A Corpus Representation Format for Linguistic Web Services: the D-SPIN Text Corpus Format and its Relationship with ISO Standards

Ulrich Heid[1], Helmut Schmid[1], Kerstin Eckart[1], Erhard Hinrichs[2]

[1]Universität Stuttgart, Institut für Maschinelle Sprachverarbeitung, [2]Universität Tübingen, Seminar für Sprachwissenschaft
Azenbergstr. 12, 70174 Stuttgart, Wilhelmstr. 19, 72074 Tübingen
heid,schmid,eckartkn@ims.uni-stuttgart.de, eh@sfs.uni-tuebingen.de

Abstract

In the framework of the preparation of linguistic web services for corpus processing, the need for a representation format was felt, which supports interoperability between different web services in a corpus processing pipeline, but also provides a well-defined interface to both, legacy tools and their data formats and upcoming international standards. We present the D-SPIN text corpus format, TCF, which was designed for this purpose. It is a stand-off XML format, inspired by the philosophy of the emerging standards LAF (Linguistic Annotation Framework) and its "instances" MAF for morpho-syntactic annotation and SynAF for syntactic annotation. Tools for the exchange with existing (best practice) formats are available, and a converter from MAF to TCF is being tested in spring 2010. We describe the usage scenario where TCF is embedded and the properties and architecture of TCF. We also give examples of TCF encoded data and describe the aspects of syntactic and semantic interoperability already addressed.

1. Layers of corpus formats for linguistic web services

1.1. Scenario – Requirements

Part of the work of the D-SPIN project[1] is devoted to the creation of web services for the linguistic annotation and exploration of corpus data. Several standard tools, provided by different project members, have been made available as web services: tokenizers, taggers and lemmatizers, a parser, tools for word frequency and collocation association calculation, etc. These tools are accessible to users via WebLicht[2], a tool chainer providing both infrastructural services and a GUI for combining the individual tools (cf. Hinrichs et al. (2009), Hinrichs et al. (2010)).

Using this scenario may involve the processing of substantial amounts of corpus data through the pipeline[3]. To support the interoperability of the corpus processing pipeline, i.e. the possibility to transmit a corpus from one annotation or exploration tool to the next, and to keep the management of the data flow between different tools efficient, corpus data entering the pipeline as well as being sent from one tool to the next need to be encoded in a common internal format (cf. section 2.). This format supports interoperability between the individual web services based on their requirements for input and output and allows for individual tools in the chain being replaced by others of the same level: e.g. chain one includes a statistical parser and chain two includes a rule based parser, while the other tools in the chain remain the same.

In addition, results of a given processing chain should be exchangeable with external tools and/or users and thus easy to reinterpret (section 3.). This includes the mapping to different formats and on that account syntactic and semantic interoperability aspects.

The two requirements of (i) efficiency for internal data management and (ii) explicitness for external data exchange seem at first sight to be conflicting. For internal purposes, a slim format for the representation of raw and/or annotated data is required, and for external exchange, a representation should be sought which is well-documented and easily convertible.

1.2. Architecture

The two requirements are in our view best satisfied by the use of three different but closely related formats. Each of the tools made available in the WebLicht chain uses its own internal format. As we do not want to change the original tools when inserting them in the chain, these formats need to be preserved; we design wrappers for each tool, to ensure clear interfaces.

The tool chain infrastructure requires a slim format for the transport of corpus data between tools. For this purpose we designed TCF, the *D-SPIN Text Corpus Format*. The wrapper developed for each individual tool maps data between the tool format and TCF. As TCF is a slim stand-off XML format, it is more efficient than other more verbose XML formats. If needed the efficiency could be further increased by transmitting the data in a binary encoding format[4].

For external exchange purposes, we intend to use the up-

[1]D-SPIN stands for Deutsche SPrachressourcen-INfrastruktur; the D-SPIN project is financed by the German Federal Ministry of Research and Education, BMBF and coordinated by University of Tübingen; it is a national German complement to the EU-project CLARIN. See the URLs `http://www.d-spin.org` and `http://www.clarin.eu` for details.

[2]cf. the (password-protected) web site: `http://clarin.sfs.uni-tuebingen.de:8080/WebLicht1/`

[3]We have successfully run experiments with a 10 million word corpus.

[4]e.g. EXI - Efficient XML Interchange Format: `http://www.w3.org/XML/EXI/` or ISO/IEC 24824-1:2007 Information technology – Generic applications of ASN.1: Fast infoset.

