The Integration of a Part-of-Speech Tagger into the ALEP Platform[1]

Thierry Declerck
DFKI GmbH
(German Research Center for Artificial Intelligence)
Stuhlsatzenhausweg 3
D-66123 Saarbruecken (Germany)
declerck@dfki.uni-sb.de

Heinz Dieter Maas
IAI
(Institute for Applied Information Science)
Martin-Luther-Str 14
D-66123 Saarbruecken (Germany)
maas@iai.uni-sb.de

Abstract: We describe how part-of-speech information delivered by a tagger (the `mpro` tool) has been integrated into the ALEP (Advanced Language Engineering Platform) system. For this we extended an approach described within the LS-GRAM project, which consisted in defining the Text Handling component of ALEP in such a way that so-called "messy details" are handled within this subsystem, hence keeping the (linguistic) parser free from such tasks. We just extended the tagging strategy used for this purpose to normal words and modified the default tagging of words proposed by the ALEP system in order to incorporate information delivered by the part-of-speech tagger. The resulting tagging is converted by means of "lift" rules into partial linguistic descriptions, which provide the direct input to the grammatical analysis. We show that this procedure substantially reduces the parse times of the system.

1 Background

The starting point of our work has been the treatment of so-called 'messy details' within the ALEP platform[2]. *Messy details* are text constructs which do not lend themselves well to treatment by traditional techniques for linguistic analysis, whence their 'messiness'. Typical examples are numbers, codes or other (sequences of) word-forms which can occur in many variations, making impossible a comprehensive treatment by traditional means.

Word level phenomena are usually the most frequently occurring messy details, including such things as dates, numbers and proper names. For any realistic NLP application these types of constructs must be processed efficiently, the alternative being coding them individually in some lexicon and/or implementing sets of grammar rules for parsing them syntactically.

This problem area was given priority in the LS-GRAM project[3] which aimed to integrate an approach to messy details into a large-scale grammar implementation. How we dealt with this phenomena has been described in 2 short papers[4].

2 Preprocessing in ALEP

2.1 The Default Text Handling Component of ALEP

The ALEP platform provides the user with a Text Handling (TH) component which allows a "preprocessing" of inputs. Input texts will first go through a processing chain consisting in a SGML-based tagging of their elements. The default setup of the system defines the following processing chain: the text is first converted to an EDIF (Eurotra Document Interchange Format) format. Then a subsequent recognition process is provided: paragraph recognition, sentence recognition and word recognition. The

---

[1] We would like to thank the anonymous reviewers for their precious comments on our contributions for this workshop.
[2] See [ASD], [AUG] and [Alshawi et al. (1991)].
[3] The LS-GRAM (Large-Scale GRAMmars for EC languages) project, was funded by the CEC, number LRE 61029.
[4] See [Bredenkamp et al. (1996)] and [Schmidt et al. (1996)].

output of those processes consists in the tagging of the recognized elements: <P> for paragraphs, <S> for sentences, <W> for words (in case of morphological analysis, the tag <M> is provided for morphemes) and <PT> for punctuation signs. Some specialized features are also provided for the tagged words, allowing to characterize them more precisely, so for example <ACRO> for acronyms or <D> for digits and so on.

The simple ASCII input string "John loves Mary.", after going through the recognition process of the default TH component, will take the form:

```
<P><S> <W>John</W><W>loves</W><W>Mary</W><PT>.</PT></S></P>
```

<P> and </P> mark the beginning and the respective ending of the recognized paragraph structure. The same remark is valid for the other tags and their corresponding structures. The complete output of the TH subsystem for the word level is as follows:

```
<W TYPE="WORD" CASE="FIRST">john</W>
```

