A Flexible Framework for Integrating Annotations from Different Tools and Tag Sets

Christian Chiarcos* — **Stefanie Dipper**** — **Michael Götze***
Ulf Leser*** — **Anke Lüdeling****** — **Julia Ritz*** — **Manfred Stede***

* *Institut für Linguistik, Universität Potsdam*
Karl-Liebknecht-Str. 24-25 – D-14476 Golm
(chiarcos|goetze|julia|stede)@ling.uni-potsdam.de

** *Sprachwissenschaftliches Institut, Ruhr-Universität Bochum*
Universitätsstr. 150 – D-44801 Bochum
dipper@linguistics.rub.de

*** *Institut für Informatik, Humboldt-Universität zu Berlin*
Unter den Linden 6 – D-10099 Berlin
leser@informatik.hu-berlin.de

**** *Institut für deutsche Sprache und Linguistik, Humboldt-Universität zu Berlin*
Unter den Linden 6 – D-10099 Berlin
anke.luedeling@rz.hu-berlin.de

ABSTRACT. *We present a general framework for integrating annotations from different tools and tag sets. When annotating corpora at multiple linguistic levels, annotators may use different expert tools for different phenomena or types of annotation. These tools employ different data models and accompanying approaches to visualization, and they produce different output formats. For the purposes of uniformly processing these outputs, we developed a pivot format called PAULA, along with converters to and from tool formats. Different annotations are not only integrated at the level of data format, but are also joined on the level of conceptual representation. For this purpose, we introduce OLiA, an ontology of linguistic annotations that mediates between alternative tag sets that cover the same class of linguistic phenomena. All components are integrated in the linguistic information system ANNIS: Annotation tool output is converted to the pivot format PAULA and read into a database where the data can be visualized, queried, and evaluated across multiple layers. For cross-tag set querying and statistical*

Traitement Automatique des Langues. Volume 49 – n° 2/2008, pages 217 à 246

evaluation, ANNIS uses the ontology of linguistic annotations. Finally, ANNIS is also tied to a machine learning component for semiautomatic annotation.

RÉSUMÉ. Dans ce papier, nous présentons une plateforme générale pour intégrer des annotations originaires de nombreux outils différents et employant des ensembles d'étiquettes divers. Quand un corpus fait l'objet d'une annotation multi-niveaux, les annotateurs peuvent profiter d'utiliser plusieurs outils experts différents, chacun adapté aux phénomènes ou types d'annotation envisagés. Ces outils employent différents modèles de données (accompagné par de différents méthodes de visualisation), et produisent des formats de sortie distincts. Pour permettre de processer ces sorties d'une manière uniforme, nous avons développé un format pivot, appelé PAULA, et des convertisseurs formats des et aux formats originals des outils. Les annotations ne sont pas integrées seulement au niveau de format, mais aussi au niveau de la représentation conceptionelle. Pour cela, nous introduisons OLiA, une ontologie des annotations linguistiques, qui met en relation les ensembles d'étiquettes alternatifs qui néanmoins recouvrent le même phénomène linguistique. Tous ces composants sont part du système d'information ANNIS: les données en format de sortie des outils d'annotation sont converties au format pivot PAULA et lues dans une base de données où on peut les visualiser, rechercher et exploiter à travers les multiples niveaux. Pour l'exploitation à travers les ensembles d'étiquettes différents, ANNIS est lié à l'ontologie susmentionnée. En outre, la plateforme comprend un composant export dans un environnement d'apprentissage automatique pour soutenir l'annotation semi-automatique.

KEYWORDS: Multi-level annotation; Corpus creation and maintenance; Linguistic database; Ontology-based querying

MOTS-CLÉS : Annotation multi-niveaux; Création et maintienabilité de corpus; Base de données linguistique; Recherche basée sur des ontologies

1. Introduction

A growing line of linguistic research today is dedicated to the investigation of less-resourced, less-studied languages (Asian, African, native American languages) and specific varieties of major languages deviating from written standard language (for example, dialectal and historical varieties or small genres). This paper proposes a fully-implemented architecture for creating and exploiting such small, deeply annotated corpora. Much of the annotation for such corpora has to be done manually because corpora are often too small to train automatic tools, or the annotation task is simply too difficult to be automated. Since, obviously, manual linguistic annotation is labor-intensive and expensive, it is of utmost importance to provide software environments that ensure the efficiency of the overall process.

Nowadays, a variety of annotation tools are freely available, which support different styles of annotation for different purposes, such as layer-based transcription or labelling of words/phrases, coreference links, syntax trees, or discourse trees. Combining different annotations of the same data leads to so-called "multi-level annotation", which has received surging interest in recent years. Such architectures were originally developed for multi-modal corpora which integrate spoken language, written representations of it, annotation, and perhaps visual material (films, etc.). Wittenburg (to appear) provides an overview of the history and formats of such multi-modal corpora. In recent years, multi-layer architectures are more and more used also for text corpora with many (possibly conflicting) annotation layers—see, e.g., the variety of annotations produced on *Wall Street Journal* data, starting with the Penn TreeBank (Marcus *et al.*, 1993) and turned into a multi-level framework by Pustejovsky *et al.* (2005). In this paper we focus on such examples.

When multiple annotations are integrated into a single database, inter-relationships between the annotations can be explored both qualitatively (by issuing database queries that combine levels) and quantitatively (by running statistical analyses or machine learning algorithms). We are convinced that these methods can significantly improve linguistic research: for instance, the researcher can formulate queries to find specific examples or counterexamples for a research hypothesis in "real" data, involving distinct levels of analysis, or perform statistical analyses to gather evidence for the distribution of particular feature patterns in corpora. Further, using multi-layer architectures, it becomes possible to represent and compare multiple, even conflicting, annotations of the same linguistic level for the same data, for example, competing syntax annotations.

However, when such multi-layer corpora are to be created with existing task-specific annotation tools, a new problem arises: output formats of the annotation tools can differ considerably, and annotations need to be aligned in order to be useful for purposes such as those mentioned above. To solve these problems, we have developed a software framework involving (i) a generic standoff representation format, (ii) conversion scripts from tool output to the generic format, (iii) alignment of multiple annotations, and (iv) a database that allows for visualization, retrieval, and statistical

analyses of the data. Our work is embedded in a large and long-term "collaborative research center" on information structure[1] at the University of Potsdam and Humboldt-University Berlin. Thus, our framework has been primarily developed to account for the specific resources and goals of that center. The architecture and methodologies, however, are applicable to numerous other scenarios involving different research questions and other types of annotations.

Having such a software infrastructure at hand, a natural step is to also integrate corpora that are already well-established and have proven to be beneficial to linguistics research. This makes sense both for the manual querying scenario and for the statistical analysis scenario. But, of course, this poses another problem: given a set of existing corpora, particular levels of linguistic description, such as morphosyntax, are very often annotated according to different annotation schemes or tag sets. Thus, in addition to the *technical* integration of different XML formats, another task arises, i.e., the *conceptual* integration of multiple annotation schemes. Different annotation schemes rely on independent, often theory-specific, conceptualizations of tags and categories and often different theoretical motivations. In response, our approach offers to ensure interoperability also with respect to tag sets by linking annotations to ontological representations. In particular, the application of ontologies allows us to specify complex relationships between annotations and reference concepts, whereas a traditional mapping approach is only capable of expressing a 1:n-mapping.

The paper is organized as follows: Section 2 provides more background information, introduces related work on technical integration and conceptual integration of heterogeneous annotations, and outlines the general architecture of our system. Section 3 introduces PAULA, the standoff XML format that technically mediates between different annotation formats. Section 4 describes OLiA, an ontology of linguistic annotations that conceptually mediates between different tag sets. Then, Section 5 describes ANNIS, our linguistic database, its implementation, and the associated query languages. Section 6 gives an example for the utility of "annotation mining" across different levels of annotation, and Section 7 summarizes the main contributions of the paper and points to areas for future research.

2. A Flexible Framework for Integrating Annotations

Nowadays the need for standardized annotation schemes and representation formats is widely recognized. Language resources must be well-documented and annotations easy to interpret if they are to be beneficial for users other than the corpus developers themselves. Standardization of technical representation formats concerns both the *physical* and *logical* data structures (see, e.g., Schmidt (2005), Ide *et al.* (2003)). Moreover, we also consider the *conceptual* integration of annotations, which has been subject to several standardization initiatives (Leech *et al.*, 1996; Atwell *et al.*, 1994; Erjavec, 2004; Ide *et al.*, 2003).

