en la figura1.11, que conté una regla per a transformar el pretèrit perfet simple en espanyol (*dije*) pel pretèrit perfet perifràstic en català (*vaig dir*). En aquesta regla s'afegeix una FL amb lema *anar* i símbol morfològic *vaux* ("verb auxiliar"), el qual ha de prendre la informació de majúscules del verb en espanyol (que té la posició "1"en el patró), perquè es traduisca *Dije* per *Vaig dir*, *dije* per *vaig dir* i *DIJE* per *VAIG DIR*.

1.6.37 Element d'obtenció del patró de majúscules/minúscules <case-of>

Serveix per a obtenir el patró de majúscules/minúscules, és a dir, un dels valors "`aa`", "`Aa`" o "`AA`", d'una cadena, que s'obté com en el cas de l'etiqueta <**clip**>. Té els mateixos atributs: `pos`, la posició que ocupa la paraula en

```
<rule>
  <pattern>
    <pattern-item n="verb"/>
    <pattern-item n="a"/>
  </pattern>
  <action>
    <choose>
      <when>
        <test>
          <in caseless="yes"/>
            <clip pos="1" side="sl" part="lem"/>
            <list n="verbsestat"/>
          </in>
        </test>
        <let>
          <clip pos="2" side="tl" part="lem"/>
          <lit v="en"/>
        </let>
      </when>
      <!-- ... -->
```

Figura 1.10: Fragment d'una regla que detecta un patró format per un verb més la preposició *a* i a continuació comprova si el verb (el lema indicat en `lem`) de la LO (`sl`) és un dels que s'han inclòs en la llista de verbs d'estat (definida en la figura 1.4). En cas afirmatiu, el lema de la segona paraula de la LM (`tl`) es canvia per *en* .

el patró detectat; `part`, l'atribut concret a què es fa referència (normalment el lema), amb els atributs predefinits que ja s'esmenten en la pàgina 23, i a més, només per al cas del mòdul transfer/intrachunk, l'atribut `side`, el costat de la traducció, `sl` per a la LO i `tl` per a la LM. En la figura 1.11 es pot veure un exemple d'utilització d'aquest element, i en l'apartat següent (sobre ¡**modify-case**>) es troba una explicació més detallada de l'exemple.

1.6.38 Element de modificació d'estat de majúscules/minúscules <modify-case>

Aquesta instrucció serveix per a modificar les majúscules o minúscules de la primera expressió (el primer paràmetre, típicament un lema) mitjançant un literal o una variable (el segon paràmetre). El primer paràmetre pot ser una <**var**>, un <**clip**> o un <**case-of**>, mentre que el segon pot ser qualsevol cosa que retorne un valor. En principi, serà <**var**> o <**lit**>.

Els valors que poden prendre aquests paràmetres són "`Aa`", per a expressar que la "part esquerra" d'aquesta modificació de les majúscules ha de tenir la primera lletra en majúscules i la resta en minúscules;"`aa`", per a posar tot en minúscules i "`AA`", per a posar tot en majúscules. En la figura 1.11 es pot veure un exemple de la seua utilització. En aquesta regla es modifica l'estat de majúscules del lema en LM en posició "1", que correspon a *dir*, ja que, encara que a l'eixida de la regla siga la segona forma lèxica (*vaig dir*), és la traducció de la FL la que té la posició 1 en LO, i per tant continua tenint assignada aquesta posició en la LM. A aquest lema se li assigna el valor "`aa`"en cas que el lema

en LO tingui l'estat "Aa". A la resta de casos no cal especificar res, ja que l'estat de majúscules/minúscules en la FL amb posició 1 coincidirà en LO i en la LM i, per tant, es transfereix automàticament.

1.6.39 Element d'assignació <let>

La instrucció d'assignació <**let**> assigna el valor de la part dreta de l'assignació (una cadena literal, un clip, una variable, etc.) a la part esquerra (un clip, una variable, etc.). Es pot veure un exemple d'ús en la figura 1.12.

1.6.40 Element de sortida <out>

En la instrucció d'eixida s'especifiquen les formes lèxiques que s'envien a l'eixida del mòdul després d'haver estat sotmeses a les operacions de transferència estructural pertinents. L'ús d'aquest element varia segons el mòdul. Per una banda, l'ús del mode transfer i del postchunk són similars, perquè l'eixida que s'ha d'imprimir en els dos casos és entrada de generador. Els mòduls intrachunk i interchunk tenen modes d'ús diferents: el primer per a crear els segments i el segon per a modificar els segments sense modificar la seua part interna.

Ús en el mode transfer i en el mòdul postchunk

La instrucció envia cada forma lèxica dintre d'un conjunt <**lu**>, que pot estar contingut dintre d'un element <**mlu**> si el que s'envia és una multiparaula composta de dues o més FL. A més, s'envien també els espais en blanc o superblancs (<**b**>) que calgui col·locar entre FL i FL. En les figures 1.11 i 1.13 es pot veure un exemple d'ús.

Ús en mode intrachunk

L'eixida d'aquest mòdul s'espera que siga una seqüència de chunks separada per blancs. No conté directament formes lèxiques, si no és dins dels segments.

Ús en el mòdul interchunk

En aquest mòdul les formes lèxiques de les paraules són inaccessibles, per tant només són possibles les operacions amb els chunks. La informació de lema i d'etiquetes que hi ha dins de l'element <**chunk**> corresponen al lema del chunk i a les etiquetes del chunk exclusivament.

Es pot veure un exemple d'ús a la figura 1.15.

1.6.41 Element d'unitat lèxica <lu>

El seu nom prové de *lexical unit* o "unitat lèxica" i és l'element mitjançant el qual s'envia cada forma lèxica del patró en LM a l'eixida de la regla, dintre de l'element<**out**>. Mitjançant aquest element pot enviar-se la forma lèxica completa, amb l'atribut whole d'un <**clip**>, o bé, en cas necessari, indicar explícitament cadascuna de les seves parts (lema i etiquetes, especificades mitjançant cadenes <**clip**>, cadenes literals <**lit**>, etiquetes <**lit-tag**>, variables <**var**>,

```
<rule>
  <pattern>
    <pattern-item n="pretind"/>
  </pattern>
  <action>
    <out>
      <lu>
        <get-case-from pos ="1">
        <lit v="anar"/>
        </get-case-from>
        <lit-tag v="vaux"/>
        <clip pos="1" side="sl" part="persona"/>
        <clip pos="1" side="sl" part="nbr"/>
      </lu>
      <b/>
    </out>
    <choose>
      <when>
        <test>
          <equal>
            <case-of pos="1" side="sl" part="lemh"/>
            <lit v="Aa"/>
          </equal>
        </test>
        <modify-case>
          <case-of pos="1" side="tl" part="lemh"/>
          <lit v="aa"/>
        </modify-case>
      </when>
    </choose>
    <out>
      <lu>
        <clip pos="1" side="tl" part="lemh"/>
        <clip pos="1" side="tl" part="averb"/>
        <lit-tag v="inf"/>
        <clip pos="1" side="tl" part="lemq"/>
      </lu>
    </out>
  </action>
</rule>
```

Figura 1.11: Regla per a traduir de castellà a català que transforma els verbs en pretèrit perfet simple o indefinit en castellà (*dije*) en la forma de pretèrit perfecte perifràstic usual en català (*vaig dir*). Al mateix temps assigna la informació correcta de majúscules/minúscules a les dues paraules resultants.

```
<rule>
  <pattern>
    <pattern-item n="det"/>
    <pattern-item n="nom"/>
  </pattern>
  <action>
    <choose>
      <when>
        <test>
          <and>
            <not>
              <equal>
                <clip pos="2" side="tl" part="gen"/>
                <clip pos="2" side="sl" part="gen"/>
              </equal>
            </not>
            <not>
              <equal>
                <clip pos="2" side="tl" part="gen"/>
                <lit-tag v="mf"/>
              </equal>
            </not>
            <not>
              <equal>
                <clip pos="2" side="tl" part="gen"/>
                <lit-tag v="GD"/>
              </equal>
            </not>
          </and>
        </test>
        <let>
          <clip pos="1" side="tl" part="gen"/>
          <clip pos="2" side="tl" part="gen"/>
        </let>
      </when>
    </choose>
    <!-- Altres operacions de concordança de gènere i nombre -->
```

Figura 1.12: Fragment d'una regla en la qual el patró detectat és determinant–nom (continua en la fig. 1.13): en aquesta part de la regla, s'assigna el gènere del nom al determinant en cas que el nom canvie de gènere entre LO (`sl`) i la LM (`tl`) durant el procés de transferència lèxica entre ambdues llengües.

```
<!-- ... -->
  <out>
    <lu>
      <clip pos="1" side="tl" part="whole"/>
    </lu>
    <lu>
      <clip pos="2" side="tl" part="whole"/>
    </lu>
    </out>
  </action>
</rule>
```

Figura 1.13: Fragment d'una regla (ve de la fig. 1.12). Al final de la regla, i després de diverses operacions, es dóna eixida a la informació resultant mitjançant l'atribut `whole`, el qual conté el lema i els símbols morfològics de cada FL (posicions 1 i 2 del patró).

```
<out>
  <chunk name="pr" case="caseFirstWord">
    <tags>
      <tag><lit-tag v="PREP"/></tag>
    </tags>
    <lu>
      <clip pos="1" side="tl" part="whole"/>
    </lu>
  </chunk>
  <b pos="1"/>
  <chunk name="probj" case="caseOtherWord">
    <tags>
      <tag><lit-tag v="SN"/></tag>
      <tag><lit-tag v="tn"/></tag>
      <tag><clip pos="2" side="tl" part="pers"/></tag>
      <tag><clip pos="2" side="tl" part="gen"/></tag>
      <tag><clip pos="2" side="tl" part="nbr"/></tag>
    </tags>
    <lu>
      <clip pos="2" side="tl" part="lem"/>
      <lit-tag v="prn"/>
      <lit-tag v="2"/>
      <clip pos="2" side="tl" part="pers"/>
      <clip pos="2" side="tl" part="gen" link-to="4"/>
      <clip pos="2" side="tl" part="nbr" link-to="5"/>
    </lu>
  </chunk>
</out>
```

Figura 1.14: Instrucció d'eixida que imprimeix dos segments separats per un blanc. La seqüència que s'imprimeix és la de preposició seguida de sintagma nominal. Les etiquetes que s'enllacen a fora com a referència són la persona, el gènere i el nombre del sintagma nominal. Usa les etiquetes <**tag**> per a especificar les etiquetes del segment, i el valor dels atributs `name` i `case` per a especificar la forma lèxica del chunk.

```
<out>
  <b pos="1">
  <chunk>
    <clip pos="2" part="lem"/>
    <clip pos="2" part="tags"/>
    <clip pos="2" part="chcontent"/>
  </chunk>
</out>
```