Here we know that we deal with a word beginning with uppercase and having as data content the string "john". In the default case, it is this information which is the input to the TH-LS component (Text-Handling to Linguistic Structure) of the system. Within this subsystem one has to specify so-called 'tsls' (text structure to linguistic structure) rules, which transform the TH output into partial linguistic descriptions[5]. The syntax of these rules is as follows:

```
tsls_rule( <ld>, <tag_name>, [<features>], <tag_content> )
```

where:
<ld> is a (partial) Linguistic Description (LD). <tag_name> is the name of a SGML tag (e.g. <S>, <W>). <features> is a list of feature-value descriptions of the tag's features. <tag_content> is the atomic content of the string within the tag (optional). This kind of mapping rule allows the flow of information between text structures and (partial) linguistic descriptions. As shown in figure 1 – a tsls rule for the word level – this flow of information is in case of the default TH processing just a mapping between the tag <W> (with an empty list of feature-value descriptions[6]) and some information described within the linguistic type *ld*. And this is exactly the place where we can integrate richer linguistic information provided by part-of-speech (PoS) taggers, as can be seen in section 3.3.

$$\begin{array}{l} \text{ts\_ls\_rule(} \\ \quad {}_{ld}\left[\begin{array}{ll} \text{SPEC} \mid \text{LEVEL\_SPEC} \mid \text{M\_W} & \text{yes} \\ \text{SIGN} \mid \ldots \mid \text{SYN} \mid \text{CONSTYPE} & {}_{phrasal}\left[\begin{array}{ll} \text{MIN} & \text{yes} \\ \text{CONSTR} & \text{(w\_form;lexical)} \end{array}\right] \end{array}\right], \\ \text{W, [ ]).} \end{array}$$

Figure 1: *Correspondence established between the partial linguistic description and the* <W> *markup of text*

2.2 Extensions of the Default Text Handling Component

The TH component of the ALEP platform also foresees the use of the tag <USR> in case the text is tagged by user-defined programs. In the context of LS-GRAM grammars such user-defined taggers have been implemented in both `awk` and `Perl` and have been named `tagit`. Within the `tagit` programs patterns have been defined in order to match relevant *messy details*, so for example patterns for currency expressions (occuring quite often in the economical newspapers analyzed by the LS-GRAM grammars).

When `tagit` matches a pattern against the input, the matched string is replaced with an appropriate <USR> tag. Thus an expression like "Dreiundvierzig Milliarden Dollar" is matched by the pattern `currency_measure` and is replaced by the SGML tag:

[5] This partial linguistic description will then provide the direct input to the linguistic parser.

[6] The values of the features TYPE and CASE are not playing a role here.

```
<USR LEVEL=M TYPE=CURRENCY_MEASURE VAL="Dreiundvierzig Milliarden Dollar">
 Dreiundvierzig_Milliarden_Dollar</USR>
```

Once the <USR> tags are inserted, the output of the TH component is processed by a specialized tsls rule within the set of lift rules, as shown in figure 2:

$$
\begin{array}{l}
\text{ts\_ls\_rule(} \\
{}_{ld}\left[\begin{array}{ll} \text{SPEC} \mid \text{USR\_TYPE} & \boxed{1} \\ \text{SIGN} \mid \text{STRING} \mid \text{FIRST} & \left\langle \boxed{2} \mid \_ \right\rangle \end{array}\right], \\
\texttt{USR},\ [\ \texttt{TYPE=}\boxed{1}\ ]\ ,\ \boxed{2}\ ).
\end{array}
$$

Figure 2: *The general tsls rule doing the conversion for all* <USR> *tags*

In this case, we have a more complex flow of information between text structure and (partial) linguistic description. The list of feature-value descriptions of the tag <USR> is not empty (it contains information delivered by the user-defined taggers). So a value-sharing can be defined between the text feature TYPE and the linguistic feature USER_TYPE. The feature USER_TYPE stays for the name of a generic lexicon entry, in our example: "currency_measure" (see figure 3). The feature STRING of the generic entry will be determined by co-sharing with the data-content of the text structure (e.g. "dreiundvierzig_milliarden_dollar" where the three items of the original string are grouped to one unit).