1. `http://www.sfb632.uni-potsdam.de/`

2.1. *Representation Formats*

The logical data structure refers to the *data models* used to represent the linguistic phenomena and their properties. We can distinguish three major types of data structures: (i) "annotation graphs": labeled directed acyclic graphs (LDAGs) whose nodes refer to a time line; annotation graphs are typically used for modeling time-aligned information (Bird *et al.*, 2001); (ii) structural annotations: LDAGs whose nodes refer to other nodes; usually used for syntactic and other tree-like annotations; (iii) feature structures, used, e.g., for syntactic analyses in frameworks such as HPSG and LFG, but rarely used in the context of corpus annotation.

The division between the paradigms of time-aligned annotation graphs and hierarchical structures has weakened in recent years. For instance, the data model of annotation graphs has been generalized, resulting in the ATLAS format (Laprun *et al.*, 2002), which supports both annotation graphs and hierarchical structures. Similarly, the NITE Object Model (Carletta *et al.*, 2003b), the DDD ODAG model (Dipper *et al.*, 2004; Faulstich *et al.*, 2005), and the general-purpose Linguistic Annotation Framework (LAF, Ide *et al.* (2003)) serve both camps.

The physical data structure, on the other hand, refers to the "exterior" representation of the data. The de facto standard for representing and exchanging data is XML, which is furthermore well-suited for permanent storage. Often, a standoff architecture is used, which stores primary data and its annotations separate from each other (as proposed, e.g., in the TEI (Sperberg-McQueen *et al.*, 1994) and MATE guidelines (Dybkjær *et al.*, 1998)). For the serialization of structural annotations, a natural way to represent trees is by using XML embedding structures. If structural annotations contain non-tree-like structures (e.g. crossing branches for discontinuous constituents), extra means like `xlink` attributes have to be employed (König *et al.*, 2000). Such representational means are harder to interpret than the straightforward representation via XML embedding and more difficult to incorporate into standard querying mechanisms (Trißl *et al.*, 2007).

Besides these two types of data structures there is a third one which is usually completely hidden from the user: the implementation model, i.e., the data format that is used for internal processing. For this format, there are essentially three options: (i) proprietary, tool-specific formats, (ii) XML, (iii) relational databases. Concentrating on non-proprietary solutions, one advantage of using XML both as the exchange and implementation format is that it allows for seamless file management. Yet this comes at the price of severe difficulties in formulating queries spanning several files. For example, XQuery does not easily handle queries to standoff formats, where annotations and primary texts are stored in different XML files. Querying such structures with XQuery does either require the use of XPointers or the use of embedded functions, both of which are not properly optimized by current XQuery implementations. This puts the burden of writing fast queries back on the developer and away from the database system. In the relational model, however, the modeling of non-hierarchical annotations and performant queries to these is relatively simple, though the underly-

ing model is more complex. As efficient processing of non-tree-like structures is one of the primary goals of our implementation, we thus accomplish our implementation with a relational database rather than an XML database. Using relational databases offers the additional advantages of a well-established technology (in terms of scalability, robustness, tool-support, etc.), but it requires the installation and maintenance of an extra software infrastructure.

2.2. *Conceptual Integration*

Conceptual integration is necessary when dealing with multiple annotation schemes, when either different terms are used for the same phenomenon, or the phenomenon is conceptualized in different ways. One possible solution for the integration of different annotation schemes is the standardization of tag sets, i.e., the direct mapping between a particular annotation and a meta tag in a reference tag set. Such meta tags are then either based on one particular standardized annotation scheme (Leech *et al.*, 1996; Erjavec, 2004), an interlingua mediating between tag sets (Atwell *et al.*, 1994), or a set of reference categories (Ide *et al.*, 2003).

For the specification of reference concepts and in order to abstract from concrete annotations, Farrar *et al.* (2003) and de Cea *et al.* (2004) have developed ontology-based accounts for the modeling of linguistic terminology relevant to annotation purposes.

In our approach, this ontology-based account is extended, in that not only is reference terminology specified within the ontology, but also the original annotation schemes and the linking between schemes and reference concepts are represented by means of an ontology. In particular, this allows for detailed specifications of the linking between annotations and the underlying reference concepts, and also for the robust and lossless integration of heterogeneous annotations.

This represents an important methodological advantage over standardization accounts, such as Leech *et al.* (1996), as annotations and reference concepts need not be defined in a 1:1 (or 1:n) relation; rather, complex relationships can be expressed. As compared to other ontology-based architectures, like de Cea *et al.* (2004), Farrar *et al.* (2003), our mapping between annotations and reference concepts is represented in the ontology itself, rather than hidden in opaque transformation scripts, and thus it is transparent, flexible, and modifiable. Users can explore and edit the mapping between annotations using standard OWL browsers and editors, e.g., Protégé.[2]

By integrating this ontology with our linguistic information system, we gain the possibility of searching across large amounts of heterogeneously annotated data by means of a single instruction formulated in the query language of the database.

2. `http://protege.stanford.edu`

2.3. *System Architecture*

Turning now to projects that actually deal with data annotated at multiple layers, one can identify different types of approaches: some projects define task-specific (albeit flexible) formats or extend existing ones (Baumann *et al.*, 2004; Erk *et al.*, 2004) and build specialized tools for these formats. Large long-term projects like NITE (Carletta *et al.*, 2005; Carletta *et al.*, 2003a), or ATLAS (Laprun *et al.*, 2002) develop entire toolkits for multi-level annotation in general, i.e., libraries for data and annotation management, which can be used by various kinds of "customers". These toolkits are, in general, not "ready to use" but require certain "data preprocessing" by the user (e.g., by specifying stylesheets).

Operating at the level of physical data, Witt *et al.* (2005) merge multiple XML annotations of the same primary data into one XML format, leaving the original annotations intact as far as possible. For the representation of structurally conflicting markup, elements are broken up and transformed into milestones. In contrast, Ide *et al.* (2007) propose one common pivot standard format, "GrAF", which all annotations have to be mapped onto. The format makes use of generic XML element names such as `node` and `edge` and encodes feature-value annotations by generic XML attributes `name` and `value` (e.g. `name="cat" value="NP"`).

In our approach, we pursue a strategy similar to Ide *et al.* (2007). Our representation format PAULA can be mapped onto the GrAF format. Also, the function of the ontology applied to the conceptual integration of different annotations can be compared to the Data Category Registry (DCR) described by Ide *et al.* (2004), although it is based on a more expressive formalism, i.e., the OLiA ontologies.

On this basis, PAULA and OLiA establish a neutral level of representation for different types of annotations, which then allows the integrated processing of heterogeneous resources in the linguistic information system ANNIS (see Section 5). ANNIS supports querying and visualizing the data and its multi-level annotation, and includes ontology-based query evaluation which allows for searching data annotated with different tag sets. Furthermore, we have developed a server-based implementation of ANNIS to ensure high-speed query execution even for very large corpora. See Figure 1 for a sketch of the overall system architecture. This integrated, "ready-to-use" architecture distinguishes our approach from the earlier approaches mentioned above. Throughout the paper, we keep referring to a particular annotation example, so that the reader can relate the different aspects of the system to one another. To this end, we are using two structurally comparable example sentences from German data collections: the TIGER corpus (Brants *et al.*, 2004) and the QUIS corpus (Götze *et al.*, 2005). (1) below shows an example from the QUIS corpus with glossing annotation (second line) according to the LISA guidelines (Dipper *et al.*, 2007) and a word-by-word translation. Example (2) is taken from the TIGER corpus; we here provide morphological annotation (second line) according to TIGER guidelines (Albert *et al.*, 2003) and a translation.