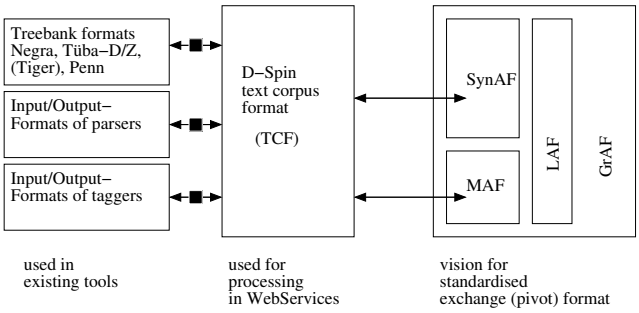

Figure 1: Architecture for D-SPIN formats and their interrelationships

coming international corpus annotation standard LAF (*Linguistic Annotation Framework*, ISO/DIS 24612 (2009)) and its related specifications for the linguistic layers of morphosyntax (MAF, *Morphosyntactic Annotation Framework*, ISO/DIS 24611 (2008)) and syntax (SynAF, *Syntactic Annotation Framework*, ISO/DIS 24615 (2009)). We are in the process of creating converters between TCF and the versions of these formats available and documented as of spring 2010 (see below, section 3.4.4.).

Moreover, since we want to keep the D-SPIN tool chains compatible with existing tools and data formats from outside D-SPIN, TCF also should be convertible to/from such legacy formats as e.g. the TiGerXML format[5]. Figure 1 schematically represents our approach.

2. A text corpus format for the D-SPIN web services

In this section, we present the principles of TCF, the experimental D-SPIN text corpus format. As tool building and format definition work is still ongoing, we describe the version of spring 2010.

The TCF is a slim stand-off XML format. TCF contains a header specifying character encoding and the version of the D-SPIN architecture used. We furthermore foresee a section for document related metadata. Here, conformity with the CLARIN Metadata Initiative (CMDI)[6] will be ensured; to date, the metadata part of TCF has not yet been fully worked out.

In the tradition of stand-off formats, the object language data are encoded in different sections, for the corpus text, its tokens, as well as annotations at token and at sentence level. Nevertheless TCF can deal with a sort of inline annotation at least on the token layer, by including the full forms into the token element. Figure 2 gives an overview of the structure of TCF.

The example given in figure 2 is taken from a text which has been tokenized, tagged, lemmatized and parsed (with the BitPar parser, Schmid (2004)). Where appropriate, information about the tagset used for each annotation is given.

```
<?xml version="1.0" encoding="UTF-8"?>
<D-Spin xmlns="http://www.d-spin.org/data"
     version="0.3">
  <MetaData
       xmlns="http://www.d-spin.org/data/metadata"/>
  <TextCorpus
       xmlns="http://www.d-spin.org/data/textcorpus"
       lang="de">
    <text>Charles Perrault,Das Rotkaeppchen.
    ...
    </text>
    <tokens>                        <!-- token layer -->
      <token ID="t2">Charles</token>
      <token ID="t3">Perrault</token>
      ...
      <token ID="t1736">.</token>
    </tokens>
    <POStags tagset="STTS">    <!-- part-of-speech -->
      <tag tokID="t2">NE</tag>
      ...
      <tag tokID="t1736">\$.</tag>
    </POStags>
    <lemmas>                  <!-- lemma annotation -->
      <lemma tokID="t2">Charles</lemma>
      ...
      <lemma tokID="t1736">.</lemma>
    </lemmas>
                          <!-- (constituent) parse -->
    <parsing tagset="TigerTB">
      <parse><constituent ... </parse>
    </parsing>
  </TextCorpus>
</D-Spin>
```

Figure 2: Overview of TCF structure

```
<constituent cat="NP-TOP">
  <constituent cat="PN-NK-Nom.Sg">
    <constituent cat="NE-PNC-Nom.Sg">
      <tokenRef tokID="t2"/>
    </constituent>
    <constituent cat="NE-PNC-Nom.Sg">
      <tokenRef tokID="t3"/>
    </constituent>
  </constituent>
...
```

Figure 3: Constituent layer of TCF-annotated text

In the example in figure 2, the STTS tagset[7] has been used for tagging, and TiGer Treebank[8] annotations for parsing. Each layer of annotation can contain identifiers of the respective linguistic objects. In our example in figure 2, tokens are annotated with the 'ID' attribute. These IDs are made reference to at the level of e.g. part-of-speech and lemma annotations (stand-off). Figure 3 shows constituents of the parsing output, which are in this example according to the TiGer Treebank annotated with their category (e.g. *NP* for a noun phrase) as well as the annotations of their token components (e.g. *PNC-Nom.Sg* for a proper noun component in nominative case, singular) in case of terminal nodes (words in the example: *Charles Perrault*).