$$
{}_{ld}\left[\begin{array}{ll} \text{SPEC} \mid \text{USR\_TYPE} & \texttt{CURRENCY\_MEASURE} \\ \text{SIGN} \mid \ldots \mid \text{CAT} & {}_{cat}\left[\begin{array}{ll} \text{HEAD} & {}_{n}\left[\text{AGR} \quad (\text{pl\&p3})\right] \\ \text{SUBCAT} & \langle\rangle \end{array}\right] \end{array}\right]
$$

Figure 3: *Generic lexical entry for currency expressions*

The definition of such generic entries in the lexicon allows to keep the lexicon smaller but also to deal with a potentially infinite number of words. This strategy has been described in a paper where the treatment of other phenomena (also for other languages) is documented [Bredenkamp et al. (1996)][7].

3 Accessing results from an independent morpho-syntactical analysis

Once (positive) experiences have been made with the integration of small user-defined taggers, it seemed to be straightforward to try to integrate information resulting from the processing of the input texts by an independent (and real) tagger. We decided to base this experiment on the `mpro` tool, which is briefly presented in the next section[8]. And in the section 3.3 we will describe how the PoS information resulting from the analysis of `mpro` has been integrated in ALEP.

3.1 The `mpro` Tool

The `mpro` tool can deal with free input texts. Looks up in lexicons, which are continually updated, a complete morphology and distinct heuristics allow to provide with an accurate annotation of all items

[7] See also [Schmidt et al. (1996)].

[8] For a detailed description of the tool, see [Maas (1996)] and [Maas and Hirschfeld (1996)]. An important remark: for our experiment, we used this tool as a black box, being here interested only in its output.

of the input string. For the text tagging, morphological analysis is combined with a look-up in a word-form dictionary especially designed for this purpose. Nouns, adjectives, verbs and derived adverbs are recognized by morphological analysis, and all other words must be listed in the tagging dictionary. The morpheme lexicon of the morphological analyzer contains approx. 30 000 lexemes (36 500 allomorphs). The format of the tagging dictionary is as follows:

```
{string=Wordform,c=w,sc=CAT,lu=Citationform,...}
```

where `CAT` is the category. Depending on its value several other features may be present, e.g. 'fu' (syntactic function), 's' (semantic feature) or 'case' and 'nb' (number) etc. If values are ambiguous, they can be represented as a disjunction, e.g. 'case=nom;acc'. The order of the features is irrelevant, except that 'string' must appear in the first position. Some examples of lexical entries:

```
{lu=hier,lex=hier,cat=adv,funct=pp,sem=loc}.
{lu=heute,cat=adv,lu=heute,lex=heute,sem=temp}.
```

If a word-form cannot be identified – neither by morphology nor by looking up the tagging dictionary – the system tries to find some minimal information. Sequences of digits become cardinal numbers (additionally classified as 'year', 'month', 'day', 'integer'), sequences of capitals will have the category 'acronym'. An unknown word whose first letter is a capital letter is considered a name ('sc=n,sem=name'). This is correct in most cases. A capital letter followed by a full stop is classified as a name. Identified proper names are then appended to specialized lexicon files. The analyzer recognizes as well fixed expressions like 'in Bezug auf' or 'de facto' and abbreviations like 'usw.' and 'etc.'. For each word of the text the tagger produces at least one description and writes it on an output file in a format similar to the format of the dictionary entries:

{ori=Wordform,lu=LU,sc=CAT,snr=Snr,wnra=WT,wnrr=WS, Otherinfo}

The variables used in this term stand for the following:

- Wordform: Form of word as occurring in the text.
- LU: The citation form of the word-form (infinitive for verbs, nominative singular for nouns, etc.)
- CAT: The category of the word.
- Snr: The sentence number of the word-form.
- WT: The number of the word in the text.
- WS: The number of the word in the sentence.