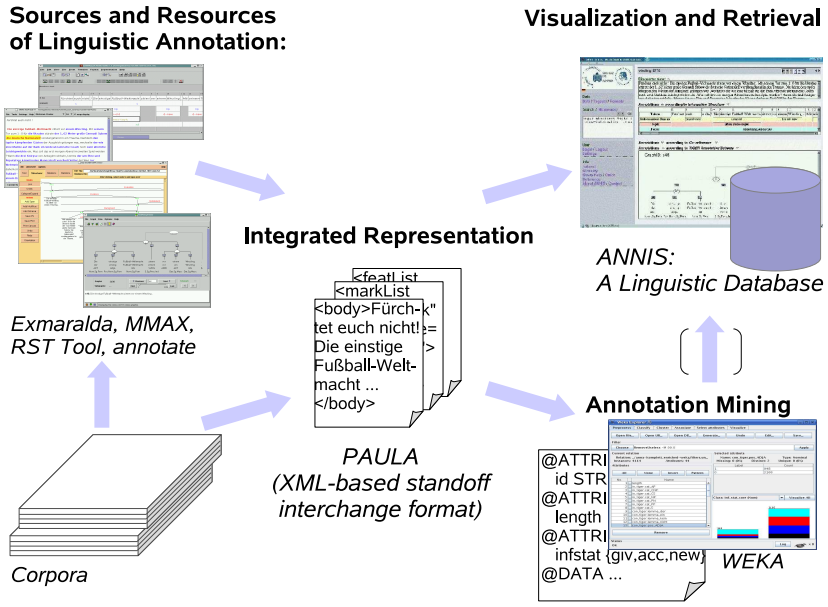

Figure 1. *Our system architecture for managing heterogeneous linguistic annotations.*

(1)

und	*ihr*	*Mann*	*hat*	*einen*
and	POSS.3.SG.F-M.SG.NOM	husband.M[SG.NOM]	have:3SG	IDEF-M.SG.ACC.
and	her	husband	has	a

Obstsalat	*zubereitet*
fruit-salad.M[SG.ACC]	prepare:PTCP.PRF
fruit salad	prepared

'(...) and her husband prepared a fruit salad'

(2)

Sein	*Tod*	*hatte*	*damals*	*eine*	*große*
Nom.Sg.Masc	Nom.Sg.Masc	3.Sg.Past.Ind		Acc.Sg.Fem	Pos.Acc.Sg.Fem
his	death	had	at this time	a	great

Protestwelle	*ausgelöst.*
Acc.Sg.Fem	Psp
wave of protest	caused

'His death caused a great wave of protest at the time.'

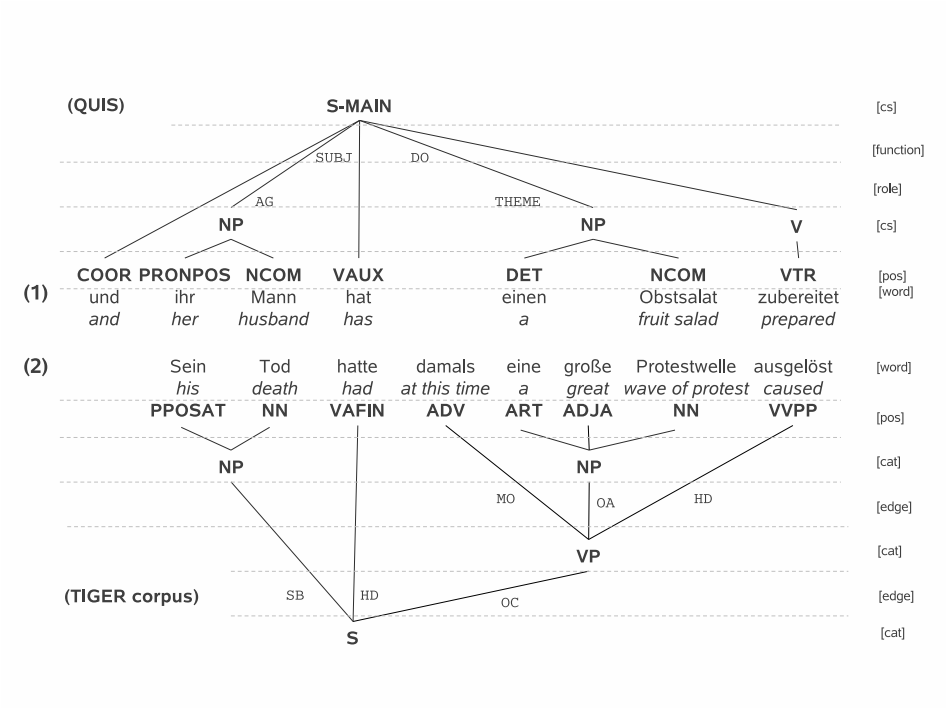

Figure 2. *Analysis of structurally comparable example sentences according to different annotation schemes.*

The syntactic analyses, according to the respective annotation schemes used in these corpora, are presented in Figure 2. Both examples are comparable in that—with the exception of the conjunction in (1) and the adjective and adverb in (2)—both examples involve the same sequence of parts of speech and syntactic structure. Table 1 specifies the tags used for the comparable phrases in (1) and (2) and Figure 2.

3. PAULA: A Generic Standoff Format for Integrating Annotations

Our representation format PAULA[3] (a German acronym for "Potsdam interchange format for linguistic annotation") focuses on the integration of different annotation structures. We assume that corpus developers apply specialized annotation tools which are tailored to the specific annotation tasks. For instance, *annotate* (Brants *et al.*, 2000) is frequently used for syntactic annotations; *Palinka* (Orasan, 2003) or

3. `http://www.sfb632.uni-potsdam.de/~d1/paula/doc/`

LISA	TIGER	
PRONPOS [pos], POSS [gloss]	PPOSAT [pos]	(attributive) possessive pronoun
NCOM [pos]	NN [pos]	common noun
SG [gloss]	Sg [morph]	singular
NOM [gloss]	Nom [morph]	nominative
NP [cs]	NP [cat]	noun phrase
S-MAIN [cs]	S [cat]	finite clause
n.a	NK [edge]	noun component
SUBJ [function]	SB [edge]	subject
		cf. Figure 2, (1), (2)

Table 1. *Comparing LISA and TIGER annotations.*

MMAX2 (Müller *et al.,* 2006) for discourse-level annotations such as coreference; the *RSTTool* (O'Donnell, 2000) for discourse structure annotation; *EXMARaLDA* (Schmidt, 2004) is applied for dialogue transcription and various layer-based annotations. For these tools (and for generic inline-XML annotations), we provide scripts that map the respective tool output to our representation format. The scripts are publicly available via the Internet: users can upload their data and annotations, and the data is converted automatically to PAULA. The user can load the PAULA data into the information system ANNIS or further export it to WEKA (see Section 6).

The mappings from the tool outputs to our format are defined such that they only transfer the annotations from one format into another, without interpreting them or adding any kinds of information.

3.1. *PAULA: Logical Structure*

The conceptual structure of the PAULA format is represented by the PAULA Object Model (POM). The PAULA Object Model operates on a labeled directed acyclic graph. Similar to the NITE Object Model (Carletta *et al.*, 2003b, NOM) and the GrAF data model (Ide *et al.*, 2007), nodes correspond to annotated structures, edges define relationships between independent nodes. Both nodes and edges are labeled, and generally labels define the specifics of the annotation. Nodes refer to other nodes, or point to a stretch of primary data. In these aspects, the POM generalizes over annotation graphs and hierarchical annotations and thus represents a generic formalism.

Besides labels that define concrete annotation values, a specialized set of labels also serves to indicate the *type* of an edge or a node. For a specific set of predefined edge labels, the POM defines the semantics of the relation expressed by the corresponding edge. As such, the *dominance* relation is characterized as a transitive, non-reflexive, antisymmetric relation. Furthermore, a dominance relation requires that the primary data covered by the dominated node is covered by the dominating node

as well. Thus, on the basis of *dominance* relations, tree structures (e.g., syntax trees) can be represented. Another predefined edge type is *reference*, a non-reflexive, anti-symmetric relation. Reference relations may occur with different annotation-specific labels. Reference relations with the same label, e.g. "anaphoric_link", or "dependency_link" are also transitive. On the basis of *reference* relations, dependency trees, coreference relations, and alignment in multilingual corpora can be expressed.

The POM differs from related proposals, e.g., GrAF, in the definition of explicit semantics for certain edge types. The specifications of the dominance relation are comparable to the NITE Object Model, but while NOM has a stronger focus on hierarchical annotation, POM also formulates the semantics of pointing relations.

On the basis of this general object model, annotation-specific data models are then defined with reference to the POM.

3.2. *PAULA: Physical Structure*

The elements of the PAULA representation format along with the corresponding POM entities are shown in Table 2. The third column gives the corresponding labels for our relational database model, which will be introduced in Section 5.2; see in particular Figure 7.