Figura 1.15: Aquesta eixida s'ha fet per a descartar el primer chunk d'una regla (caiguda de pronom. L'ús dels tres elements <**clip**>, s'fet ací amb fins il·lustratius, s'hauria pogut substituir per la part `whole` que els agruparia en un únic <**clip**> .

així com informació de majúscules i minúscules (<**get-case-from**>, <**case-of**>). Cal tenir en compte que, com s'ha explicat abans, en el cas de les multiparaules amb *lema partit* cal recol·locar la cua d'un lema multiparaula darrere dels símbols gramaticals de la paraula flexionada (o cap del lema), ja que el mòdul `pretransfer` ha avançat la cua i l'ha col·locat abans dels símbols gramaticals del començament. Aquesta recol·locació es realitza aquí, dintre de l'element <**lu**>, mitjançant els valors `lemh` i `lemq` de l'atribut `part` d'un <**clip**>. L'atribut `lemh` correspon al començament de lema, i l'atribut`lemq`, a la cua del lema. Com es veu en l'exemple1.11, la part `lemq` del <**clip**> és col·locada darrere del començament del lema i dels símbols gramaticals que l'acompanyen. Aquesta regla serviria, per exemple, per a la forma en espanyol *eché de menos*, que ha de traduir-se al català per *vaig trobar a faltar*. L'atribut `a_verb` que apareix darrere de `lemh` conté el símbol gramatical que descriu la categoria del verb (*vblex*, *vbser*, *vbhaver* o *vbmod* segons el cas). Així, l'última forma lèxica enviada per aquesta regla, per al cas de *vaig trobar a faltar*, seria en el flux de dades:

```
^trobar<vblex><inf># a faltar$
```

El símbol coixinet del flux de dades es correspon amb l'element <**g**> dels diccionaris, utilitzat per a indicar la posició de les parts invariables de una unitat multiparaula amb lema partit. És important tenir en compte que els atributs que s'inclouen dins de <**lu**> poden estar buits. Així, un verb que entre per la regla de la figura 1.11 i que no sigui un multimot amb lema partit, serà enviat amb l'atribut `lemq` buit, ja que no té cua de lema. D'aquesta manera no cal definir regles diferents per a formes lèxiques amb cua i sense cua. D'aquesta manera s'inclouen els participis (amb gènere i nombre) i les altres formes verbals (les quals duran aquests atributs buits). En la mateixa pàgina es pot veure una regla per a verb seguit de pronom enclític. Aquí, la cua del lema es col·loca darrere del pronom enclític; així, les formes lèxiques enviades en el cas d'una multiparaula unida a un pronom enclític, com ara *echándote de menos*, serien, en el flux de dades i en la traducció cap al català:

```
^trobar<vblex><ger>+et<prn><enc><p2><mf><sg># a faltar$
```

Per descomptat, aquesta regla serveix també per a verbs que no segueixin aquest patró de multiparaula, de manera que la forma *explicándote* seria d'aquesta manera enviada per la regla en la traducció d'espanyol a català:

```
^explicar<vblex><ger>+et<prn><enc><p2><mf><sg>$
```

Quant a l'atribut `whole` d'un <**clip**>, és important tenir en compte que només pot utilitzar-se per a enviar la forma lèxica completa en cas que la paraula enviada no puga ser una multiparaula, és a dir, que no puga contenir un lema partit. Compareu les figures 1.11 i 1.13. L'atribut `whole` pot utilitzar-se en el segon exemple perquè conté el lema `lem` i totes les etiquetes morfològiques de les formes lèxiques en posició 1 i 2 (determinant i nom). En canvi, en el primer exemple, la forma lèxica enviada dintre de <**lu**> s'envia per parts, amb un `lemh` (cap del lema) i un `lemq` (cua del lema), ja que pot ocórrer que el verb detectat en el patró sigui una multiparaula amb lema partit. En la pràctica, això significa que, en el nostre sistema, l'atribut `whole` pot utilitzar-se per a enviar qualsevol tipus de forma lèxica excepte els verbs i els noms, ja que només hem definit multiparaules amb flexió per a formes verbals i noms.

1.6.42 Element d'unitat lèxica <mlu>

El seu nom prové de *multilexical unit* o "multiforma lèxica"i s'utilitza dintre de l'element <**out**> per a enviar multiparaules compostes per més d'una forma lèxica. Cada forma lèxica d'una <**mlu**> s'envia dintre d'un element <**lu**>. A la sortida del mòdul, les formes lèxiques contingudes dintre d'aquest element apareixeran unides entre si mitjançant el símbol '+' del flux de dades. Això significa que es convertiran en una multiparaula composta per diverses formes lèxiques i que seran tractades com a una unitat pels mòduls subsegüents; per tant,ven el diccionari de generació haurà d'existir una entrada per a aquesta multiparaula perquè siga possible generar-la. En el nostre sistema, aquest element s'utilitza, per exemple, per a unir els pronoms enclítics als verbs conjugats.

1.6.43 Element d'encapsulament de fragment <chunk>

Aquest element només està definit en els mòduls intrachunk i interchunk. En el mòdul postchunk no s'usa perquè no és necessari crear cap chunk a l'eixida.

Ús en el mòdul intrachunk

En aquest mode, l'element <**chunk**> ha de tenir un atribut `name` que és el lema del segment, o bé un atribut `namefrom`, que fa referència a una variable prèviament definida, el valor de la qual s'usarà com a lema del segment. A més, pot incorporar un atribut `case` per a especificar de quina variable s'agafa la política de majúscules (amb un valor obtingut amb l'instrucció <**case-from**>, per exemple).

Es pot veure un exemple d'ús de l'element en la figura 1.14.

Ús en el mòdul interchunk

En aquest mòdul l'element <**chunk**> no especifica cap atribut, només s'usa a la manera que s'usa <**lu**> en el mode de transfer o en el postchunk per a delimitar la forma lèxica. La part invariable del segment passa al final, usualment amb

una instrucció de <**clip**> i l'ús de la part `chcontent`, el contingut invariable del chunk.

Es pot veure un exemple d'ús de l'element en la figura 1.15.

1.6.44 Element de secció d'enllaç d'etiquetes <tags>

Només transfer/intrachunk. Aquest element serveix per a especificar una llista d'etiquetes o elements <**tag**> que seran les pseudoetiquetes del chunk. No té atributs, i ha d'especificar-se al començament dels elements <**chunk**>.

1.6.45 Element d'enllaç d'etiquetes <tag>

Només transfer/intrachunk. L'element <**tag**> ha de contenir una etiqueta, que es pot especificar mitjançant una instrucció <**clip**> o bé d'etiqueta literal <**lit-tag**>. No té atributs.

1.6.46 Element de blanc <b>

L'element <**b**> fa referència als [super]blancs, i és indexat mitjançant l'atribut `pos`; per exemple, un <**b**> amb `pos="2"` es refereix als [super]blancs (incloent-hi les dades de format encapsulades pel desformatador) entre la 2a FLLO i la 3a FLLO. La gestió explícita de [super]blancs permet la col·locació correcta del format quan el resultat de la transferència estructural té més o menys elements lèxics que l'original o bé ha estat sotmès a algun tipus de reordenament.

1.6.47 Preprocessament del mòdul de transferència estructural

Els fitxers d'especificació dels mòduls de transferència estructural, també anomenats *fitxers de regles de transferència*, són preprocessats pel programa *apertium-preprocess-transfer* què calcula els patrons per a emparellar precondicions de les regles i a més indexa les regles per a accelerar el seu processament en temps d'execució. Aquesta informació s'emmagatzema en un arxiu en format binari que es llegeix conjuntament amb el diccionari bilingüe corresponent i el propi fitxer de regles de transferència per a l'execució conjunta dels mòduls de transferència lèxica i estructural.

Capítol 2

Mòdul de selecció lèxica

[Aquest capítol documenta el nou mòdul introduït a la cadena de muntatge d'Apertium i hauria d'ubicar-se entre les seccions 3.3 i 3.4 de la documentació de la versió 1 d'Apertium, a excepció de la secció 2.2 que hauria d'ubicar-se a la secció 3.1.2 (pàgina 26) on s'expliquen els diccionaris usats per Apertium.]

2.1 Introducció

La necessitat d'un mòdul per a triar la traducció correcta d'aquelles paraules que poden tenir més d'una traducció en llengua meta apareix quan ens proposem implementar traductors entre llengües menys emparentades que les que fins ara ha tractat Apertium.

El tractament de paraules que poden tenir més d'una traducció ha comportat, d'una banda, l'especificació de més d'una traducció al diccionari bilingüe i, d'altra, la introducció d'un nou mòdul el propòsit del qual és triar la traducció més apropiada d'acord amb el context.

La figura 2.1 mostra com queda la cadena de muntatge en la versió 2 d'Apertium.[1] El mòdul encarregat de la tria lèxica (selector lèxic) s'executa des-

[1] Aquesta figura reemplaça a la figura 1.1 que es troba a la pàgina 6 de la documentació de la versió 1 d'Apertium.

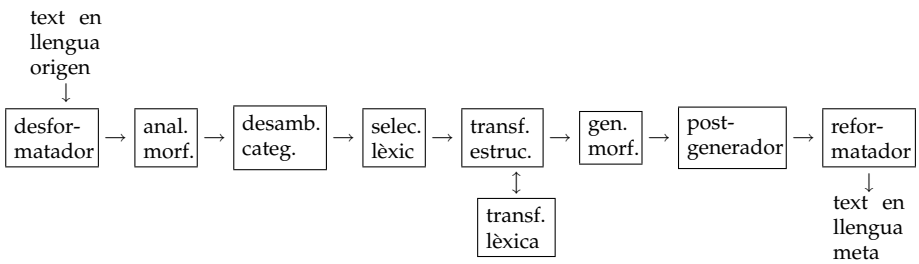

Figura 2.1: Els nou mòduls que formen la cadena de muntatge de la versió 2 del sistema de traducció automàtica Apertium.

```
...
<!ATTLIST e
  r (LR|RL) #IMPLIED
  lm CDATA #IMPLIED
  a CDATA #IMPLIED
  c CDATA #IMPLIED
  i CDATA #IMPLIED
  slr CDATA #IMPLIED
  srl CDATA #IMPLIED>

  <!-- r: restriction LR: left-to-right,
                      RL: right-to-left -->
  <!-- lm: lemma -->
  <!-- a: author -->
  <!-- c: comment -->
  <!-- i: ignore ('yes') means ignore, otherwise it is not ignored) -->
  <!-- slr: translation sense when translating from left to right -->
  <!-- srl: translation sense when translating from right to left -->
...
```

Figura 2.2: Fragment de la DTD `dix.dtd` on es defineix l'element `<e>` usat per a definir una entrada als diccionaris.

prés del desambiguador lèxic categorial i abans del mòdul encarregat de la transferència estructural; per tant, aquest nou mòdul treballa únicament amb informació de la llengua origen.