Each feature of 'Otherinfo' has the form '<property> = <value>'. The features of 'Otherinfo' depend on the value of the category CAT, e.g.

- fu: (syntactic) sub-specification.
- s: (semantic) sub-specification.
- case: case information (especially for nouns).
- nb: number (nouns, finite verbs)
- g: gender (nouns)
- tns: tense (for finite verbs)
- per: person (for finite verbs)
- The features 't', 'ts', 'ds' and 'cs' describe the word structure (derivation, compounding, prefixation). 'w' shows the number of elements in a compound word.

All the a/v pairs can be used for search procedures and inspection programs. And these a/v pairs have been inspected in order to extract the information we want to pass to the partial linguistic description of ALEP (remember: this partial linguistic description is the direct input to the parsers of ALEP).

3.2 Output of the `mpro` analysis

As an example, the output of the `mpro` analysis for the following sentence:
Große Bereiche der Dasa leiden unter dem Rückgang des einst lukrativen Rüstungsgeschäfts. (Large areas of the DASA are suffering from the decline of the once lucrative arms trade.)

```
<1> to <14>:
c=satz,r=339}
  {c=np,r=302,tr=genattr}
     {c=np,r=380}
        {ori=Grosse,wnra=1,wnrr=1,snr=1,pctr=no,pctl=no,last=no,gra=cap,c=adj,lu=gross,
         endung=e,deg=base,t=gross,cs=a,ts=gross,ds=gross,ss=a,w=1}
        {ori=Bereiche,wnra=2,wnrr=2,snr=1,pctr=no,pctl=no,last=no,gra=cap,c=noun,nb=plu,
         case=nom;gen;acc,lu=bereich,g=m,s=domain,t=bereich,cs=n,ts=bereich,ds=bereich,
         ss=domain,w=1}
        {c=np,r=212,tr=d_n,nb=sg,g=f,case=gen}
           {ori=der,wnra=3,wnrr=3,snr=1,string=der,lu=d-,c=w,sc=art,fu=def;np,p=dE:r,
            s_flex=weak,spec=def,pctr=no,last=no,pctl=no,gra=small,case=gen,nb=sg,g=f}
           {ori=DASA,wnra=4,wnrr=4,snr=1,pctr=no,pctl=no,last=no,gra=caps,c=noun,lu=dasa,
            s=coll&isto,t=dasa,cs=n,ts=dasa,ds=dasa,ss=coll&isto,w=1,case=gen,nb=sg,g=f}
  {c=hs,r=311}
     {ori=leiden,wnra=5,wnrr=5,snr=1,pctr=no,pctl=no,last=no,gra=small,c=verb,vtyp=fiv,
      tns=pres,nb=plu,per=3,lu=leiden,t=leiden,cs=v,ts=leiden,ds=leiden,ss=v,w=1}
  {c=pp,r=223,tr=genattr}
     {c=pp,r=215,tr=d_n,case=dat,nb=sg,g=m}
        {ori=unter,wnra=6,wnrr=6,snr=1,string=unter,lu=unter,c=w,sc=p,pcomp=no,pctr=no,
         last=no,pctl=no,gra=small,case=dat,nb=sg,g=m}
        {c=np,r=212,tr=d_n,case=dat,nb=sg,g=m}
           {ori=dem,wnra=7,wnrr=7,snr=1,string=dem,lu=d-,c=w,sc=art,fu=def;np,p=d":m,
                pctr=no,last=no,pctl=no,gra=small,case=dat,nb=sg,g=m}
           {ori=Rueckgang,wnra=8,wnrr=8,snr=1,pctr=no,pctl=no,last=no,gra=cap,c=noun,
            lu=rueckgang,s=process,t=rueckgang,cs=n,ts=rueckgang,ds=zurueck_$gehen~IRREG,
            ss=process,w=1,case=dat,nb=sg,g=m}
               {c=np,r=203,tr=d_a_n,case=gen,nb=sg,g=n}
                  {ori=des,wnra=9,wnrr=9,snr=1,string=des,lu=d-,c=w,sc=art,fu=def,p=dEs,
                     pctr=no,last=no,pctl=no,gra=small,case=gen,nb=sg,g=n}
                  {c=adj,r=124,case=gen,nb=sg,g=n}
                     {ori=einst,wnra=10,wnrr=10,snr=1,string=einst,lu=einst,c=w,sc=adv,
                      fu=pp,s=temp,pctr=no,last=no,pctl=no,gra=small}
                     {ori=lukrativen,wnra=11,wnrr=11,snr=1,pctr=no,pctl=no,last=no,gra=small,
                        c=adj,lu=lukrativ,endung=en,deg=base,s=va,t=lukrativ,cs=a,ts=lukrativ,
                        ds=lukrieren~ativ,ss=va,w=1}
                     {ori=Ruestungsgeschaefts,wnra=12,wnrr=12,snr=1,pctr=yes,pctl=no,last=no,
                        gra=cap,c=noun,lu=ruestungsgeschaeft,t=ruestung#geschaeft,cs=n#n,
                        ts=ru1stungs#geschaeft,ds=ruesten~ung#geschaeft,ss=ation#n,w=2,
                        case=gen,nb=sg,g=n}
  {ori=.,wnra=13,wnrr=13,snr=1,string=.,lu=stop,c=w,sc=punct,p=____,pctr=no,last=yes,pctl=no,
   gra=other}
```