PAULA element	POM entity	RelDB entity
`tok`(en)	terminal node	text_elem
`mark`(able)	non-terminal node (containing *references* to nodes)	struct_elem
`struct`(ure)	non-terminal node (containing *dominance relations* to nodes)	struct_elem
`rel`(ation)	within `struct`: *dominance*, otherwise *reference* relation	struct_elem
`feat`(ure)	annotation label	anno_attribute
`multiFeat`(ure)	bundles of annotation labels	anno_elem

Table 2. *PAULA: elements of physical and logical structure*

As a first example of the PAULA format, consider the original annotation of the phrase *ihr Mann* 'her husband' from example (1), annotated with the tool *EXMARaLDA*. Figure 3 shows selected annotation levels, as displayed by the annotation tool. EXMARaLDA's XML representation format implements annotation graphs, i.e., the primary data and all annotations refer to a common timeline, marked by timeline items (`tli`), whose IDs serve as anchors for the annotations. Annotations are called `events`, and are anchored to the timeline via `start/end` attributes. The `tier` element specifies the type of annotation (e.g. `pos`), the `event` tags contain the actual annotation values (e.g. `PRONPOS` for possessive pronoun). The following fragment displays the primary data `ihr Mann` ('her husband') and their POS annotations.

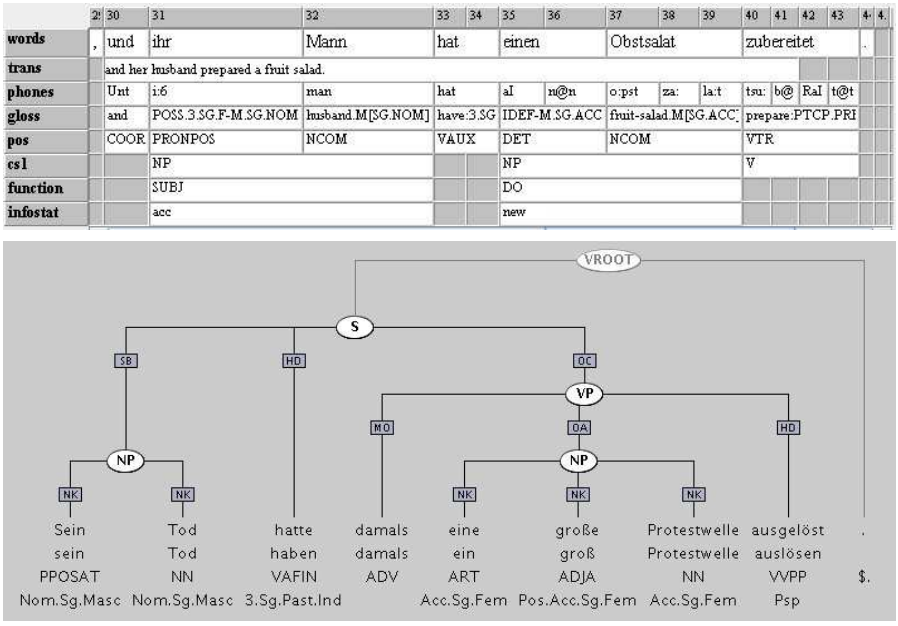

Figure 3. *Examples (1) and (2), annotated in EXMARaLDA and TIGER respectively.*

```
<tli id="T18"/>
<tli id="T19"/>
<tli id="T44"/>
...
<tier id="TIE1" category="words">
  ...
  <event start="T18" end="T19">ihr</event>
  <event start="T19" end="T44">Mann</event>
  ...
</tier>
<tier id="TIE13" category="pos">
  ...
  <event start="T18" end="T19">PRONPOS</event>
  <event start="T19" end="T44">NCOM</event>
  ...
</tier>
```

The corresponding representation of our pivot format PAULA presents the primary data in a `body` element stored, e.g., in a file called “exmaralda.85DEU.text.xml”. In a separate file, “exmaralda.85DEU.tok.xml”, `markables` are defined, i.e., segments that receive annotations. In the POM, these correspond to nodes in the graph. A first

layer of markables points to text regions in the body element, by means of XPointer expressions (see the markables with IDs tok_20/21 below). These markables correspond to terminal nodes in the POM and can be thought of as tokens—information which is encoded by the attribute type="tok" of the enclosing <markList> tag. Another layer of markables is added on top of the token markables (see the "pos-segment" markables with IDs pos_15/16); they point to the tokens by means of xlink:href attributes. The actual POS annotations "PRONPOS" and "NCOM" are encoded by feat elements ("features"), which are anchored to the second layer of markables. As in the case of markables, the type of annotation ("pos") is encoded by the type attribute of the enclosing tag; the attribute value represents the annotated value (e.g., "PRONPOS").

File *exmaralda.85DEU.text.xml*:
```
<body>... ihr Mann ...</body>
```

File *exmaralda.85DEU.tok.xml*:
```
<markList type="tok" xml:base="exmaralda.85DEU.text.xml">
 <mark id="tok_20" xlink:href="#xpointer(string-range(//body,'',97,3))"/>
    <!-- ihr -->
 <mark id="tok_21" xlink:href="#xpointer(string-range(//body,'',101,4))"/>
    <!-- Mann -->
 ...
</markList>
```

File *exmaralda.85DEU.posSeg.xml*:
```
<markList type="posSeg" xml:base="exmaralda.85DEU.tok.xml">
 <mark id="pos_15" xlink:href="#tok_20"/>
 <mark id="pos_16" xlink:href="#tok_21"/>
 ...
</markList>
```

File *exmaralda.85DEU.posSeg_pos.xml*:
```
<featList type="pos" xml:base="exmaralda.85DEU.posSeg.xml">
 <feat xlink:href="#pos_15" value="PRONPOS"/>
 <feat xlink:href="#pos_16" value="NCOM"/>
 ...
</featList>
```

For the encoding of hierarchical structures, including labeled edges, PAULA provides specific elements struct and rel. Like markables, a struct element represents a node in the POM, but in this case a node which is the parent node of a dominance relation. The dominance relation is expressed by the rel element. An annotation example with hierarchical syntax annotation is shown in Figure 3. A PAULA struct element with its daughters corresponds to a local TIGER subtree, i.e., a mother node and its immediate children. For instance, the subtree dominated by the first NP in Figure 3, *sein Tod*, 'his death', is represented by a struct element that, via rel elements, embeds the daughter tokens with IDs tok_26/27 (these are stored in a separate file called "tiger.ex.tok.xml"). The NP subtree itself is dominated by another

`struct` element, with ID `const_14`. `feat` elements encode the categorial status of these subtrees, “NP” and “S” respectively, and their grammatical functions: e.g., the `rel` element with ID `rel_39`, which connects the subtree of S with the subtree of the NP, is marked as “SB” relation by the `feat` element pointing to `#rel_39`.

File *tiger.TIG49796.const.xml*:

```
...
<struct id="const_11">
   <rel id="rel_30" type="edge" xlink:href="tiger.ex.tok.xml#tok_26"/>
      <!-- Sein -->
   <rel id="rel_31" type="edge" xlink:href="tiger.ex.tok.xml#tok_27"/>
      <!-- Tod -->
</struct>
<struct id="const_14">
  <rel id="rel_38" type="edge" xlink:href="tiger.TIG49796.tok.xml#tok_28"/>
      <!-- hatte -->
  <rel id="rel_39" type="edge" xlink:href="#const_11"/>
  <rel id="rel_40" type="edge" xlink:href="#const_13"/>
</struct>
...
```

File *tiger.TIG49796.const_cat.xml*:

```
...
 <feat xlink:href="#const_11" value="NP"/>
 <feat xlink:href="#const_14" value="S"/>
...
```

File *tiger.TIG49796.const_func.xml*:

```
...
 <feat xlink:href="#rel_30" value="NK"/><!-- Sein -->
 <feat xlink:href="#rel_31" value="NK"/><!-- Tod -->
 <feat xlink:href="#rel_39" value="SB"/>
...
```

As mentioned at the beginning of Section 3, we assume that different annotation tools are applied, which are tailored to different annotation tasks. In our framework, a text that has been annotated by different tools, at multiple levels, can be searched *across* the different annotation layers. This is achieved by first mapping the tool-specific formats to separate “packages” of PAULA files. Next, the annotations need to be *synchronized*, i.e., the primary data and the token layers from the individual packages have to be merged. From all PAULA packages, each containing individual files with primary data and token markables of their own, only one file with primary data and token markables is retained; XPointer links from the other packages are updated accordingly. Finally, to guarantee that all IDs are unique, namespaces are added to the attributes; for instance, ID `const_11` from the TIGER syntax example above is renamed to `tiger:const_11`. We refer to the overall process as “PAULA-merge”.