In fact, TCF assumes one basic tokenization underlying the chain of corpus processing; concurrent tokenization has so far been disregarded.

3. Exchanging data with TCF

3.1. Interoperability of Web Services

Web services must "speak the same language" in order to be successfully combined in a processing pipeline. D-Spin

[5] http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERSearch/doc/html/TigerXML.html

[6] http://www.clarin.eu

[7] http://www.ims.uni-stuttgart.de/projekte/corplex/TagSets/stts-table.html

[8] http://www.ims.uni-stuttgart.de/projekte/TIGER/

uses the TCF format as the common language. However, a common language is not sufficient to guarantee successful communication within a pipeline. It is also necessary to ensure that each web service receives all the information that it needs from the preceding process in the pipeline. D-SPIN uses a registry for this purpose which stores information about the input and output specifications of individual tool web services. The I/O specification of an English POS tagger which uses the Penn treebank tagset and returns lemma and POS information could be represented as follows:

```
<TextCorpus lang="en"/>                <TextCorpus>
 <tokens/>                    ⇒        <POStags tagset="PennTB"/>
</TextCorpus>                           <lemmas/>
                                      </TextCorpus>
```

The web service description specifies the annotation layers required in the input (here *tokens* and the document *lang*uage) and those added in the output (here *POStags* with a *tagset* attribute set to *PennTB* and *lemmas*). By means of the registry, the WebLicht user interface is able to determine whether a sequence of web services can be successfully chained or not, and which annotation layers will be present in the output of the pipeline.

3.2. Interoperability between data formats

Data exchange involves issues of both syntactic and semantic interoperability. By syntactic interoperability we mean the mapping of the structure of a given annotation format to another, e.g. from an inline format to an stand-off format. Semantic interoperability concerns the mapping of data categories, such as e.g. tagset mapping. Syntactic interoperability can be achieved by reformatting. Semantic interoperability requires a reinterpretation of data categories and, where possible, either their ontological anchoring or devices to describe commonalities and differences between data categories which belong to different tagsets (cf. Witt et al. (2009): interlingua-like vs. transfer-like models of interoperability).

Our current experiments have been limited to aspects of syntactic interoperability, but we are working e.g. on registering the STTS part-of-speech tagset in the ISOcat[9] data registry (cf. ISO 12620:2009), to achieve semantic interoperability at the POS level. The individual tagsets, which are used by the tools can be mapped to data categories in the ISOcat data registry where equal or equivalent data categories are available. In case there is no equal data category available, the local data categories can be related to existing categories as eg. being subsumed ("subs") by a data category in ISOcat, or a new data category in ISOcat can be defined.

3.3. Exchange with existing tools

Wrappers interface existing tools to the D-Spin tool chains and map data between the tool-specific formats and TCF. The implementation of these wrappers is quite simple. Writing wrappers for tokenizers, POS taggers and parsers only required between 60 and 160 lines of Perl code[10].

[9] http://www.isocat.org/

[10] http://www.perl.org/

We also designed converters for non-D-SPIN resources, such as the Tüba-D/Z treebank[11] and data encoded in the PAULA format (cf. Dipper (2005)). This allows us, among others, to use the D-SPIN processing tools in combination with the Tüba-D/Z treebank. As TCF is quite flexible, full conversions between TCF and the format of Tüba-D/Z as well as between TCF and the PAULA format can be achieved.

3.4. Compatibility with international representation formats

To ensure sustainability and a possibility to exchange D-SPIN-generated data with external tools and users, we are developing converters between TCF and the LAF family of upcoming international corpus representation standards[12].

3.4.1. LAF and related formats

The medium-term view of international standardization work on corpus annotation foresees both meta-standards for corpus representation and standards for the annotation of corpora at several levels of linguistic analysis. The *Linguistic Annotation Framework*, LAF, (ISO/DIS 24612, 2009) is proposed as a generic meta-standard. It is based on the general assumption that corpus data should be encoded in a graph-based framework, i.e. by means of data structures composed of nodes and edges. GrAF, the *Graph Annotation Format* (cf. Ide and Suderman (2007)) is proposed as an XML-based formalism to encode LAF data.