La resta del capítol s'organitza de la següent manera: la secció 2.2 explica la modificació feta per a especificar més d'una traducció al diccionari bilingüe; la secció 2.3 explica el preprocessament que s'ha de fer del diccionari bilingüe per tal d'obtenir els diccionaris a emprar en la tria lèxica. Finalment en la secció 2.4 s'explica com funciona i com s'ha d'entrenar el mòdul que fa la tria lèxica.

2.2 Modificació dels diccionaris

La modificació feta als diccionaris per a permetre l'especificació de més d'una traducció en llengua meta consisteix a afegir dos atributs nous al element `<e>`, el qual permet la definició d'entrades a tots els diccionaris. Aquests nous atributs només tenen sentit quan l'entrada és una entrada del diccionari bilingüe.

La figura 2.2 mostra el fragment de la DTD `dix.dtd` on es defineix l'element `e` usat per a codificar les entrades als diccionaris. Els nous atributs introduïts són:

slr (*sense from left to right*) s'utilitza per a especificar la *marca de traducció* quan la traducció és de esquerra a dreta. L'atribut pot rebre qualsevol valor; no obstant això, el més apropiat és que reba el valor del lema marcat com a `<r>`.

srl (*sense from right to left*) s'utilitza per a especificar la marca de traducció quan la traducció és de dreta a esquerra. Com abans, l'atribut pot rebre qualsevol valor, però el més apropiat és que reba el lema marcat com a `<l>`.

A més, en ambdós casos el valor de l'atribut pot acabar amb un espai en blanc i la lletra "D" (per *default*) per a indicar que aquesta és la traducció per defecte, és a dir, la traducció a emprar quan no hi ha suficient informació per a triar-ne una altra.

El següent exemple il·lustra el funcionament dels nous atributs. Considerem el cas d'un diccionari bilingüe anglès–català i les següents entrades amb més d'una traducció en llengua meta:

- *look*: que es pot traduir al català com *mirar* (per defecte) o com *parèixer*,
- *floor*: que es pot traduir al català per *pis* (per defecte) o per *terra*,
- *pis*: que es pot traduir a l'anglès per *flat* (per defecte) o per *floor* (prenent *pis* en el sentit de paviment).

Aquesta informació es representa fent ús dels dos atributs esmentats abans:

```
<e srl="flat D">
   <p>
      <l>flat<s n="n"/></l>
      <r>pis<s n="n"/><s n="m"/></r>
    </p>
</e>

<e slr="pis D" srl="floor">
   <p>
      <l>floor<s n="n"/></l>
      <r>pis<s n="n"/><s n="m"/></r>
   </p>
</e>

<e slr="terra">
   <p>
      <l>floor<s n="n"/></l>
      <r>terra<s n="n"/><s n="m"/></r>
   </p>
</e>

<e slr="mirar D">
   <p>
      <l>look<s n="vblex"/></l>
      <r>mirar<s n="vblex"/></r>
   </p>
</e>

<e slr="parèixer">
   <p>
      <l>look<s n="vblex"/></l>
      <r>parèixer<s n="vblex"/></r>
   </p>
</e>
```

2.3 Preprocessament dels diccionaris bilingües

Tot i que als diccionaris bilingües es pot especificar més d'una traducció per paraula, el motor de traducció d'Apertium treballa amb diccionaris compilats en els quals cada mot té només una traducció possible. El diccionari bilingüe s'ha de preprocessar

- per a obtenir un diccionari monolingüe que per a cada paraula en llengua origen (per exemple *look*) retorne totes les possibles marques de traducció (`look__mirar D` i `look__parèixer`), aquest diccionari serà utilitzat pel mòdul que fa la tria lèxica; i
- per a obtenir un nou diccionari bilingüe que donada una paraula amb la selecció lèxica ja feta (per exemple `look__parèixer`) en retorne la traducció (*parèixer*); aquest serà el diccionari bilingüe a emprar en la transferència lèxica.

El preprocessament esmentat abans es fa automàticament amb el següent programari:

- `apertium-gen-lextormono`, que rep tres paràmetres:
  - el sentit de traducció per al qual es vol obtenir el diccionari monolingüe usat en la tria lèxica; `lr` (*left to rigth*) per a la traducció d'esquerra a dreta, i `rl` (*right to left*) per a la traducció de dreta a esquerra;
  - el diccionari bilingüe a preprocessar; i
  - el fitxer on s'ha d'escriure el diccionari monolingüe resultant.
- `apertium-gen-lextorbil`, que rep tres paràmetres:
  - el sentit de traducció (`lr` o `rl`) per al qual es vol obtenir el diccionari bilingüe a usar pel mòdul de transferència lèxica;
  - el diccionari bilingüe a preprocessar; i
  - el fitxer on s'ha d'escriure el diccionari bilingüe resultant.

2.4 Mòdul per a la tria lèxica

El mòdul encarregat de la tria lèxica s'executa després del desambiguador lèxic categorial i abans de la transferència estructural (vegeu la figura 2.1 de la pàgina 35); per tant fa ús únicament d'informació de la llengua origen de la traducció. No obstant això, per a l'entrenament del mòdul també es fa ús d'informació de la llengua meta.

2.4.1 Entrenament

Per al entrenament del mòdul que fa la tria lèxica es necessiten un corpus en llengua origen i un altre en llengua meta; no cal que estiguen relacionats. Tots dos corpus han de ser preprocessats abans de l'entrenament. Aquest preprocessament, el qual consisteix a analitzar els còrpora i fer la desambiguació lèxica categorial, es pot fer amb `apertium-preprocess-corpus-lextor`.

L'entrenament del mòdul encarregat de la tria lèxica consisteix a fer les següents tasques:[2]

1. obtenir la llista de paraules que no hauran de ser tingudes en compte a l'hora de fer la tria lèxica (*stopwords*). Aquesta llista pot fer-se a mà o mitjançant `apertium-gen-stopwords-lextor`;
2. obtenir la llista de paraules (en llengua origen) que tenen més d'una traducció en llengua meta amb `apertium-gen-wlist-lextor`;
3. traduir a la llengua meta cada una de les paraules obtingudes en el pas anterior amb `apertium-gen-wlist-lextor-translation`;
4. entrenar amb `apertium-lextor --trainwrd` i usant el corpus preprocessat de la llengua meta un model de coaparició de paraules per a les obtingudes en el pas anterior;
5. entrenar amb `apertium-lextor --trainlch` i usant el corpus preprocessat de la llengua origen, els diccionaris generats amb els programes esmentats a la secció 2.3 i els models de coaparició de paraules estimats en el pas anterior, un model de coaparició per a cadascuna de les marques de traducció d'aquelles paraules que poden tenir més d'una traducció en llengua meta.

2.4.2 Utilització

Els models de coaparició de paraules estimats a la secció anterior per a cadascuna de les marques de traducció proporcionen la informació necessària per a portar a terme la tria lèxica amb informació del context.

La tria lèxica es fa amb `apertium-lextor --lextor`; els formats usats per a la comunicació amb la resta de mòduls del motor de traducció són:

Entrada: text en el mateix format que el d'entrada al mòdul de transferència estructural, és a dir, text analitzat i desambiguat amb les cues invariants dels multimots desplaçats abans de qualsevol etiqueta.

Eixida: text en el mateix format però amb la marca de traducció a emprar quan es faça la transferència lèxica.

El següent exemple il·lustra els formats d'entrada/eixida usats pel selector lèxic (suposem que únicament el verb anglès *get* té més d'un sentit de traducció als diccionaris):

- Text en llengua origen (anglès): *To get to the city centre*
- Entrada al selector lèxic: `^To<pr>$ ^get<vblex><inf>$ ^to<pr>$ ^the<det><def><sp>$ ^city<n><sg>$ ^centre<n><sg>$`
- Marques de traducció al diccionari bilingüe per al verb *get*: `rebre`, `agafar`, `arribar`, `aconseguir D`
- Eixida del selector lèxic: `^To<pr>$ ^get__arribar<vblex><inf>$ ^to<pr>$ ^the<det><def><sp>$ ^city<n><sg>$ ^centre<n><sg>$`

[2] L'entrenament dels models usats per a la tria lèxica s'ha automatitzat en aquells paquets lingüístics que l'usen. A més tot el programari esmentat té la seua pàgina de manual UNIX

Capítol 3

Formularis web d'inserció de dades

Aquest capítol presenta el sistema de manteniment de diccionaris d'Apertium 2. Està estructurat en dues seccions. En la secció 3.2 es dóna l'informació necessària per a instal·lar la ferramenta i realitzar adaptacions. En la secció 3.3 s'exposa el mode d'ús de l'aplicació web per a dur a terme la millora de les dades lingüístiques.

3.1 Introducció

El formularis per a la introducció de nous termes als diccionaris de les diferents llengües sorgeixen com a resposta a la necessitat de proveir una ferramenta amb la qual introduir còmodament noves paraules als diferents diccionaris i amb la possibilitat de fer-ho de forma remota des de qualsevol ordinador que tinga accés a Internet.

Així és plantegen com uns formularis que estan escrits en `php` i són utilitzables amb qualsevol navegador d'Internet, bé de forma local en el mateix ordinador on s'emmagatzemen els diccionaris o bé remotament.

3.2 Instal·lació i administració

3.2.1 Instal·lació de la ferramenta

La instal·lació s'ha de realitzar en un màquina Unix, que tinga instal·lat un servidor web Apache amb `php`. Així s'ha de procedir primerament a fer la instal·lació del servidor `php` si no estigués ja fet. Una vegada estiga resolt el punt anterior es procedeix a fer la instal·lació pròpiament dita.

S'obté el paquet *'apertium-lexical-webform-0.9'* i es descomprimeix dins del directori on vaja a deixar-se la ferramenta de formularis d'inserció.

```
# cd /ruta/dels/formularis
# tar -xvzf /ruta/apertium-lexical-webform-0.9.tar.gz
```

S'ha de tenir en compte que l'Apache sols serveix les pàgines que estan dintre del directori arrel amb què l'hem configurat. Per tant el directori on deixem els formularis ha de ser un subdirectori que estiga dintre de l'arrel del servidor Apache.