4. An Ontology of Linguistic Annotations

So far, we have described aspects of the technical integration of multi-layered annotations from different sources and their representation. However, the integration of data from different sources (and partially from different languages) not only involves the integration of technical formats but also the conceptual integration. It is well known that tag identifiers can differ widely and quite often involve idiosyncratic abbreviations. As an example, consider the great variety of tags assigned to *her* as a possessive determiner in different tag sets for English, which show a high degree of variation with varying transparency of the tag name chosen: `PP$` (Brown, Greene *et al.* (1981)), `TB` (London-Lund Corpus, Eeg-Olofsson (1991)), `PRP$` (Penn, Santorini (1990)), `DD` (POW, Souter (1989)), `PRON(poss,sing)` (ICE, Greenbaum (1992)), `APPGf` (Susanne, Sampson (1995)).

Individual tag sets, even if designed for one particular language, may differ heavily in their choice of tag names, the tag definitions, or their level of analysis. A typical but often neglected problem is the definition of tags in terms of form or function. As such, POS tag sets developed in technical contexts often integrate surface ambiguities in tag definitions in order to enhance the performance of automatic POS taggers. On the other hand, tag sets designed for manual annotation concentrate on the "proper" differentiation of different functions. To give an example, the German verb *haben* ('to have') serves as an auxiliary verb, but can also be used in its original lexical meaning, 'to own'. In the LISA scheme, both grammatical functions are properly distinguished, and *haben* is assigned the tag `VAUX` if used as an auxiliary, but `VLEX` if used in its lexical meaning. As opposed to this, in the STTS tag set (Schiller *et al.*, 1999) incorporated in the TIGER guidelines, *haben* is to be assigned the tag `VAFIN, VAINF,` etc., regardless of its current use in the clause.[4]

In order to overcome these difficulties, we employ the OLiA ontologies (Chiarcos, 2006; Chiarcos, to appear), a structured set of modular ontologies. By specifying a terminological reference, the ontologies allow for both the conceptual integration of different annotation schemes, and the lossless representation of such and similarly complicated conceptualizations: for the STTS tag `VAFIN`, thus, an appropriate ontological description would be $\texttt{VAFIN} \in \texttt{LexicalVerb} \cup \texttt{AuxiliaryVerb}$, indicating that `VAFIN` applies to either auxiliaries or lexical verbs. Moreover, an ontological representation allows us to specify the properties that constitute a given reference concept and to refer to these properties directly rather than to a reference concept which only loosely corresponds to a given concept in an annotation scheme.

In our account, a clear and transparent linking between reference terminology and the terms used in individual annotation schemes is implemented by means of con-

4. Here, we concentrate on morphosyntactic annotations for reasons of brevity. However, we have also developed ontologies for syntax, coreference, and information structure. In fact, for higher levels of annotation, the problem is even more pervasive, yet it already occurs with the most fundamental types of annotation, such as part of speech.

ceptual subsumption ($\subseteq$)[5] between different modules of an integrating structured ontology. It involves two primary modules, a set of *annotation models* (each of which is a representation of one annotation scheme) and the OLiA *reference model*, which represents a generalization over different annotation models and thus a common terminological reference.

A given annotation model is constructed solely on the basis of available annotation documentation, mostly guidelines if available, and annotated examples. Hence, it is a formalization of the annotation documentation, exhaustive with respect to the available documentation, but without any additional interpretation in terms of generally assumed linguistic categories, etc. The partial ontological representation of the `pos` and `gloss` annotations of the possessive pronoun *ihr* ('her') from example (1) in terms of the LISA annotation model is given in Figure 4.[6] In the same way, annotations of the STTS tag set are represented in a separate annotation model. While an annotation model is specific to one particular language, community, or purpose, the reference model is a general terminological resource, and consequently based on a broad range of resources, including specific annotation models, grammatical references, textbooks, but also existing terminological references such as the EAGLES recommendations for morphosyntax (Leech *et al.*, 1996), and the GOLD ontology (Farrar *et al.*, 2003). In case of divergent conceptualizations, e.g., the classification of attributive possessive pronouns as either Pronouns or Determiners, the EAGLES taxonomy was taken as an orientation.

Annotation models and the reference model represent self-contained ontologies on their own. The conceptual integration of annotation models is then performed by means of a declarative *linking* between both the reference model and a specific annotation model. In the linking, every concept (class) of the annotation model is assigned a superclass from the reference model—including complex superclasses composed with the set operators $\cup, \cap$, or $\setminus$.

For the annotation model fragment in Figure 4, the corresponding linking of concepts and the property `hasNumber` with their respective counterparts in the reference model is illustrated in Figure 5. Following the linking, the concise annotation of the possessive pronouns in the examples (1) and (2) in Figure 3 can be rephrased in terms of the reference model. Thus, an ontological description such as `PossessivePronoun` naturally expands (by means of $\subseteq$ and $\in$) into a disjunction of several specific annotations according to different annotation models, e.g., subsuming

5. More appropriate than $\subseteq$, etc. would be the operators $\sqsubseteq$, etc. However, for the sake of convenience, we stick to commonly well-understood set operators.

6. Note that the ontology accounts for both inherent and morphologically expressed properties. With respect to gender, the property `hasGender` has two sub-properties `hasInherentGender` and `hasGrammaticalGender`, with different values for *ihr*, i.e. `hasInherentGender(Feminine)` and `hasGrammaticalGender(Masculine)`. Similarly, the `thirdGloss` feature in Figure 4 is subject to property `hasInherentPerson` rather than `hasGrammaticalPerson`.

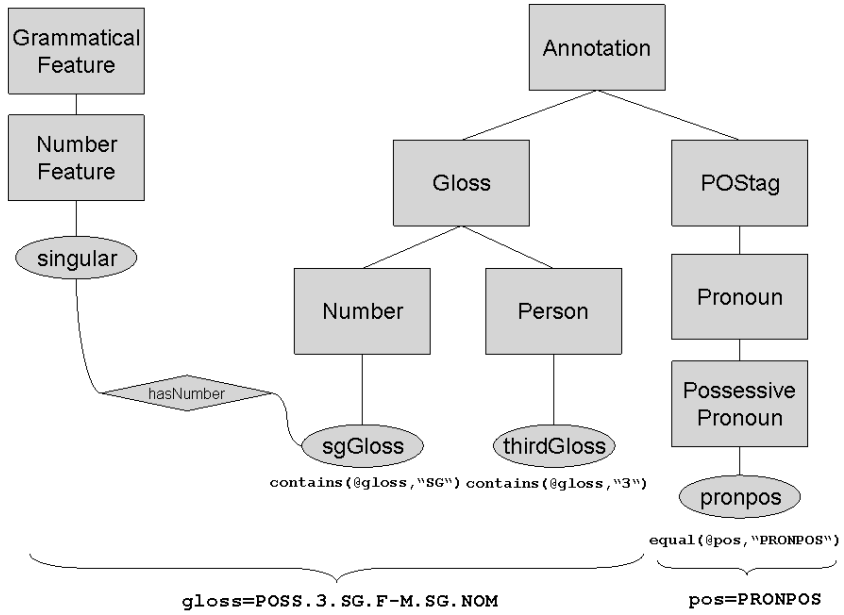

Figure 4. *Fragment of the LISA Annotation Model.*

both tags for possessive pronouns used in the examples, i.e., the tag `PRONPOS` in (1) and the tag `PPOSAT` in (2).

Beyond the form of a particular tag in a given tag set, every individual in the ontology that represents an annotation value is also assigned a property `hasTier`, which identifies the annotation layer on which the corresponding annotation is used, here `pos`. Thus, `PossessivePronoun` translates into conditions as described by (3).