For individual levels of linguistic description, proposals for encoding conventions have been made. The *Morphosyntactic Annotation Framework*, MAF (ISO/DIS 24611, 2008) covers the output of tokenizing and tagging, and the current proposals in SynAF *(Syntactic Annotation Framework*, ISO/DIS 24615 (2009)) describe the encoding of unambiguous syntactic structures[13]. The medium-term view is to encode terminals of parse trees according to MAF, and non-terminals and syntactic structure according to SynAF.

3.4.2. Encoding parsed data in LAF

We chose to work with the LAF format first, which is slightly more abstract than MAF and SynAF. In an experiment, we encoded material parsed with the BitPar parser (cf. Schmid (2004)) according to LAF. Figure 4 shows the BitPar parse tree[14].

[11] http://www.sfs.uni-tuebingen.de/de_tuebadz.shtml

[12] This is in line with the CLARIN Action Plan for Standardization (CLARIN: http://www.clarin.eu/) which invites CLARIN members to explore possibilities of relating CLARIN work with the standardization proposals currently being worked out by ISO TC 37/SC4.

[13] For a proposal to encode explicitly underspecified representations of syntactic ambiguity, see Kountz et al. (2008), Eckart (2009).

[14] Annotations: Morphology: Nom - nominative case, Pl - plural number, Fem - feminine gender; POS tags: ART - determiner, NN - normal noun, VVFIN - finite verb (full), PTKVZ - separated verb particle; categories: TOP - structural top node, S - sentence, NP - noun phrase; syntactic functions: SB/Pl - plural subject, HD - head, SVP - separated verb particle.

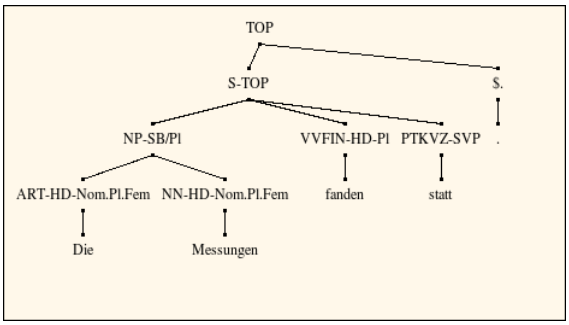

Figure 4: BitPar parse tree

```
   |D|i|e| |Messungen fanden statt|.|
   0 1 ...                          27
<graf:region id="r1" anchors="0 3"/>
<graf:region id="r2" anchors="4 13"/>
<graf:region id="r3" anchors="14 20"/>
<graf:region id="r4" anchors="21 26"/>
<graf:region id="r5" anchors="26 27"/>
```

Figure 5: LAF/GrAF regions

LAF uses character offsets to anchor tokenization to primary data, cf. figure 5.
In LAF tokens are referenced from annotations of higher levels; LAF and TCF are similar in this respect. As can be seen in figure 6, nodes refer to the annotation set used (attribute 'as' in figure 6) and to a label which may link to the actual tagset used (attributes provided for semantic interoperability). Nodes carry annotations in the form of feature structures.
Edges are referred to in terms of the nodes they link, and also annotated with 'type' and 'label' attributes for semantic interoperability and, possibly, with feature structures.

3.4.3. Encoding parsed data in MAF and SynAF

We also started some experiments regarding the encoding of annotated text data with MAF and SynAF. As these formats are still under development, the example presented in figure 7 and figure 8 shows our intuition about what MAF/SynAF encoded annotations could look like.
The feature names and values which are used in the annotations should be mapped to the data categories of ISOcat as described in section 3.2.. For this purpose MAF provides a 'tagset' element, where also libraries for compact representation of feature structures can be included. However the

```
<graf:node id ="n1">
  <graf:link to="r1"/>
  <graf:as type="BitPar">
    <graf:a label="msd">
      <ns1:fs>
        <ns1:f name="pos">
          <symbol value="ART"/>
        </ns1:f>
        <ns1:f name="case">
          <symbol value="Nom"/>
        </ns1:f>
        <ns1:f name="number">
          <symbol value="Pl"/>
        </ns1:f>
        ...
      </ns1:fs>
    </graf:a>
  </graf:as>
</graf:node>
```