We shouldn't look at too much details here, but just notice, that the features 'c', 'cs' and 'sc' are giving us part-of-speech information about the items.

Since we were actually only interested in the PoS information delivered by the tagger, we wrote a small function extracting this information from the output of `mpro` and put it into a list, which has the following form:

```
STRING,CAT,STEM
Grosse,a,gross
Bereiche,n,bereich
...
```

3.3 Integration of Part-of-Speech Information into Linguistic Analysis

As we saw in section 2.1 the annotation of the words provided by the TH component of ALEP is a minimal one. And this is quite normal, if one considers that the input of the TH component is a pure ASCII text. We also saw in section 2.2 that the list of feature-value descriptions of the tag `<USR>` can be enriched by the user. And what about the words of the input string which have not been marked by the `<USR>` tag? Can the list of feature-value descriptions of the tag `<W>` be enriched by the user? We experimented on this and the answer was positive.

The list described at the end of section 3.1 is accessed by the user-defined tagger (`tagit`) which produces an output of the TH component enriched with the PoS information provided by the `mpro` tool[9]:

```
<W TYPE="WORD" CASE="FIRST" CATEGORY="a">grosse</W>
<W TYPE="WORD" CASE="FIRST"  CATEGORY="n">bereiche</W>
...
```

where our user-defined tagger just added the user-defined feature CATEGORY to the list of the feature-value of the tag `<W>`.[10] The `tagit` program also converts the labels of the categories used in the tagger to the ones used in the grammar[11]. In order to pass this additional information to the (partial) linguistic description, an extended tsls rule is necessary:

$$
\begin{array}{l}
\text{ts\_ls\_rule(} \\
{}_{ld}\left[\begin{array}{ll}
\text{SPEC} \mid \text{LEVEL\_SPEC} \mid \text{M\_W} & \text{yes} \\
\text{SIGN} \mid \ldots \mid \text{SYN} & {}_{syn}\left[\begin{array}{ll}
\text{CONSTYPE} & {}_{phrasal}\left[\begin{array}{ll} \text{MIN} & \text{yes} \\ \text{CONSTR} & \text{(w\_form;lexical)} \end{array}\right] \\
\text{CATEGORY} \mid \text{CAT} & \boxed{1}
\end{array}\right]
\end{array}\right], \\
\texttt{W, [CATEGORY=}\boxed{1}\texttt{]).}
\end{array}
$$

Figure 4: *Value-sharing of a feature of the text structure tag* `<W>` *and the* CAT *feature of the linguistic description*

Sharing the value of both the `CATEGORY` feature of the text structure and of the feature 'cat' of the linguistic description is enough in order to allow the 'lifting' of the information delivered by the PoS tagger. The advantage of this strategy: before starting, the parser will process items already containing PoS information, which are typically a contextual information and normally not available to the parser.