(3)
```
equal(@pos, ‘‘PRONPOS’’) ∨                                  (LISA)
equal(@pos, ‘‘PPOSAT’’) ∨ equal(@pos, ‘‘PPOSS’’)            (STTS)
```

Using these explicit references to a particular annotation layer, it is also possible to retrieve annotation values from different annotation layers in combination (cf. Figure 5). Hence, the query `PossessivePronoun and hasNumber(Singular)` requires the combination of information from multiple annotation layers, i.e., from both `[pos]` and `[morph]`, cf. (4):

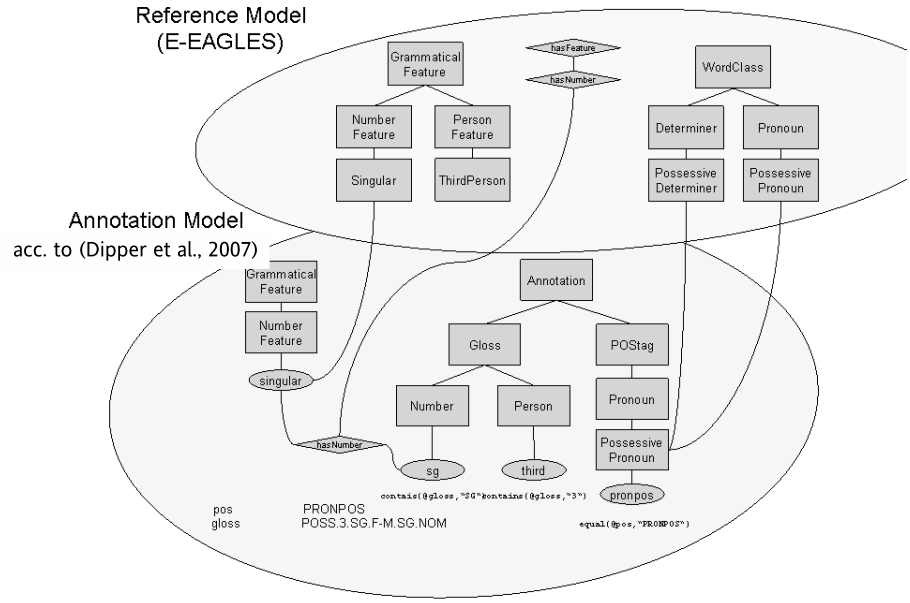

Figure 5. *Fragment of the Reference Model and its linking with the LISA Annotation Model.*

(4)	(equal(@pos, ‘‘PRONPOS’’) ∧ contains(@gloss, ‘‘SG’’)) ∨ ((equal(@pos, ‘‘PPOSAT’’) ∨ equal(@pos, ‘‘PPOSS’’)) ∧ contains(@morph, ‘‘.Sg.’’))	(LISA) (STTS, TIGER)

Using this approach, the user can formulate an ontological description without having to be aware of the different ways this information is represented in the annotation schemes. And in fact, for different corpora, different strategies for the splitting of annotation layers are used, e.g., representing morphological information as an independent layer (as in the TIGER scheme), or combining it with information on lexical semantics (as in the LISA scheme), or merging it with part-of-speech annotation (as, for grammatical number, in Sampson (1995)). This abstract perspective on the information conveyed in the annotation motivates the application of the ontology for concept-based corpus querying, cf. Section 5.3.

The main advantage of this structured account is its avoiding the plain 1:n-mapping between categories from different annotation schemes and “standard” categories, as is required in classical standardization approaches, such as that of Leech

et al. (1996). Also, the notion of Data Categories, advocated by ISO TC37 SC4, is a step toward solving the issue, but suffers from similar limitations. In our approach, relations of high complexity ($\cup, \cap, \setminus$) can be specified and the necessary *interpretation* of categories in the annotation scheme represented in an explicit, transparent, and modifiable way.

This tripartite structure of annotation models, reference model, and the linking between them can be augmented by the optional linking of the reference model with additional *external reference models*, i.e., ontological formalizations of community- or language-specific terminological systems. Currently, we provide a linking with three external reference models, the General Ontology of Linguistic Description (GOLD, Farrar *et al.* (2003)), developed in the context of language documentation, the OntoTag ontologies (de Cea *et al.*, 2004) developed in the context of Semantic Web applications, and an OWL representation of the Data Category Registry. Thus, annotations are not only tied to the OLiA Reference Model, but also to other existing terminological resources.

We claim that this modular approach is more flexible, as it allows the users to specify their own linkings, annotation models, and external reference models, and to modify these using established OWL editors. In contemporary annotation practice, the "technological counterpart" to this approach is the standoff paradigm (see Section 3).

5. Querying Multiply Annotated Corpora

Having discussed both technical and conceptual issues of data integration, we now turn to the task of accessing integrated, multi-level corpora. This includes, besides the *identification* of language resources by means of meta data (addressed by initiatives such as OLAC[7] or IMDI[8] and the respective tools), the tasks of *querying* and *visualizing* the data.

The overall goal of our linguistic information system ANNIS[9] is to provide easy access to heterogeneous multi-level annotations by providing suitable means both for querying and visualization. By supplying import facilities for the PAULA pivot format described in Section 3, we support the idea of distributed annotation with specialized ready- and easy-to-use tools. Different *types* of annotation (markables, trees/graphs, links) are distinguished in the data model, and can be visualized accordingly. In existing frameworks, such as the NITE XML Toolkit (NXT, Carletta *et al.* (2003a)) or GATE (Cunningham *et al.*, 2002), integrating new corpora may necessitate adaptations to the visualization filter (in NXT: stylesheets). At present, our usage scenarios include the development and analysis of historical corpora, construction of a typological database with data from 16 different languages (Götze *et al.*, 2005), and the creation of a text corpus with rich discourse-related annotations (Stede, 2004).

7. `http://www.language-archives.org/`
8. `http://www.mpi.nl/IMDI`
9. `http://www.sfb632.uni-potsdam.de/annis/`

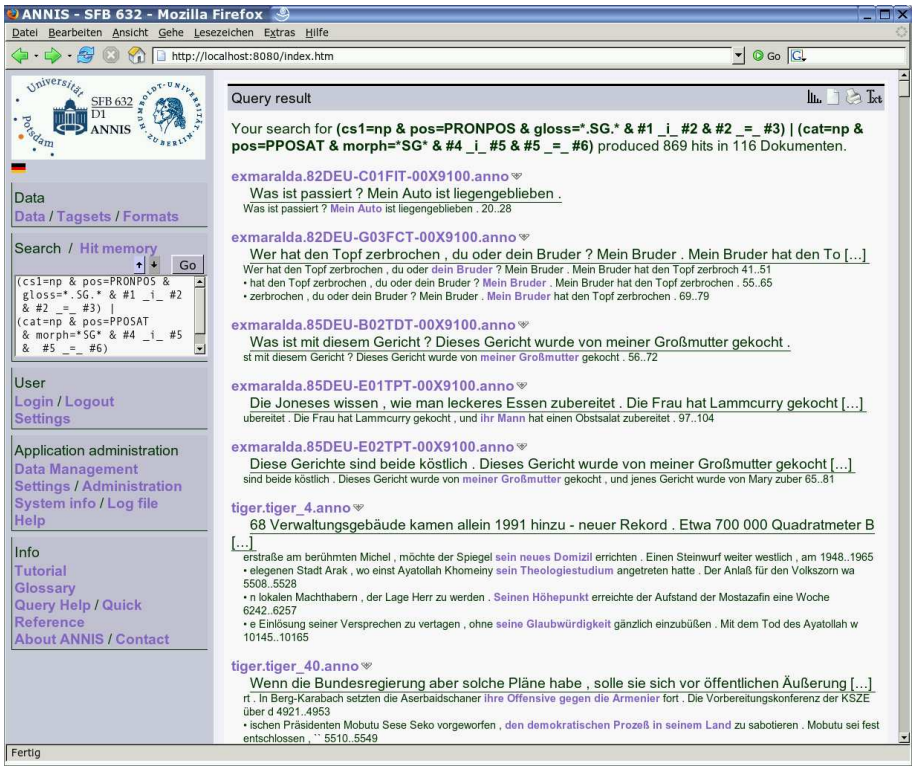

Figure 6. *ANNIS screenshot, displaying a query (small window in the left menu) and the corresponding results listed in the main window.*

ANNIS is a web application that is available both as a standalone version on a local computer (e.g., for fieldwork with a laptop) and as a server-based version. In both cases, it is accessible with standard web browsers, see Figure 6. Its query language ANNIS-QL builds upon widely used query languages employed in TIGERSearch[10] or CQP[11], allowing for relatively straightforward query formulation by users. While the standalone version of ANNIS operates on the data in main memory, the server version employs a database backend for querying and visualization.

In the following sections, we focus on the facilities for querying multi-level corpora in ANNIS. First, we illustrate the usage of our query language with the standalone version of ANNIS. Then, turning to the server version, we describe our approach to importing PAULA files into a relational database and executing ANNIS-QL queries

10. `http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERSearch`
11. `http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench`

on the data. Finally, we will show examples of concept-based querying, which relies on the ontology described in Section 4.