Figure 6: LAF/GrAF node

```
<maf:token xml:id="t1" form="Die" xlink:href=
 "plaintext.xml#xpointer(string-range(//text,'',0,3))"/>
<maf:token xml:id="t2" form="Messungen" xlink:href=
 "plaintext.xml#xpointer(string-range(//text,'',4,13))"/>

<maf:wordForm tokens="t1">
  <fs>
    <f name="partOfSpeech"><symbol value="ART"/></f>
    <f name="case"><symbol value="Nom"/></f>
    <f name="grammaticalNumber"><symbol value="Pl"/></f>
    <f name="grammaticalGender"><symbol value="Fem"/></f>
  </fs>
</maf:wordForm>
<maf:wordForm tokens="t2">
  <fs>
    <f name="partOfSpeech"><symbol value="NN"/></f>
    <f name="case"><symbol value="Nom"/></f>
    <f name="grammaticalNumber"><symbol value="Pl"/></f>
    <f name="grammaticalGender"><symbol value="Fem"/></f>
  </fs>
</maf:wordForm>
```

Figure 7: MAF-annotations – SynAF terminal nodes

```
<synaf:node id="s1_n1">
  <fs>
    <f name="grammaticalUnit"><symbol value="NP"/></f>
    <f name="grammaticalNumber"><symbol value="Pl"/></f>
  </fs>
</synaf:node>
<synaf:edge id="s1_e1" s_node="s1_n1" t_node="t1">
  <fs>
    <f name="syntacticFunction"><symbol value="HD"/></f>
  </fs>
</synaf:edge>
<synaf:edge id="s1_e2" s_node="s1_n1" t_node="t2">
  <fs>
    <f name="syntacticFunction"><symbol value="HD"/></f>
  </fs>
</synaf:edge>
```

Figure 8: SynAF non-terminal node and edges to its children

example shows the spelled out feature structures.
The encodings refer to the same example as encoded with LAF in section 3.4.2., the BitPar tree in figure 4. Figure 7 shows the terminal nodes of the tree encoded as tokens and word forms in MAF. In this example the tokens refer to the primary data document via xpointers, but regarding the token element, MAF supports stand-off as well as an embeding notation. For the stand-off annotation an appropriate addressing scheme can be chosen. In MAF tokens and word forms share n-to-m relations and with the reference to token IDs, discontinuous word forms can be handled.
Figure 8 shows the encoding of the BitPar NP node along with the edges to its terminal nodes of the word forms *Die* and *Messungen*.

3.4.4. Relating TCF and ISO standards

As should have become clear from the presentation of TCF, LAF and MAF/SynAF, TCF has deliberately been designed in a similar spirit as the international standards, but with a view to keeping space and processing requirements as low as possible.
The similarity between LAF/MAF/SynAF and TCF allows for a relatively straightforward mapping at the level of syntactic interoperability. Since LAF is rather a metaformat, there is no need to be able to convert the full complexity of LAF into TCF. Instances of LAF, like MAF and SynAF or other tool formats wrapped in LAF, however, are of interest for conversion into TCF.
A partial converter from MAF to TCF has already been re-

alized. The complexity of this partial conversion is also easily managable: the converter consists in the current state of about 120 lines of code.
Figure 9 shows a case where a MAF token and its annotation is confronted with the respective TCF representation.
Nevertheless the converter is still in the process of being completed: MAF's encoding of n-to-m relations between word forms and tokens, as well as its way to handle discontinuous word forms (e.g. *er **setzt** den Hut **ab***) and compounds still need to be accommodated in TCF.
A task for semantic interoperation is the ongoing mapping of STTS to ISOcat feature structures. The approach for TCF here is to keep the e.g. the MAF conversion partial on this aspect. It is not intended to pass a complete tagset defnition section of e.g. a MAF document through a web service tool chain. The tagset definition with the ISOcat mapping could be kept in a separate file which is included in the exchange format after the tool chain was processed. A similar approach could be taken into account for other kinds of rather verbose meta data.

4. WebLicht

The WebLicht platform is of interest for both, web service providers as well as experienced and unexperienced users of linguistic tools and data.
On the one hand new web services can be added to WebLicht. To connect to the chains using TCF, the provider needs to write a wrapper and has to register the web service in a centralized repository. Registration includes stating the address of the available web service, technical metadata, which means specifing input and output layer, and descriptive metadata ,e.g. author, description of the web service (cf. Hinrichs et al. (2010)).
On the other hand the user interface is suited for the experienced as well as the unexperienced user of linguistic tools and data. Users can upload own text data or use text data available in WebLicht. Thereafter users can build a tool chain of available web services. The chainer only offers compatible web services as choice options. After processing the chain the results are displayed in TCF and with an appropriate visualization e.g. tables for part-of-speech tags. The user can download the final results as well as the results obtained after each step in the chain.
To find out more about the WebLicht platform see Hinrichs et al. (2010) and `http://weblicht.sfs.uni-tuebingen.de/weblicht.shtml`.