By the way, it was possible to considerably reduce the parsing time of the system. It should suffice to consider the following table, comparing the parse time for the German sentence we introduced above: "Grosse Bereiche der Dasa leiden unter dem Rueckgang des einst lukrativen Ruestungsgeschaefts.".

[9]As far as we can judge, the disambiguation provided by `mpro` is quite accurate. But in case we still have ambiguities in the output of the tagger, we have to take this into account: all readings will be passed to the linguistic parser.

[10]For this step we didn't modify the Document Type Definition of the TH component of ALEP. But for sure, we are interested in respecting conventions for the SGML-marking. This will be done as far as possible in accordance with the TEI guidelines [Sperberg and Burnard (1994)].

[11]Thus our approach can be used in conjunction with arbitrary taggers. The small `tagit` program will ensure the conversion of labels and categories. We are also assuming, for the purpose of the experiment, that the PoS information delivered by `mpro` is correct. This can be problematic, as has been noticed by a reviewer of this paper, and we are aware of this. This point will be the topic of another paper, concerned more directly with the accuracy of PoS tagging.

Two distinct parsers have been used, the "basic" and the "record" parser[12]. One should look at the figures (CPU time) staying under the "total" column. The results of the "basic" parser are displayed in each case first.
Test results for grammar without POS information

	total	BaWordSeg	BaLift	BaAna	Refine
sol : 1	29.840	0.120	0.010	29.710	0.000
allsols	40.550	0.190	0.020	40.340	0.000

	total	RWordSeg	RLiftAna	Refine
sol : 1	6.250	0.200	6.050	0.000
allsols	6.260	0.200	6.060	0.000

Test results for grammar with POS information

	total	BaWordSeg	BaLift	BaAna	Refine
sol : 1	6.850	0.110	0.010	6.730	0.000
allsols	13.820	0.190	0.020	13.610	0.000

	total	RWordSeg	RLiftAna	Refine
sol : 1	4.850	0.190	4.660	0.000
allsols	4.860	0.190	4.670	0.000

The improvement of performance is partly due to the fact that after the segmentation of the input words has been achieved, the grammar is concerned first with the reconstruction of the words, enriched with linguistic information contained in the morpheme lexicon of the grammar. For this process affixes have been described as the parsing heads. And affixes (in German) being highly ambiguous (for example the affix *en* of the word *leiden* above), there are several entries for some of them in the morpheme lexicon, implying that the process of word construction will run several times, also when not necessary. But once the PoS is known before the process of word construction is started, the homograph affixes not corresponding to the PoS won't be considered any longer, thus reducing the search space for the parser. This remark being also valid for ambiguities at the word level.

So our strategy will be profitable everywhere where the input words are built with ambiguous affixes. In any case, the basic parser will be (considerably) faster, whereas the record parser (which already shows very good performances) will be improved only in case we have really strong ambiguities at both morpheme and word level).

Similar results have been obtained for other sentences: so for example (1) "In den Wochen vor Weihnachten konnte der stolze Vorsitzende der DASA ein gutes Jahresergebnis verkünden." and (2) "Das Luft- und Raumfahrtunternehmen hat das Jahr 1991 mit einem Gewinn von vierhundert Millionen Mark wiederholt." (this sentence gets more than one syntactic analysis, due to pp attachment, which resolution is postponed to the refinement processing step.). Both sentences have been tested on a slower machine and the results (total) are:

```
without PoS information
basic parser:  91.201 (1)   record parser: 17.384  (1)
              106.583 (2)                  22.049  (2)
with PoS information:
basic parser:  43066  (1)   record parser: 17.384  (1)
               65.733 (2)                  21.936  (2)
```

We can observe here again the substantial improvement of the performance of the basic parser. We observe also that the record parser remains, for those examples, insensitive to the integration of PoS. We still have to investigate more precisely in which cases the performances of the record parser can be improved.