5.1. *ANNIS and its Query Language ANNIS-QL*

Similar to existing query languages, ANNIS-QL offers query operators for both hierarchical and sequential relations. The latter are of particular relevance for querying multi-level annotations, since sequential (or temporal) information often constitutes the only relationship between annotations of different annotation levels. The following is a simple query searching for nominal phrases beginning with a possessive singular pronoun:[12]

(5) `cs=np & pos=PRONPOS & gloss=*SG* & #1 _l_ #2 & #2 _=_ #3`

This query matches *ihr Mann* from Example (1). The feature names `cs`, `pos` and `gloss` match PAULA nodes (`struct` or `mark`) that have the corresponding label (`feat`). A corresponding query can also be formulated for the annotation according to the TIGER annotation as shown in Example (6). In this way, ANNIS-QL offers queries across different corpora.

(6) `cat=NP & pos=PPOSAT & morph=*Sg* & #1 _l_ #2 & #2 _=_ #3`

Moreover, the query language allows accessing different annotations of the same corpus, so that, for instance, competing analyses indicating disagreements between annotators (`ann1` and `ann2`) can be found, as in Example (7) with respect to the "givenness" of an item:

(7) `ann1::givenness=new & ann2::givenness=giv & #1 _=_ #2`

The negation operator '!' allows us to formulate queries that check for completeness of annotations. This is illustrated by Example (8), which checks (across layers) whether all referring expressions are annotated for the feature *givenness*.

(8) `aboutness=ref & !givenness=* & #1 _=_ #2`

12. The queries in ANNIS-QL specify constraints over the annotations (e.g. `cs=np` or `givenness=new`), optionally about their annotation set (`ann1::`, as in (7)), and their relations ('`_l_`' requires left alignment of both arguments; '`_=_`' states that both arguments refer to the same primary data). Every atom in an ANNIS-QL query, e.g., `cat=np`, introduces a variable of the form $\#n$, with n being its number in the sequence of atoms. The examples also show that wildcards can be used.

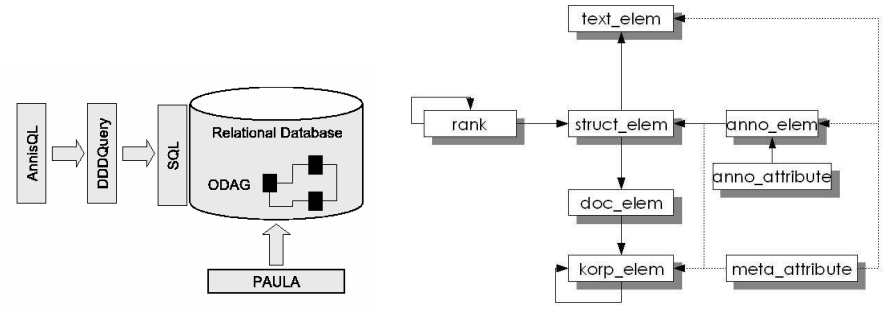

Figure 7. *Query Execution Architecture and Implementation Model of ANNIS server.*

5.2. *ANNIS server — A Relational Database Backend*

ANNIS-QL was designed as a concise and simple query language directly usable by linguists. Its current implementation is based on a software architecture which assumes that the corpus to be searched (i.e., a set of PAULA files) is loaded entirely into the main memory of the computer before any query evaluation takes place. During the loading of the corpus, the ANNIS-QL processor builds a set of main memory data structures which are later traversed for query execution. This is designed for research scenarios where users want to work efficiently with individual, relatively small corpora, which can also be stored on a laptop. For example, the aforementioned TIGER corpus consisting of 900,000 tokens can be handled in this way, and queries are processed very fast.

However, as soon as a corpus grows in size, traversing it entirely for answering a query becomes inefficient. We therefore developed a second implementation of ANNIS-QL which we call "ANNIS server", and which currently is in a prototypical stage. It builds on top of a relational database (we currently use PostGreSQL[13], a mature and open source relational database management system). Corpora in the PAULA format are loaded into a relational database that implements a slight variant of the DDD ODAG model (ordered directed acyclic graphs) described in Dipper *et al.* (2004) and Faulstich *et al.* (2005). The schema is shown in Figure 7. The mapping to the corresponding elements of PAULA and POM was given in Table 2 in Section 3.2.

This schema implements a meta-model based approach to the storage of structured linguistic annotations. Its fundamental elements are *structural elements* (in short: elements) which are associated to intervals of a text counted in tokens.[14] Each *element*

13. `http://www.postgresql.org`
14. These elements correspond to nodes in the PAULA object model, and `struct`, `mark`, or `tok` elements in the PAULA format.

has a type which is represented as an attribute. *elements* may dominate other *elements*, where the order in which the children of an element appear is fixed. This information is encoded in the *rank* table, which uses a method of indexing a tree structure in a relational database described in Georgiadis *et al.* (2007). The set of structured elements annotated in a text must be cycle-free, but may contain multiple root nodes. Using this model, we can represent various types of linguistic structures, including simple, token-associated annotation such as word lemmata or part-of-speech, and structured, potentially non-consecutive annotations such as phrases, multi-token entities, chunks, or syntax trees. *elements* may also be annotated with *meta annotations* regarding their source, author, date of creation, etc. Such meta annotations can be grouped together into *annotation sets*, which allows us to represent, for instance, different and diverging part-of-speech annotations for the very same text. Finally, texts may be grouped together into corpora. Thus a query may specifically be directed only to a fraction of all texts in the database. This meta-model also provides a very high level of flexibility and extensibility. Adding new types of annotation or new attributes to annotations does not require any changes in the relational schema, only in the set of values that are allowed in certain positions.

We presented a language, DDDQuery, for querying linguistic data stored in the ODAG model in Faulstich *et al.* (2006). DDDQuery is a language that extends XPath by various new operators to handle DAGs (because XPath can only handle tree structures), and to enable typical linguistic query predicates that are not present in XPath. However, DDDQuery is a rather complex and verbose language that is not suited for (and was never meant for) being used by end users, i.e., linguists. Instead, its design was focused on enabling a fast translation of queries into efficient SQL programs which can be executed in relational databases.

We use DDDQuery as an intermediate language between ANNIS-QL and the database backend. This architecture, which is shown in Figure 7, has the advantage that we did not have to develop a low-level translation of ANNIS-QL into SQL but only one from ANNIS-QL into DDDQuery. This was considerably simpler, since both languages target linguistic data and therefore share many predicates. However, the double translation comes at a certain price, i.e., time used for translating queries. But we found this price to be very small compared to the time it takes to evaluate complex queries.

We give two examples of this two-step translation. First, consider a query searching for all occurrences of the token *sein* (‘his’) as a possessive pronoun in the TIGER/STTS annotation scheme. This query is expressed in ANNIS-QL in the following form:

(9) `pos=PPOSAT & ‘‘sein’’ & #1 _=_ #2`

This query is automatically translated into the following DDDQuery:

(10) `ANNO[@pos = ’PPOSAT’]/STRUCT/element-span::"sein"`

The query may be read from left to right. It first searches for all structured elements with part-of-speech annotation "PPOSAT". From all such instances, it traverses the annotation graph stored in the ODAG model. All elements that are not dominated by a `struct` element associated to a token *sein* are discarded, while all others are returned.

As a second example, consider a more elaborate query searching for all occurrences of the token *sein* as part of the subject of a sentence. In ANNIS-QL, this is conveniently expressed as

(11)
```
rel=sb & "sein" & #1 _i_ #2
```

The corresponding DDDQuery describes precisely how matches of the query can be found in the ODAG model:

(12)
```
whole-text::"sein"/overlapping::STRUCT#(t1)$t1 &
ANNO#(a2)[@rel = 'sb']/STRUCT#(t2)$t2 &
$t1/overlapping::$t2
```

The query first identifies all occurrences of *sein* anywhere in the corpus. The variable $t1$ is bound to all structures overlapping any such occurrence. In the next clause, variable $t2$ is bound to all structures that are annotated as subjects (`sb`). Finally, the third clause combines the results of the previous two clauses by filtering only those bindings of $t1$ which are overlapped by bindings of $t2$.

In the second step, DDDQueries are translated into SQL queries which are executed by the database backend. Thus, memory management is handled by the server, as is optimization of the SQL queries. Note that such queries are rather long; for instance, the SQL query for the first example has seven joins, and the SQL query for the second example has 14. Despite this complexity, our experiences show that these queries are optimized very well by the relational engines and are answered very fast. However, we have not yet performed sufficient testing on really large corpora to prove the scalability of this approach.

5.3. *Concept-based Corpus Querying*

Examples (5) and (6) in Section 5.1 show that due to the generic representation of PAULA, quasi-identical ANNIS-QL queries can be applied to originally different input formats (LISA and TIGER respectively). The parallelism of both queries is illustrated in (13), a merged version of both queries.