Seguidament s'edita el fitxer de configuració que està en *private/config.php* i es dóna valors adequats a les variables de configuració:

- `$anmor`: ruta sencera on es troba l'analitzador morfològic *'lt-proc'*.
- `$dicos_path`: ruta on es troben els diccionaris finals i el binaris compilats de cada diccionari. Este directori ha de contindre un sub-directori per a cada diccionari amb el que pot treballar el formulari. El nom del subdirectori ha de seguir la següent sintaxi: `paradigmes-ll-rr` , on *ll* i *rr* són les inicials del parell de llengües del traductor en concret. En cada directori es deixen els diccionaris finals amb què treballa el traductor i els corresponents binaris compilats. Aquests directoris es poden substituir per enllaços simbòlics en cas que es troben en diferents llocs.
- `$usuaris_professionals`: un llistat amb els usuaris professionals del sistema que tenen permís per a inserir als diccionaris del formulari i validar paraules pendents de confirmació.
- `$mail`: Adreça electrònica de l'administrador encarregat dels formularis. Quan algú sol.licite donar-se d'alta com a usuari s'enviarà un correu electrònic a aquesta adreça.

Una vegada s'han configurat els paràmetres d'aquest fitxer el servidor de formularis ja es troba en funcionament.

3.2.2 Estructura de directoris

Tots els fitxers necessaris per al correcte funcionament s'estructuren de la següent forma:

- `/index.php`: presenta el formulari inicial d'inserció. Té un apartat per a cada parell de llengües on s'introdueix el lema en llengua origen i en llengua meta i permet triar el tipus de categoria gramatical que es tracta. En prémer el botó *'Go on'* avancem a la pàgina següent on seleccionarem els paradigmes de flexió corresponent a cadascun dels lemes en llengua origen i llengua meta.
- `/dics`: directori amb els diccionaris de les entrades que van inserint-se des dels formularis. Estan els fitxers amb les entrades dels usuaris no professionals (pendents de ser validades) i els diccionaris amb les entrades en format `XML` dels usuaris professionals
- `/private`: aquí està la major part dels mòduls utilitzats en els formularis. També conté els directoris amb la definició de paradigmes per a totes les llengües amb les que podem treballar, aquests directoris tenen la forma `paradigmes-ll-rr` sent *ll* i *rr* les inicials del parell de llengües del traductor en concret. L'ordre escollit per a les dues llengües, primer *ll* i després *rr* depèn de l'ordre en que s'hagen construït les entrades per

al diccionari bilingüe. També estan en aquest directori els fitxers encarregats de fer tot el processament de les paraules a inserir. Aquests son:

- `resultado.php`: Aquest *php* és cridat quan s'insereixen dues paraules de qualsevol parell de llengües des del mòdul *index.php*. Bàsicament el que fa és establir el parell de llengües amb què es treballarà (*$LR* i *$RL*) i la categoria gramatical de la paraula que s'inserirà (*$tipus*). S'inclou al mòdul *selec.php* que serà el següent que es crida en el procés d'inserció. En el cas que el *tipus* de la paraula a inserir fos una unitat multi-paraula (*Multi Word Verb*) aleshores és el mòdul *multip.php* el que s'inclou i es crida en comptes de *selec.php*. Els elements *Multi Word Verb* o unitats multi-paraula estan formats per un verb que pot flexionar seguit d'una cua invariable d'una o més paraules.
- `selecc.php`: És el mòdul encarregat de proporcionar la selecció del paradigma escollit per a cada una de les dues paraules, la paraula en llengua origen i en llengua meta. Proporciona una llista de paradigmes per a escollir que depenen del tipus gramatical de l'entrada que estem creant. Quan es selecciona un nou paradigma per a un dels dos lemes, proporciona uns exemples de la flexió del lema a inserir segons el paradigma que hem escollit. Si acceptem els paradigmes escollits, el mòdul crida a *insertarPro.php* o *insertar.php* depenen de si l'usuari que està fent la nova inserció està considerat com a professional o no professional respectivament.
- `multip.php`: Té la mateixa funció que el mòdul *selecc.php* però per a les unitats multi-paraula. Utilitza les mateixes variables i realitza les mateixes operacions, però quan mostra els exemples ho fa flexionant el verb i afegint totes les paraules que formen la cua. La manera de funcionar del mòdul és anàloga a la del mòdul *selecc.php* i la seua descripció detallada la trobarem a l'apartat 3.2.3.
- `valida.php`: Aquest mòdul es crida quan un usuari professional decideix validar paraules que estan en la cua d'entrades pendents de validació. Llegeix el fitxer amb la cua de paraules per validar i el recorre un a un, agafa les dades de l'entrada corresponent (*LRlem, RLlem, paradigmaLR, paradigmaRL, LR, RL*, etc...) i crida a *selecc.php* per a continuar amb el procés d'inserció d'eixa entrada particular.
- `insertarPro.php`: Aquest mòdul es crida quan ja hem seleccionat el paradigma adequat per a la paraula en llengua origen i en llengua meta (operació realitzada en *selecc.php*) i volem visualitzar com quedarà l'entrada en el format `XML` dels tres diccionaris, monolingüe origen, bilingüe i monolingüe meta. Des d'aquesta pantalla es podrà modificar directament el codi i finalment acceptar la nova entrada o cancelar l'operació.
- `ins_multip.php`: Té la mateixa funció que *insertarPro.php* però està pensat per a les unitats multi-paraula, de manera que té un tractament especial amb l'entrada per a que el codi `XML` que s'insereix als diccionaris siga correcte.

- `insertar.php`: És el mòdul equivalent a *insertarPro.php* però quan l'operació està realitzant-la un usuari no professional. Les accions que es realitzen en este cas són molt més senzilles ja que l'única cosa que es fa és inserir en el fitxer de termes pendents de validar els lemes i els paradigmes escollits per l'usuari no professional, i queden allí fins que un usuari professional els valida.
- `verSemi.php`: Este mòdul té la funció de mostrar el fitxer amb les entrades inserides per usuaris no professionals i que estan pendents de ser validades. Pot servir com ajuda per als usuaris professionals que abans de començar a validar paraules poden comprovar quines estan en la cua per a validar. Per a cridar-lo apareix un enllaç en el formulari generat per *selec.php* des d'on es pot cridar aquesta funcionalitat.
- `paradigmas.xsl`: Full d'estils que s'utilitza per a generar els fitxers de paradigmes amb que treballen els mòduls del formulari. S'utilitza amb l'especificació dels paradigmes de cada llengua escrit en format `XML`. Aquest punt es tractarà més extensament en l'apartat *Fitxers de Paradigmes*.
- `creaparadigma.awk`: Fitxer `awk` que també s'utilitza per a generar els fitxers de paradigmes de treball.
- `gen_paradig.sh`: Script que es pot fer servir si volem que automàticament es generen els fitxers de paradigmes de tots els parells de llengües que tenim instal·lats.

L'especificació detallada de les tasques de cada mòdul es troba en les seccions corresponents.

3.2.3 Fitxers php

resultado.php

Depenent del valor de la variable `$nomtrad` actualitzada per *index.php* el mòdul assigna els valor adequats a `$LR` i `$RL` (el tipus de llengua origen i llengua meta respectivament). Seguidament, depenent de la categoria gramatical de la paraula que s'inserirà, assignem el valor adequat a la variable $tipus i cridem a *selec.php* o *multip.php* depenent de si es tracta d'una paraula simple o d'una unitat multi-paraula.

selecc.php

Aquest mòdul té la funció de triar el paradigma corresponent a les paraules que volem introduir. Es tria el paradigma de la paraula en llengua origen i en llengua meta.

Depenent de la categoria gramatical de la paraula que estem introduint, s'assignen uns determinats valors a unes variables que darrerament s'utilitzaran, aquestes són

- `cadFich`: categoria gramatical del lema.
- `show`: cadena que es mostra al formulari indicant la categoria gramatical de la paraula que està inserint-se.

- `tag`: cadena amb l'etiqueta `XML` d'eixida de l'analitzador morfològic per a aquesta categoria gramatical.
- `tagout`: cadena amb el codi `XML` que indica la categoria gramatical de la paraula. Esta cadena s'utilitzarà quan es compose l'entrada final `XML` que s'inserirà al diccionari.
- `nota`: cadena amb possibles comentaris a inserir en el codi `XML` de l'entrada.

Els formularis treballen amb 4 tipus de diccionaris:

- *Semiprofessionals*: Aquests contenen les paraules inserides des del formulari per usuaris no professionals i que estan pendents de validació. Acaben amb l'extensió *"semi.dic"*
- *Diccionaris del formulari*: Contenen les paraules que han sigut inserides des del formulari per usuaris professionals i també les que han estat validades des dels diccionaris semiprofessionals. Acaben amb l'extensió *"webform"*.
- *Finals*: Són els fitxers amb totes les entrades escrites en codi `XML` . Aquests fitxers són els que utilitza definitivament el traductor després de rebre un tractament adequat de compilació. Acaben amb l'extensió *"dix"*.
- *Finals compilats*: Després d'haver compilat els diccionaris finals ja poden ser utilitzats pels binaris del traductor. Acaben amb l'extensió *"bin"*

Aquests diccionaris són utilitzats pels formularis i hi ha variables que contenen les rutes on es poden trobar. També es donen valors a les variables que mantenen les rutes on s'han de buscar els fitxers auxiliars i de configuració:

- `path`: ruta per als diccionaris temporals.
- `fich_LR`: diccionari de la llengua origen de les paraules inserides mitjançant el formulari i que encara no són al diccionari final ni al diccionari compilat.
- `fich_RL`: diccionari de la llengua meta de les paraules inserides mitjançant el formulari i que encara no són al diccionari final ni al diccionari compilat.
- `fich_LRRL`: diccionari bilingüe de les paraules inserides mitjançant el formulari i que encara no són al diccionari final ni al diccionari compilat.
- `fich-semi`: entrades inserides mitjançant el formulari per usuaris no professionals i que estan pendents de validació.
- `path_paradigmasLR`: ruta dels fitxers amb els paradigmes de flexió per a llengua origen.
- `path_paradigmasRL`: ruta dels fitxers amb els paradigmes de flexió per a llengua meta.
- `anmor`: ruta de l'analitzador morfològic.

- `aut_LRRL`: ruta del binari morfològic de llengua origen a llengua meta.
- `aut_RLLR`: ruta del binari morfològic de llengua meta a llengua origen.