5. Comparison of TCF with other data formats

On a map of representation formats for text corpus data, TCF occupies an intermediate position between general formats as used in frameworks such as UIMA[15] or GATE[16], and 'linguistic' formats such as MAF and SynAF. It is similar in its principles to LAF, with the major difference that LAF has never been intended as a processing format but rather as a meta format and is therefore quite verbose.

[15] `http://incubator.apache.org/uima/`
[16] `http://gate.ac.uk/`

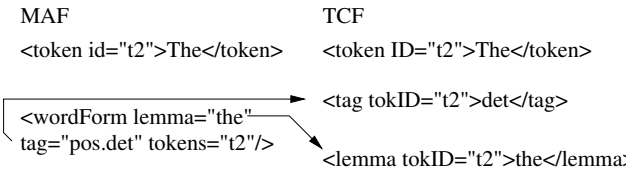

Figure 9: MAF vs. TCF: an example

On the one hand, TCF is able to provide interoperability between different tools as well as a common representation for different annotation layers, instead of only one specific layer such as e.g. SynAF, for syntactic annotation. On the other hand, TCF is designed especially for the application in a tool chain of different web services.

6. Conclusion

We showed the motivation for, and the architecture and main properties of the D-SPIN-internal corpus format TCF, and its relationships with both legacy formats for resources and international standards. Our work is both an attempt to provide formats suitable for an efficient processing of corpus data in a service-oriented architecture, and a contribution to ongoing discussions about standard formats for corpus representation.
Future work will be devoted to a full-fledged elaboration of TCF and to converters between TCF and the upcoming ISO standards. This may also provide input, from practical applications, to the ongoing specification of LAF, MAF and SynAF. Another objective is to further work out the representation of syntactic ambiguity. We are aware of the issues related with semantic interoperability: addressing some of them will be another challenge for the medium term.

7. References

Stefanie Dipper. 2005. XML-based Stand-off Representation and Exploitation of Multi-Level Linguistic Annotation. In *Berliner XML Tage*, pages 39–50.

Kerstin Eckart. 2009. Repräsentation von Unterspezifikation in relationalen Datenbanksystemen. Diplomarbeit, Universität Stuttgart.

Erhard Hinrichs, Marie Hinrichs, Thomas Zastrow, Gerhard Heyer, Volker Boehlke, Uwe Quasthoff, Helmut Schmid, Ulrich Heid, Fabienne Fritzinger, Alexander Siebert, and Jörg Didakowski. 2009. Weblicht: Web-based LRT services for German. In *Workshop on linguistic processing pipelines, GSCL Jahrestagung 2009*, Potsdam.

Marie Hinrichs, Thomas Zastrow, and Erhard Hinrichs. 2010. WebLicht: Web-based LRT Services in a Distributed eScience Infrastructure. In *Proceedings of LREC 2010*, Malta.

Nancy Ide and Keith Suderman. 2007. Graf: A graph-based format for linguistic annotations. In *Proceedings of the Linguistic Annotation Workshop, 1-8*.

ISO 12620:2009. Terminology and other language and content resources – Specification of data categories and management of a Data Category Registry for language resources.

ISO/DIS 24611. 2008. Language resource management - Morpho-syntactic annotation framework (MAF).

ISO/DIS 24612. 2009. Language resource management - Linguistic annotation framework (LAF).

ISO/DIS 24615. 2009. Language resource management - Syntactic annotation framework (SynAF).

Manuel Kountz, Ulrich Heid, and Kerstin Eckart. 2008. A LAF/GrAF-based encoding scheme for underspecified representations of dependency structures. In *Proceedings of the 6^{th} Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco, May. [CD-ROM].

Helmut Schmid. 2004. Efficient Parsing of Highly Ambiguous Context-Free Grammars with Bit Vectors. In *Proceedings of the 20th International Conference on Computational Linguistics, Coling'04*, volume 1, pages 162–168, Geneva, Switzerland.

Andreas Witt, Ulrich Heid, Felix Sasaki, and Gilles Sérasset. 2009. Multilingual language resources and interoperability. *Language Resources and Evaluation*, 43:1 – 14.