[12]Both parsers are non-deterministic bottom-up head-out parsers, but the record is more efficient since it subsumes the common information of distinct alternatives.

4 Concluding Remarks

We experimented with the integration of the results of a PoS tagger. The first results are very promising, since the parse times of the ALEP system have been substantially improved. Further steps will consist in the full integration of the results of the tagger and the corresponding reorganization (simplification and reduction) of the lexicon of the grammar. So for example, since `mpro` is providing a complete morphological analysis, we could drop the Two Level Component implemented in the German LS-GRAM grammar and start the linguistic analysis of ALEP directly at the word level, getting rid of the whole subset of analysis rules concerned with word construction. We expect another significant improvement of parse times.

Once this has been done, we plan to describe lifting rules for more complex parts of speech, like the ones resulting from partial parsing, thus generalizing our approach to more complex syntactic units, which could also lead to a reorganization and simplification of the syntactic component of the grammar.

We understand our work also as a possible contribution to a discussion on hybrid approaches to NLP processing, thus integrating corpus-based knowledge and the methods of unification-based grammar formalisms. In the future we should try to define which parts of textual inputs should be processed by means of so-called preprocessing tools and what remains the task of the unification-based formalisms (ALEP being one example). The topic of the workshop and the length of this paper do not allow the development of such a discussion at this place.

References

[ASD] BIM/SEMA, *ALEP System Documentation.* 1993. CEC Luxembourg.

[AUG] Simpkins, N.K., Groenendijk, M., Cruickshank G. *ALEP–1 User Guide.* 1993. CEC Luxembourg.

[Alshawi et al. (1991)] Alshawi H., Arnold D. J., Backofen R., Carter D. M., Lindop J., Netter K., Pulman S., Tsuji J., Uszkoreit H. 1991. *Eurotra ET6/1: Rule Formalism and Virtual Machine Design Study (final report).* CEC Luxembourg.

[Badia et al. (1995)] Badia T., Bredenkamp A., Declerck T. Hentze R., Marimon M., Schmidt P., Theofilidis A. *LS-GRAM Rule Coding Manual, Deliverable D - WP7.* Version 1. CEC, 1995, Luxembourg.

[Bredenkamp et al. (1996)] Bredenkamp A., Declerck T., Fouvry F., Music B. 1996. *Efficient Integrated Tagging of Word Constructs.* COLING 96. pp. 1028–1031.

[Maas (1996)] Maas Heinz-Dieter. 1996. *MPRO – Ein System zur Analyse und Synthese deutscher Wörter.* In Hausser Roland, editor, *Linguistische Verifikation, Sprache und Information Nr. 34* Max Niemeyer Verlag, Tübingen

[Maas and Hirschfeld (1996)] Maas Heinz-Dieter, Hirschfeld Diane 1996. *Improving the Functionality of a Text-to-Speech System by Adding Morphological Knowledge..* In Görz Guenther and Hoelldoblerund Steffen, editors, *KI-96: Advances in Artificial Intelligence.* Springer (Lecture Notes in Artificial Intelligence 1137).

[Schmidt et al. (1996)] Schmidt P., Rieder S., Theofilidis A., Declerck T. 1996. *Lean formalisms, linguistic Theory and Applications. Grammar Development in ALEP.* COLING 96. pp. 286–291.

[Sperberg and Burnard (1994)] Sperberg-McQueen C.M., Burnard L. (editors) 1994. *Guidelines for Electronic Text Encoding and Exchange (TEI P3).* Chicago, Oxford.