Seguidament s'insereix el codi html del que ha de fer depenent de l'acció seleccionada. Les accions que es produeixen per ordre seqüencial són les següents:

- Comprova que el lema en llengua origen a inserir no hi siga ja en els diccionaris de paraules inserides pel formulari. Si s'hagués cridat a `selecc.php` des de la pantalla de validació de paraules (`valida.php`) aleshores comprova que el lema no hi siga ja en el fitxer de paraules inserides per usuaris no professionals. També ho comprova en el diccionari complet final.
- Fa la mateixa comprovació corresponent per a la llengua meta.
- Seguidament s'escriu codi per a seleccionar restriccions del sentit de la traducció.
- Es defineixen una sèrie de funcions que s'utilitzaran en la generació d'exemples per als lemes quan triem el corresponent paradigma. Aquestes son:
  - `esVocalFuerte`
  - `esVocalDebil`
  - `esVocal`
  - `PosicioVocalTall`

  Aquestes funcions es descriuen més endavant en l'apartat corresponent a *insertarPro.php*.
- S'obri el fitxer de paradigmes per a mostrar una finestra de selecció desplegable amb els paradigmes que podem triar per al lema en llengua origen que estem tractant. Per a fer això s'ha de comprovar seqüencialment els paradigmes corresponents a la categoria gramatical del lema i comprovar que el paradigma es pot aplicar al lema corresponent.
- Seguidament es fa el mateix però per als paradigmes del lema en llengua meta.
- Amb els lemes i amb els paradigmes corresponents seleccionats, s'han de generar els exemples de com quedaria la flexió d'eixos lemes segons els paradigmes escollits. Per aconseguir això necessitem per una banda l'arrel del lema (`raiz_LR i raiz_RL`) i per altra les terminacions d'exemple per al paradigma escollit (`paradigma_LR i paradigma_RL`), terminacions que s'obtenen del fitxer de paradigmes. Finalment compon una cadena amb els exemples generats (`ejemplos_LR i ejemplos_RL`) i els mostra
- Si estem en aquesta pantalla perquè venim de validar paraules (`valida=1`) aleshores afegeix al formulari un botó per a eliminar l'entrada actual en cas que no volguérem validar-la i inserir-la.

- Si l'usuari que ha entrat en aquesta pantalla és un usuari reconegut com a professional, aleshores afegeix al formulari un botó per a que eixe usuari puga triar l'opció de validar paraules inserides per usuaris no professionals
- Finalment tenim el tractament de l'acció corresponent activada al polsar un dels botons de confirmació que es mostren a la part inferior del formulari. Si l'acció és *"Delete"* , cosa que sols pot ocórrer quan s'estan validant entrades, elimina l'entrada corresponent del fitxer amb les entrades d'usuaris no professionals. Sinó, l'acció és la de confirmar (botó *"Go on"*), i cridem al mòdul `insertarPro.php` o `insertar.php` depenent de si l'usuari es professional o no professional respectivament. Aquests mòduls s'encarreguen de fer l'inserció als diccionaris.

Quan s'ha inserit la paraula, es torna a mostrar la pàgina de `validar.php` o de `selecc.php` depenent de si havíem entrat des d'un procediment de validació (i aleshores *valida=1*) o des d'una inserció normal.

multip.php

El codi i el comportament d'aquest mòdul és el mateix que el de *selecc.php*. L'única diferència és que aquest està pensat per a tractar les unitats multiparaula, mentre que *selec.php* tracta la resta de unitats. Així, la diferència més evident es que tenim les variables `$LRcua` i `$RLcua` que contenen la cua invariable que va després de la part variable de la multi-paraula. Quan es mostren els exemples, a banda de mostrar la part variable flexionada segons el paradigma escollit, també es mostra un quadre de text modificable amb la cua invariable.

Quan es premi el botó de seguir amb l'inserció de l'entrada als diccionaris, es crida al mòdul *ins_multip.php* en comptes de *insertarPro.php*.

valida.php

Aquest mòdul és cridat quan un usuari professional polsa el botó *"validate pairs"*. El control passa a aquest mòdul que llegeix el diccionari d'entrades pendents de ser validades ($fichSemi) del parell de llengües corresponent. Aleshores s'entra en un bucle que recorre aquest fitxer i va llegint les entrades una a una. Amb la informació de l'entrada particular dóna valor a una sèrie de variables que seran utilitzades en els mòduls que faran el tractament posterior, com ara:

$LRlem	$RLlem
$paradigmaLR	$paradigmaRL
$direccions	$tipo
$comentarios	$user
$geneLR	$geneRL
$numLR	$numRL
$LR	$RL

Una vegada establerts els valors adequats per a aquestes variables es cedeix el control a *selec.php* que tracta l'entrada com si d'una inserció d'usuari professional es tractara. Després d'inserir les entrades als diccionaris mitjançant *insertarPro.php*, el flux torna a *valida.php* que continua amb la següent entrada per validar.

insertarPro.php

Una vegada introduïts els lemes i seleccionats els paradigmes corresponents en *selec.php*, aquest mòdul és l'encarregat de generar les corresponents entrades en codi `XML` i inserir-les als diccionaris monolingües i al bilingüe.

Realitza moltes operacions semblants a les realitzades en *selec.php*, com ara generar els exemples de la paraula flexionada. Així primerament dóna valor a `cadFich`, `show`, `tag`, `tagout`, `nota` depenent de la categoria gramatical (`$tipus`) de la paraula a inserir. Assigna rutes a les variables de ubicació de fitxers i defineix unes funcions necessàries de igual manera a com ocorria en *selec.php*.

- `esVocalFuerte`: Retorna *true* si la vocal és forta, és a dir *a, e, o*.
- `esVocalDebil`: Retorna *true* si la vocal és feble, és a dir *a, e, o*.
- `esVocal`: Retorna *true* si el caràcter que li passem com argument és una vocal.
- `diptongo`: Retorna *true* si les dues lletres que se li passen com arguments formen diftong. Això ocorrerà sempre que les dues vocals no siguin fortes.
- `acentuar`: Se li passa una cadena de text i l'accentua seguint les regles del castellà en funció del parametre *$siguienteletra*.
- `esMayuscula`: Retorna *true* si el caràcter està en majúscula.
- `TieneAcento`: Retorna *true* si la cadena té accent.
- `acentua`: Accentua l'última vocal accentuable d'una paraula amb un accent tancat o obert depenent del sentit indicat en el paràmetre $sentit.
- `PonQuitaAcento`: Posa o lleva l'accent de la primera cadena passada com argument en funció de si la segona cadena argument té o no té accent.
- `PosicioVocalTall`: Torna la posició dins del lema ($lema) on es troba la vocal ($vocal) que separa l'arrel de la desinència. Busquem la vocal del final cap al principi i tornem la primera ocurrència de $vocal.

Ara es realitzen les mateixes operacions que en *selec.php*, comprova que l'entrada no hi siga ja als diccionaris i genera els exemples de la paraula flexionada amb el paradigma prèviament escollit. Seguidament el que fem és construir la cadena amb el codi `XML` que va a inserirem al diccionari monolingüe de llengua origen. Amb la informació que disposem dels lemes provinent de *selec.php*, es genera una cadena de text (`$cad_LR`) que conté el codi `XML` per al diccionari monolingüe. Aquesta cadena es mostra en una finestra de text que pot ser modificada manualment. El mateix procés es repeteix per a generar la cadena corresponent al diccionari monolingüe de llengua meta (`$cad_RL`) i per al diccionari bilingüe (`$cad_bil`). Seguidament, es concatena a aquestes variables de cadena els comentaris i el nom d'usuari que insereix l'entrada, si s'escau. Finalment s'acaba de compondre la pantalla de formulari, amb botons d'acceptar, eliminar i tornar cap enrere. El codi que fa el tractament de cadascuna de les possibles accions es troba al final del fitxer:

- `Inserir:` En aquest cas, fa unes substitucions de caràcters perquè l'entrada tinga el format adequat per als diccionaris, i insereix les cadenes `$cad_LR, $cad_bil, $cad_RL` als diccionaris monolingüe origen, bilingüe i monolingüe meta respectivament (`$fich_LR, $fich_LRRL, $fich_RL`). Si es produeix algun tipus d'error en inserir l'entrada informa d'aquest fet amb una notificació per pantalla. Si *insertarPro.php* havia estat cridat des d'un procediment de validació de paraules (*$valida=1*), aleshores insereix un botó "*Continue*"per a seguir validant paraules. En cas contrari insereix un botó de tancar la finestra que ens permetrà acabar.

- `Eliminar:` En aquest cas, elimina l'entrada del fitxer d'entrades pendents de validar.

ins_multip.php

Realitza les mateixes operacions que *insertarPro.php* però quan l'entrada a inserir és una unitat multi-paraula. La diferencia principal és que ara disposem de dues variables addicionals `$LRcua` i `$RLcua` que contenen la part invariable de la multi-paraula. Quan s'insereix l'entrada als diccionaris s'ha d'afegir aquesta cua al lloc convenient i convertir els espais en blanc per l'etiqueta de blanc `<b/>`.

insertar.php

La funció d'aquest mòdul és molt senzilla. Compon una cadena de text amb la informació provinent de *selec.php* separada per tabuladors. Aquesta cadena conté tota la informació necessària per a generar una entrada al diccionari:

```
$LRlem.$RLlem.$paradigmaLR.$direccion.$paradigmaRL.
$tipo.$comentarios.$user.$geneLR.$geneRL.
```

Aquesta entrada es guarda en un fitxer ($fichSemi) que conté la cua amb les entrades pendents de validar inserides per usuaris no professionals. Quan un usuari considerat professional vulga validar entrades pendents, el mòdul *valida.php* llegirà aquest mateix fitxer.

verSemi.php

S'encarrega de mostrar per pantalla el fitxer d'entrades pendents de validació, i ho fa així: llegeix el fitxer amb les entrades (*$fichSemi*) i entra en un bucle que recorre totes les entrades que es troben al fitxer. Per a cadascuna d'elles mostra un línia amb la següent informació:

```
$LRlem $paradigmaLR $direccion $RLlem
$paradigmaRL $tipo $comentarios
```

3.2.4 Fitxers de diccionari

Els fitxers amb les entrades inserides des de el formulari es troben en `/dics`. Ací es troben dos tipus de fitxers:

- `apertium-ll-rr.xx.webform`: El fitxer que conté entrades en codi `XML` preparades per a ser copiades en els diccionaris finals. El fitxer te el format mostrat, sent `ll-rr` les inicials corresponents al traductor al

que fa referència eixe fitxer i `xx` les inicials a la llengua del monolingüe al que fa referència o les inicials del bilingüe. Per exemple, les inicials del traductor espanyol-català són `es-ca`. Per aquest traductor tenim el monolingüe de espanyol (`es`), el de català (`ca`) i el bilingüe (`es-ca`) Així, en aquest directori tindrem els següents fitxers per al traductor espanyol-català:

```
apertium-es-ca.es.webform apertium-es-ca.ca.webform
            apertium-es-ca.es-ca.webform
```

- `oo-mm.semi.dic`: El fitxer que conté les entrades pendents de ser validades d'un determinat traductor. `oo-mm` són les inicials corresponents al traductor concret. Per exemple, el traductor espanyol-català tindrà aquest fitxer per al semi-professional: `es-ca.semi.dic`

3.2.5 Fitxers de paradigmes

Els paradigmes utilitzats per a cada parell de llengües s'especifiquen en dos arxius de format `XML` anomenats `paradig.ll-rr.xx.xml` on `xx` són les inicials corresponents a una llengua i `ll-rr` les inicials corresponents a un parell de llengües. Aquests arxius estan formats per un conjunt d'entrades corresponents a paradigmes o models de flexió de les paraules d'una llengüa en concret. El fitxer `XML` es composa de les següents parts:

- Capçalera/Arrel del document d'especificació.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="paradigmas.xsl"?>
<!DOCTYPE form SYSTEM "form.dtd">
<form lang="oc" langpair="oc-ca">
```

A l'atribut *lang* estan les inicials de la llengua per a la qual estan especificantse paradigmes i a l'atribut *langpair* estan les inicials corresponents al parell de llengües del traductor a què correspon el fitxer. Cal que al mateix directori on estan els fitxers de paradigmes estiga `form.dtd` on s'especifica la DTD que segueixen aquests fitxers. Aquesta DTD està detallada a l'anexe A.5.

- Un conjunt d'elements que defineixen els paradigmes. Per a explicar el format dels elements s'utilitzarà el següent exemple:

```
<entry PoS="adj" nbr="sg_pl" gen="mf">
        <endings>
                <stem>amable</stem>
                <ending/>
                <ending>s</ending>
        </endings>
        <paradigms howmany="1">
                <par n="amable__adj"/>
```

```
        </paradigms>
</entry>
```

Cada paradigma està especificat en un element `<entry>`. Aquest element pot tenir tres atributs:

- *PoS*: la categoria gramatical del paradigma. Pot prendre els valors: acr, adj, adv, noun, pname, pr, verb. Aquest és obligatori per a qualsevol categoria gramatical.
- *nbr*: els nombres admesos pel paradigma. Pot tenir els valors: sg, pl, sg_pl, sp.
- *gen*: els gèneres admesos pel paradigma. Pot tenir els valors: m, f, m f, mf.

A més, està format per dos elements:

- `endings`: l'arrel i les terminacions utilitzades per a seleccionar el paradigma al formulari i mostrar els exemples de flexió.
- `paradigms`: especificació del paradigma o paradigmes que defineixen la flexió seguida per una entrada en concret. Necessita de l'atribut *howmany* que indica el nombre de paradigmes que utilitza l'entrada. Per a cadascun dels paradigmes utilitzats hi ha una línia que especifica el nom del paradigma del diccionari, i que té el següent format:

  ```
  <par n="long__adj"/>
  ```

A partir del fitxer de paradigmes escrit en `XML`, s'han de generar els fitxers amb els que directament treballen els mòduls dels formularis. Si s'executa el script `/private/gen_paradig.sh` el procés es realitza automàticament per a tots els parells de llengües de què disposem:

```
#  cd private
#  ./gen_paradig.sh
```

Per a inserir un nou paradigma als formularis, cal preparar l'entrada corresponent al fitxer de paradigmes en format `XML` i després actualitzar els arxius de treball amb l'ordre anterior.

El procés automatitzat també es pot realitzar de forma manual si no volem actualitzar els fitxers de tots els parells de llengües que tenim instal·lats. La generació manual dels arxius de treball, es fa amb un full d'estil `XSL` mitjançant aquesta ordre:

```
#  xsltproc paradigmas.xsl fitxer_de_paradigmes.xml
                                    | ./creaparadig.awk
```

Aquesta acció genera un fitxer de treball per a cada categoria gramatical. Els fitxers generats es troben en els directoris `/private/paradigmas.ll-rr`. Cada directori conté fitxers amb els paradigmes que es poden utilitzar per a cada parell de llengües `ll-rr` i per a cada component gramatical. Cadascun d'aquests directori conté els fitxers següents:

- `paradigacr_xx`: paradigmes per acrònims de la llengua `xx`.

- `paradigadj_xx`: paradigmes per adjectius de la llengua `xx`.
- `paradigadv_xx`: paradigmes per adverbis de la llengua `xx`.
- `paradigcnjadv_xx`: paradigmes per a conjuncions adverbials de la llengua `xx`.
- `paradigcnjcoo_xx`: paradigmes per a conjuncions copulatives de la llengua `xx`.
- `paradigcnjsub_xx`: paradigmes per a conjuncions subordinades de la llengua `xx`.
- `paradignoun_xx`: paradigmes per a noms de la llengua `xx`.
- `paradigpname_xx`: paradigmes per a noms propis de la llengua `xx`.
- `paradigpr_xx`: paradigmes per a preposicions de la llengua `xx`.
- `paradigverb_xx`: paradigmes per a verbs de la llengua `xx`.

Els fitxers estan composts per una entrada per línia. Cada entrada conté la informació següent:

exemples nombre de paradigmes paradigmes_model (nombres) (gèneres)

El separador utilitzat per a les diferents parts de l'entrada es el tabulador.

- *Exemples*: són les terminacions que s'utilitzaran per a generar els exemples quan escollim aquest paradigma com a prototip per a una determinada paraula a inserir.
- *Nombre de paradigmes*: és el nombre de paradigmes del diccionari que s'utilitzaran per a flexionar adequadament aquest model de flexió en particular.
- *Paradigmes model*: és el nom del paradigma o paradigmes que estan al diccionari i que són els que s'utilitzaran per a flexionar la nova entrada.
- *(Nombres)*: Només es posa en el cas de noms, adjectius i acrònims. Fa referència al nombre del paradigma.
- *(Gèneres)*: Només es posa en el cas de noms, adjectius i acrònims. Fa referència al gènere del paradigma.

Així per exemple, al traductor espanyol-català tindríem el directori `/private-/paradigmas.es-ca` on trobaríem dos fitxers `XML`: `paradig.es-ca.es.xml` i `paradig.es-ca.ca.xml` amb l'especificació dels paradigmes utilitzats en cada llengua. A partir d'aquests fitxers es poden generar tots els fitxers de paradigmes per un parell de llengües concret amb l'ordre:

```
#  cd private/paradigmas.es-ca
#  xsltproc ../paradigmas.xsl paradig.es-ca.es.xml
                                  | ../creaparadig.awk
#  xsltproc ../paradigmas.xsl paradig.es-ca.ca.xml
                                  | ../creaparadig.awk
```

O bé, es poden generar automàticament per a tots els parells de llengües amb:

```
# ./private/gen_paradig.sh
```

Entre els fitxers de treball generats hi hauria un anomenat `paradigverb_ca` que conté els possibles paradigmes de verbs per al català, on una possible línia seria la següent:

```
abra/çar /ço /ci 1 abalan/çar__vblex
```

que deriva de l'entrada en `XML`:

```
<entry PoS="verb">
        <endings>
                <stem>abra</stem>
                <ending>çar</ending>
                <ending>ço</ending>
                <ending>ci</ending>
        </endings>
        <paradigms howmany="1">
                <par n="abalan/çar__vblex"/>
        </paradigms>
</entry>
```

3.3 Utilització dels formularis

3.3.1 Introducció

Quan un usuari vulga inserir noves entrades a un diccionari ha de connectar-se mitjançant un client navegador a l'adreça on s'haja instal·lat el servidor de formularis, per exemple:

```
http://xixona.dlsi.ua.es/forms
```

S'obrirà una pàgina web amb el portal d'entrada a `Opentrad Apertium-Insertion Form`. En el marc de l'esquerra hi ha enllaços per a obtenir més *informació* , *descarregar* els programes i *contacte* per a sol·licitar registrar-se com a usuari del sistema. Per a registrar-se com a usuari cal enviar un correu electrònic al webmaster encarregat dels formularis.

Per a inserir noves paraules, s'omplirà el formulari amb les dades corresponents i polsarem el botó *'Go On'*; en aquest moment cal autentificar-se com a usuari registrat o no deixarà continuar amb el procediment d'inserir paraules. Quan un usuari es registra pot ser registrat de dues formes, com a usuari professional o com a usuari no professional. Ambdós tenen unes habilitacions diferents i que s'explicaran en apartats següents.

3.3.2 Inserció d'entrades

Usuari professional

Quan vulguem inserir una nova entrada als diccionaris cal que ens dirigim a l'apartat corresponent al parell de llengües que vulguem ampliar. Aleshores

especifiquem el lema en llengua origen i en llengua meta i triem el tipus de categoria gramatical de les entrades que inserirem. Seguidament, polsem el botó *'Go on'* per a seguir amb el procés d'inserció.

En aquest punt apareix una nova finestra on mostra els lemes i uns paràmetres que caracteritzen la nova entrada al diccionari. Si l'entrada ja hi és en algun dels diccionaris, s'avisarà a l'usuari i el sistema automàticament escollirà sentit unidireccional (d'esquerra a dreta o a l'inrevés). Si no és en cap, triarà doble sentit de traducció. En aquesta nova finestra podem fer estes tres accions, si s'escau:

- Escollir el paradigma corresponent per al lema en llengua origen no).[1]
- Escollir la direcció de traducció de l'entrada si és diferent de la proposada automàticament.
- Afegir comentaris en l'entrada que quedaran al diccionari final.

Una vegada hem fet les operacions corresponents, polsem el botó *'Go on'* si volem confirmar l'entrada o *'Close'* si senzillament volem cancelar l'operació d'inserció.

La següent i última pantalla que se'ns mostra conté les tres entrades en format `XML` generades per als diccionaris monolingüe llengua origen, bilingüe i monolingüe llengua meta. Aquestes entrades es mostren en tres finestres de text i es poden editar per si cal fer alguna modificació. Una vegada revisades les entrades, polsarem el botó *'Insert'* per a inserir-les definitivament als diccionaris corresponents. També tenim la possibilitat de polsar el botó *'Go back'* en cas que volguérem tornar al pas anterior.

Usuari no professional

Quan un usuari entra al sistema d'inserció com a usuari no professional, el procediment d'inserció de paraules és el mateix que per a l'usuari professional, però les entrades inserides no es guarden als diccionaris generats pels formularis sinó que queden en una cua d'entrades pendents de validació. Per inserir les paraules d'aquesta cua als diccionaris cal que un usuari professional les valide.

3.3.3 Validació d'entrades pendents d'aprovació

Els usuaris professionals, en la pantalla on s'escull el paradigma dels lemes inserits, tenen dos enllaços addicionals:

- *See pairs to be validated*: Ací obrirem una finestra on podrem veure el contingut del fitxer amb les paraules que estan pendents de validació; paraules que hauran estat inserides per usuaris no professionals. Aquesta finestra és merament informativa i es tanca polsant el botó *'Close'*.

[1] Escollir el paradigma és un procés amb el que s'ha d'anar amb molta cura. Cal triar el paradigma exacte que descriu el comportament del lema que estem inserint. En el cas d'adjectius, noms i acrònims, cal escollir el paradigma que s'ajusti a la flexió desitjada i també als gèneres admesos. En el cas d'acrònims cal fixar-se en el gènere i nombre que admeteix cada paradigma dels que podem escollir; per exemple BBC serveix per a acrònims femenins singulars i SA per a acrònims femenins que admeteixen plural). En el cas de noms propis cal escollir el paradigma corresponent depenent de si es tracta d'un nom propi de cosa (un diari), de persona o de lloc.

- *Validate pairs*: Aquesta opció permet a un usuari professional validar una a una les entrades pendents. Quan polsem este botó, s'obri la finestra de selecció de paradigmes que s'ha vist en l'apartat d'inserció de noves paraules. Esta finestra es mostra amb les dades que havia escollit l'usuari que havia afegit eixa entrada. Ara, l'usuari professional podrà modificar els lemes, esborrar l'entrada (botó *'Delete'*) o seguir amb el procés d'inserció. Si segueix amb l'inserció de l'entrada, el procés és el mateix que en una inserció normal, sols que quan l'entrada queda definitivament afegida als diccionaris de formulari el control torna a mostrar la següent entrada a validar de la cua

  Aquest procés és repeteix fins que es validen totes les paraules de la cua o finalitzem el procés en polsar el botó *'Close'*.

Apèndix A

Definicions de tipus de document (DTD) en XML

[Les DTDs de les seccions A.1, A.2 i A.3 substitueixen la DTD del transfer d'Apertium. De la mateixa manera, la DTD del diccionari que es presenta en A.4 substitueix l'antiga DTD del diccionari d'Apertium.]

A.1 DTD del mòdul de transferència estructural (prechunk)

DTD per al format de les regles de transferència estructural. Aquesta definició ve amb el paquet `apertium` (última versió) que es pot descarregar de `http://www.sourceforge.net`. La descripció dels diferents elements es troba en l'apartat 1.6.

```
<!ENTITY % condition "(and|or|not|equal|begins-with|
                       ends-with|contains-substring|in)">
<!ENTITY % container "(var|clip)">
<!ENTITY % sentence "(let|out|choose|modify-case|
                      call-macro|append)">
<!ENTITY % value "(b|clip|lit|lit-tag|var|get-case-from|
                   case-of|concat)">
<!ENTITY % stringvalue "(clip|lit|var|get-case-from|
                         case-of)">

<!ELEMENT transfer (section-def-cats,
                    section-def-attrs,
                    section-def-vars,
                    section-def-lists?,
                    section-def-macros?,
                    section-rules)>

<!ATTLIST transfer default (lu|chunk) #IMPLIED>

<!ELEMENT section-def-cats (def-cat+)>
```

```
<!ELEMENT def-cat (cat-item+)>
<!ATTLIST def-cat n ID #REQUIRED>

<!ELEMENT cat-item EMPTY>
<!ATTLIST cat-item lemma CDATA #IMPLIED
                   tags CDATA #REQUIRED >

<!ELEMENT section-def-attrs (def-attr+)>

<!ELEMENT def-attr (attr-item+)>
<!ATTLIST def-attr n ID #REQUIRED>

<!ELEMENT attr-item EMPTY>
<!ATTLIST attr-item tags CDATA #IMPLIED>

<!ELEMENT section-def-vars (def-var+)>

<!ELEMENT def-var EMPTY>
<!ATTLIST def-var n ID #REQUIRED>

<!ELEMENT section-def-lists (def-list)+>

<!ELEMENT def-list (list-item+)>
<!ATTLIST def-list n ID #REQUIRED>

<!ELEMENT list-item EMPTY>
<!ATTLIST list-item v CDATA #REQUIRED>

<!ELEMENT section-def-macros (def-macro)+>

<!ELEMENT def-macro (%sentence;)+>
<!ATTLIST def-macro n ID #REQUIRED>
<!ATTLIST def-macro npar CDATA #REQUIRED>

<!ELEMENT section-rules (rule+)>

<!ELEMENT rule (pattern, action)>
<!ATTLIST rule comment CDATA #IMPLIED>

<!ELEMENT pattern (pattern-item+)>

<!ELEMENT pattern-item EMPTY>
<!ATTLIST pattern-item n IDREF #REQUIRED>

<!ELEMENT action (%sentence;)*>

<!ELEMENT choose (when+,otherwise?)>

<!ELEMENT when (test,(%sentence;)*)>

<!ELEMENT otherwise (%sentence;)+>

<!ELEMENT test (%condition;)+>
```

```
<!ELEMENT and ((%condition;),(%condition;)+)>

<!ELEMENT or ((%condition;),(%condition;)+)>

<!ELEMENT not (%condition;)>

<!ELEMENT equal (%value;,%value;)>
<!ATTLIST equal caseless (no|yes) #IMPLIED>

<!ELEMENT begins-with (%value;,%value;)>
<!ATTLIST begins-with caseless (no|yes) #IMPLIED>

<!ELEMENT ends-with (%value;,%value;)>
<!ATTLIST ends-with caseless (no|yes) #IMPLIED>

<!ELEMENT contains-substring (%value;,%value;)>
<!ATTLIST contains-substring caseless (no|yes) #IMPLIED>

<!ELEMENT in (%value;, list)>
<!ATTLIST in caseless (no|yes) #IMPLIED>

<!ELEMENT list EMPTY>
<!ATTLIST list n IDREF #REQUIRED>

<!ELEMENT let (%container;, %value;)>

<!ELEMENT append (%value;)+>
<!ATTLIST append n IDREF #REQUIRED>

<!ELEMENT out (mlu|lu|b|chunk)+>

<!ELEMENT modify-case (%container;, %stringvalue;)>

<!ELEMENT call-macro (with-param)*>
<!ATTLIST call-macro n IDREF #REQUIRED>

<!ELEMENT with-param EMPTY>
<!ATTLIST with-param pos CDATA #REQUIRED>

<!ELEMENT clip EMPTY>
<!ATTLIST clip pos CDATA #REQUIRED
               side (sl|tl) #REQUIRED
               part CDATA #REQUIRED
               queue CDATA #IMPLIED
               link-to CDATA #IMPLIED>

<!ELEMENT lit EMPTY>
<!ATTLIST lit v CDATA #REQUIRED>

<!ELEMENT lit-tag EMPTY>
<!ATTLIST lit-tag v CDATA #REQUIRED>

<!ELEMENT var EMPTY>
<!ATTLIST var n IDREF #REQUIRED>
```

```
<!ELEMENT get-case-from (clip|lit|var)>
<!ATTLIST get-case-from pos CDATA #REQUIRED>

<!ELEMENT case-of EMPTY>
<!ATTLIST case-of pos CDATA #REQUIRED
                  side (sl|tl) #REQUIRED
                  part CDATA #REQUIRED>

<!ELEMENT concat (%value;)+>

<!ELEMENT mlu (lu+)>

<!ELEMENT lu (%value;)+>

<!ELEMENT chunk (tags,(mlu|lu|b)+)>
<!ATTLIST chunk name CDATA #IMPLIED
                namefrom CDATA #IMPLIED
                case CDATA #IMPLIED>

<!ELEMENT tags (tag+)>
<!ELEMENT tag (%value;)>

<!ELEMENT b EMPTY>
<!ATTLIST b pos CDATA #IMPLIED>
```

A.2 DTD del mòdul interchunk

DTD per al format de les regles de transferència estructural del mòdul interchunk. Aquesta definició ve amb el paquet `apertium` (última versió) que es pot descarregar de `http://www.sourceforge.net`. La descripció dels diferents elements es troba en l'apartat 1.6.

```
<!ENTITY % condition "(and|or|not|equal|begins-with|
                       ends-with|contains-substring|in)">
<!ENTITY % container "(var|clip)">
<!ENTITY % sentence "(let|out|choose|modify-case|
                      call-macro|append)">
<!ENTITY % value "(b|clip|lit|lit-tag|var|get-case-from|
                   case-of|concat)">
<!ENTITY % stringvalue "(clip|lit|var|get-case-from|
                         case-of)">

<!ELEMENT interchunk (section-def-cats,
                      section-def-attrs,
                      section-def-vars,
                      section-def-lists?,
                      section-def-macros?,
                      section-rules)>

<!ELEMENT section-def-cats (def-cat+)>

<!ELEMENT def-cat (cat-item+)>
<!ATTLIST def-cat n ID #REQUIRED>

<!ELEMENT cat-item EMPTY>
<!ATTLIST cat-item lemma CDATA #IMPLIED
                   tags CDATA #REQUIRED >

<!ELEMENT section-def-attrs (def-attr+)>

<!ELEMENT def-attr (attr-item+)>
<!ATTLIST def-attr n ID #REQUIRED>

<!ELEMENT attr-item EMPTY>
<!ATTLIST attr-item tags CDATA #IMPLIED>

<!ELEMENT section-def-vars (def-var+)>

<!ELEMENT def-var EMPTY>
<!ATTLIST def-var n ID #REQUIRED>

<!ELEMENT section-def-lists (def-list)+>

<!ELEMENT def-list (list-item+)>
<!ATTLIST def-list n ID #REQUIRED>

<!ELEMENT list-item EMPTY>
<!ATTLIST list-item v CDATA #REQUIRED>
```

```
<!ELEMENT section-def-macros (def-macro)+>

<!ELEMENT def-macro (%sentence;)+>
<!ATTLIST def-macro n ID #REQUIRED>
<!ATTLIST def-macro npar CDATA #REQUIRED>

<!ELEMENT section-rules (rule+)>

<!ELEMENT rule (pattern, action)>
<!ATTLIST rule comment CDATA #IMPLIED>

<!ELEMENT pattern (pattern-item+)>

<!ELEMENT pattern-item EMPTY>
<!ATTLIST pattern-item n IDREF #REQUIRED>

<!ELEMENT action (%sentence;)*>

<!ELEMENT choose (when+,otherwise?)>

<!ELEMENT when (test,(%sentence;)*)>

<!ELEMENT otherwise (%sentence;)+>

<!ELEMENT test (%condition;)+>

<!ELEMENT and ((%condition;),(%condition;)+)>

<!ELEMENT or ((%condition;),(%condition;)+)>

<!ELEMENT not (%condition;)>

<!ELEMENT equal (%value;,%value;)>
<!ATTLIST equal caseless (no|yes) #IMPLIED>

<!ELEMENT begins-with (%value;,%value;)>
<!ATTLIST begins-with caseless (no|yes) #IMPLIED>

<!ELEMENT ends-with (%value;,%value;)>
<!ATTLIST ends-with caseless (no|yes) #IMPLIED>

<!ELEMENT contains-substring (%value;,%value;)>
<!ATTLIST contains-substring caseless (no|yes) #IMPLIED>

<!ELEMENT in (%value;, list)>
<!ATTLIST in caseless (no|yes) #IMPLIED>

<!ELEMENT list EMPTY>
<!ATTLIST list n IDREF #REQUIRED>

<!ELEMENT let (%container;, %value;)>

<!ELEMENT append (%value;)+>
```

```
<!ATTLIST append n IDREF #REQUIRED>

<!ELEMENT out (b|chunk)+>

<!ELEMENT modify-case (%container;, %stringvalue;)>

<!ELEMENT call-macro (with-param)*>
<!ATTLIST call-macro n IDREF #REQUIRED>

<!ELEMENT with-param EMPTY>
<!ATTLIST with-param pos CDATA #REQUIRED>

<!ELEMENT clip EMPTY>
<!ATTLIST clip pos CDATA #REQUIRED
               part CDATA #REQUIRED>

<!ELEMENT lit EMPTY>
<!ATTLIST lit v CDATA #REQUIRED>

<!ELEMENT lit-tag EMPTY>
<!ATTLIST lit-tag v CDATA #REQUIRED>

<!ELEMENT var EMPTY>
<!ATTLIST var n IDREF #REQUIRED>

<!ELEMENT get-case-from (clip|lit|var)>
<!ATTLIST get-case-from pos CDATA #REQUIRED>

<!ELEMENT case-of EMPTY>
<!ATTLIST case-of pos CDATA #REQUIRED
                  part CDATA #REQUIRED>

<!ELEMENT concat (%value;)+>

<!ELEMENT chunk (%value;)+>

<!ELEMENT pseudolemma (%value;)>

<!ELEMENT b EMPTY>
<!ATTLIST b pos CDATA #IMPLIED>
```

A.3 DTD del mòdul postchunk

DTD per al format de les regles de transferència estructural del mòdul postchunk. Aquesta definició ve amb el paquet `apertium` (última versió) que es pot descarregar de `http://www.sourceforge.net`. La descripció dels diferents elements es troba en l'apartat 1.6.

```
<!ENTITY % condition "(and|or|not|equal|begins-with|
                      ends-with|contains-substring|in)">
<!ENTITY % container "(var|clip)">
<!ENTITY % sentence "(let|out|choose|modify-case|
                     call-macro|append)">
<!ENTITY % value "(b|clip|lit|lit-tag|var|get-case-from|
                   case-of|concat)">
<!ENTITY % stringvalue "(clip|lit|var|get-case-from|
                         case-of)">

<!ELEMENT postchunk (section-def-cats,
                     section-def-attrs,
                     section-def-vars,
                     section-def-lists?,
                     section-def-macros?,
                     section-rules)>

<!ELEMENT section-def-cats (def-cat+)>

<!ELEMENT def-cat (cat-item+)>
<!ATTLIST def-cat n ID #REQUIRED>

<!ELEMENT cat-item EMPTY>
<!ATTLIST cat-item name CDATA #REQUIRED>

<!ELEMENT section-def-attrs (def-attr+)>

<!ELEMENT def-attr (attr-item+)>
<!ATTLIST def-attr n ID #REQUIRED>

<!ELEMENT attr-item EMPTY>
<!ATTLIST attr-item tags CDATA #IMPLIED>

<!ELEMENT section-def-vars (def-var+)>

<!ELEMENT def-var EMPTY>
<!ATTLIST def-var n ID #REQUIRED>

<!ELEMENT section-def-lists (def-list)+>

<!ELEMENT def-list (list-item+)>
<!ATTLIST def-list n ID #REQUIRED>

<!ELEMENT list-item EMPTY>
<!ATTLIST list-item v CDATA #REQUIRED>

<!ELEMENT section-def-macros (def-macro)+>
```

```
<!ELEMENT def-macro (%sentence;)+>
<!ATTLIST def-macro n ID #REQUIRED>
<!ATTLIST def-macro npar CDATA #REQUIRED>

<!ELEMENT section-rules (rule+)>

<!ELEMENT rule (pattern, action)>
<!ATTLIST rule comment CDATA #IMPLIED>

<!ELEMENT pattern (pattern-item+)>

<!ELEMENT pattern-item EMPTY>
<!ATTLIST pattern-item n IDREF #REQUIRED>

<!ELEMENT action (%sentence;)*>

<!ELEMENT choose (when+,otherwise?)>

<!ELEMENT when (test,(%sentence;)*)>

<!ELEMENT otherwise (%sentence;)+>

<!ELEMENT test (%condition;)+>

<!ELEMENT and ((%condition;),(%condition;)+)>

<!ELEMENT or ((%condition;),(%condition;)+)>

<!ELEMENT not (%condition;)>

<!ELEMENT equal (%value;,%value;)>
<!ATTLIST equal caseless (no|yes) #IMPLIED>

<!ELEMENT begins-with (%value;,%value;)>
<!ATTLIST begins-with caseless (no|yes) #IMPLIED>

<!ELEMENT ends-with (%value;,%value;)>
<!ATTLIST ends-with caseless (no|yes) #IMPLIED>

<!ELEMENT contains-substring (%value;,%value;)>
<!ATTLIST contains-substring caseless (no|yes) #IMPLIED>

<!ELEMENT in (%value;, list)>
<!ATTLIST in caseless (no|yes) #IMPLIED>

<!ELEMENT list EMPTY>
<!ATTLIST list n IDREF #REQUIRED>

<!ELEMENT let (%container;, %value;)>

<!ELEMENT append (%value;)+>
<!ATTLIST append n IDREF #REQUIRED>
```

```
<!ELEMENT out (b|lu|mlu)+>

<!ELEMENT modify-case (%container;, %stringvalue;)>

<!ELEMENT call-macro (with-param)*>
<!ATTLIST call-macro n IDREF #REQUIRED>

<!ELEMENT with-param EMPTY>
<!ATTLIST with-param pos CDATA #REQUIRED>

<!ELEMENT clip EMPTY>
<!ATTLIST clip pos CDATA #REQUIRED
               part CDATA #REQUIRED>

<!ELEMENT lit EMPTY>
<!ATTLIST lit v CDATA #REQUIRED>

<!ELEMENT lit-tag EMPTY>
<!ATTLIST lit-tag v CDATA #REQUIRED>

<!ELEMENT var EMPTY>
<!ATTLIST var n IDREF #REQUIRED>

<!ELEMENT get-case-from (clip|lit|var)>
<!ATTLIST get-case-from pos CDATA #REQUIRED>

<!ELEMENT case-of EMPTY>
<!ATTLIST case-of pos CDATA #REQUIRED
                  part CDATA #REQUIRED>

<!ELEMENT concat (%value;)+>

<!ELEMENT mlu (lu+)>

<!ELEMENT lu (%value;)+>

<!ELEMENT b EMPTY>
<!ATTLIST b pos CDATA #IMPLIED>
```

A.4 DTD del format dels diccionaris

DTD per al format dels diccionari. Aquesta definició ve amb el paquet `apertium` i amb el paquet `lttoolbox`.

```
<!ELEMENT dictionary (alphabet?, sdefs?,
      pardefs?, section+)>

<!ELEMENT alphabet (#PCDATA)>

<!ELEMENT sdefs (sdef+)>

<!ELEMENT sdef EMPTY>
<!ATTLIST sdef n ID #REQUIRED>

<!ELEMENT pardefs (pardef+)>

<!ELEMENT pardef (e+)>
<!ATTLIST pardef n CDATA #REQUIRED>

<!ELEMENT section (e+)>

<!ATTLIST section id ID #REQUIRED
                  type (standard|inconditional|postblank) #REQUIRED>

<!ELEMENT e (i | p | par | re)+>
<!ATTLIST e r (LR|RL) #IMPLIED
            lm CDATA #IMPLIED
            a CDATA #IMPLIED
            c CDATA #IMPLIED

<!ELEMENT par EMPTY>
<!ATTLIST par n CDATA #REQUIRED>

<!ELEMENT i (#PCDATA | b | s | g | j | a)*>

<!ELEMENT re (#PCDATA)>

<!ELEMENT p (l, r)>

<!ELEMENT l (#PCDATA | a | b | g | j | s)*>

<!ELEMENT r (#PCDATA | a | b | g | j | s)*>

<!ELEMENT a EMPTY>

<!ELEMENT b EMPTY>

<!ELEMENT g (#PCDATA | a | b | j | s)*>
<!ATTLIST g i CDATA #IMPLIED>

<!ELEMENT j EMPTY>

<!ELEMENT s EMPTY>
```

```
<!ATTLIST s n IDREF #REQUIRED>
```

A.5 DTD dels paradigmes del formulari

DTD per al format dels fitxers de paradigmes que utilitza l'eina de formularis. Aquesta definició ve amb el paquet `apertium-lexical-webform`.

```
<!ELEMENT form (entry)+>

<!ATTLIST form
        lang CDATA #REQUIRED
        langpair CDATA #REQUIRED>

<!ELEMENT entry (endings, paradigms)+>

<!ATTLIST entry
        PoS CDATA #REQUIRED
        nbr CDATA #IMPLIED
        gen CDATA #IMPLIED>

<!ELEMENT endings (stem, ending+)>

<!ELEMENT stem (#PCDATA)>

<!ELEMENT ending (#PCDATA)>

<!ELEMENT paradigms (par+)>

<!ATTLIST paradigms howmany CDATA #REQUIRED>

<!ELEMENT par EMPTY>

<!ATTLIST par n CDATA #REQUIRED>
```