(13)
$$\left\{\begin{array}{c}\texttt{cs}\\ \texttt{cat}\end{array}\right\} \texttt{= NP \& pos =} \left\{\begin{array}{c}\texttt{PRONPOS}\\ \texttt{PPOSAT}\end{array}\right\} \texttt{\&} \left\{\begin{array}{c}\texttt{gloss}\\ \texttt{morph}\end{array}\right\} \texttt{=} \left\{\begin{array}{c}\texttt{*SG*}\\ \texttt{*Sg*}\end{array}\right\} \texttt{\&}$$
```
#1 _l_ #2 & #2 _=_ #3
```

The conceptual integration of the different annotation schemes now allows us not only to generalize over different structures annotated in the original format, but also to formulate a single search query for both tasks, since the ontology described in Section 4 not only provides information about tag names (`hasTag` property) and layer identifiers (`hasTier` property), but also links concrete annotations with the reference terminology specified in the Reference Model. For cases where users are searching across different corpora or are not sure of the tags for a certain annotation concept (see Section 4), we provide for more abstract queries. A query preprocessor retrieves all tag descriptions that correspond to an ontological description and translates them into a disjunction of specific annotation values. If multiple annotation schemes are considered, such a description may be expanded into a disjunction of tags from different tag sets and/or tiers.

Ontology-sensitive sub-queries are composed according to the following context-free grammar[15]:

ONTOQUERY := CUE `in` {ONTOEXP}
ONTOEXP := ONTOCONCEPT |
(ONTOEXP ONTOOP ONTOEXP) |
ONTOPROPERTY(ONTOFEATURE)
ONTOOP := `and` | `or` | `without`

In this way, multiple queries for part-of-speech tags from different annotation schemes can be replaced by a single ontology-sensitive corpus query. Query (13) for NPs containing possessive pronouns can thus be abbreviated as in (14).

(14)
```
cat in {NounPhrase} & pos in {PossessivePronoun} & morph in
{hasNumber(Singular)} & #1 _l_ #2 & #2 _=_ #3
```

The CUE expressions `cat`, `pos`, and `morph` are then replaced by the values of the `hasTier` property, the ONTOEXP expressions by the corresponding `hasTag` values. Thus (13) translates into a regular ANNIS-QL query in which different alternative tags and layer identifiers are represented by means of a disjunction that also covers the sub-queries (5) and (6).

As opposed to working with complex regular expressions, this ontology-driven tag expansion allows the user to generalize over the specific form of annotations and tag names, requiring only a conceptualization of the search task rather than detailed information about the principles of tag name formation.

15. ONTOCONCEPT, ONTOPROPERTY and ONTOFEATURE correspond to word classes, properties and grammatical features specified in the reference model. ONTOQUERYs can be embedded in arbitrary code that remains untouched during query expansion.

6. Annotation Mining

In the previous sections, we explained different aspects of integrating and querying linguistic corpora. This section will now give an example showing how these resources can be used to make the annotation process more efficient.

With corpora annotated on multiple layers, we expect to profit from using machine-learning methods in two ways: (i) detecting interdependencies between layers, and (ii) semi-automating the annotation process. For these purposes, we built a component that maps our pivot format PAULA (see Section 3) to the Attribute Relation File Format (ARFF) used in WEKA (Witten *et al.*, 2005), an open source data mining software.

In a preprocessing step, the data is enriched by adding to each part-of-speech tag its corresponding direct superclass(es) from the ontological reference model (Section 4). For the export, it is necessary for the user to choose the elementary unit (e.g., token, noun phrase (NP), or sentence; in the following, we assume NPs). For each instance i, one data set will be formed. Then the user can assign the annotation levels to three different categories: (i) feature/value pairs *directly annotated* to i (i.e., annotations making use of the same markables as the basic unit), (ii) annotations extending to a *part of* i (e.g., part-of-speech tags within an NP), and (iii) annotations whose extension may *include* the extension of i (e.g., the focus of a sentence containing the NP i).[16] Features of categories (ii) and (iii) are represented in B-I-O notation (Ramshaw *et al.*, 1995), where B stands for 'at the beginning', I for 'in' and O for 'outside the phrase'. In the case of phrases, we also compute the length of each instance (measured in tokens).

When using WEKA, one typically trains classifiers of some type, such as support vector machines or decision trees. Some preliminary results for a classification of NPs with respect to their information status[17] are presented in Table 3, and part of a sample decision tree is given in Figure 8[18]. The training data originate from the Potsdam Commentary Corpus (Stede, 2004). From the selection of features and their prominence in the decision tree, we find that lexical choice (pronoun vs. full nominal phrase)—represented by part-of-speech tags in a generalized way—is indeed an indicator for recognizing information status.

Finally, the results of the classification are re-imported to the pivot format and can then be presented to human annotators for correction.

16. The current implementation works on spans (extensions) of annotations. It works for annotations in the form of feature/value pairs and labeled edges; links, sets, etc. are not covered.
17. According to the LISA scheme, a referential NP is either `giv`(en) (previously mentioned in the context), `acc`(essible) (inferable from the utterance situation or from the context via bridging), or else `new`.
18. Abbreviations used in Figure 8: (non)ref = (non)referential; con = contains; in = included in; onto.pos = POS superclass (from the ontology); tiger.cat = constituent category (TIGER scheme); AP = adjective phrase; S = sentence.

classifier		correctly classified	
name	strategy	absolute	percent
ZeroR	predict most frequent value ('new')	1335	38.63%
OneR	prediction depends on phrase length	1797	52.00%
J48 (C4.5)	decision tree (see Figure 8)	2217	64.15%
size of training set		3456	NPs
All experiments evaluated using 10-fold cross validation.			

Table 3. *Classification results for information status of NPs.*

```
length ≤ 3
| con_onto_pos_PersonalPronoun = B
| | con_tiger.pos_NN = B: giv
| | con_tiger.pos_NN = I: acc
| | con_tiger.pos_NN = O ...
...
| con_onto_pos_PersonalPronoun = I
| | length ≤ 2:  giv
| | length > 2:  new
| con_onto_pos_PersonalPronoun = O
| | con_onto_pos_PossessivePronoun = B
| | | con_tiger.pos_NN = I: acc
...
```

Figure 8. *Sample decision tree (excerpt) for information status of NPs.*

7. Summary and Discussion

We have given an overview of our implemented software environment for producing multi-layer annotated corpora: a pivot format serving as "interlingua" between annotation tools, an ontology-based approach for mapping between tag sets, and an information system that integrates the various annotations, and allows for querying the data (either by posing simple queries or by using the ontology) and for statistical analyses.

Our approach is related to other recent approaches aiming to integrate annotations from different source formats, in particular to NITE and LAF. Both operate on the basis of standoff XML pivot formats, as does our format. However, the NITE Object Model operates on the basis of multi-rooted trees, whereas our data model (POM) also

specifies the semantics of pointing relations. Consequently, our underlying data base implementation is based on a relational database rather than an XML database. GRaF, on the other hand, the pivot format of LAF, basically operates upon general graph structures, and is therefore not specifically optimized for the processing of linguistic annotations. In this sense, our approach is more specific to linguistic annotations, though still representing a highly generic level of description upon which any annotation of textual data can be represented.

Our approach also integrates the OLiA ontologies as a terminological reference that specifies the semantics of different annotations with respect to the OLiA Reference Model, to GOLD, or to the Data Category Registry (DCR) that is developed as a component of the Linguistic Annotation Framework. However, as compared to the direct mapping between the DCR and concrete annotations, our ontological linking allows for greater expressivity, including the set operators $\cup$, $\cap$, and $\setminus$, which may be used to constitute complex reference concepts. As compared to other approaches that involve the direct transformation of annotations in order to map onto reference concepts (de Cea *et al.*, 2004; Farrar *et al.*, 2003), the formalization of the linking as RDF descriptions allows the application of standard OWL editors, and is thus more transparent, modifiable, and scalable than implementation-specific scheme transformation rules.

Our framework supports linguists in using the most suitable (XML-based) annotation tools for their specific purposes, and allows for combining the different, possibly quite heterogeneous annotations into the same database. In order to cover different application scenarios, we have developed two versions of the ANNIS information system. One is a standalone version where all data resides in main memory, leading to very efficient query execution. For larger corpora, we have also built a server-based version on top of a standard relational database. The ATLAS, NITE, and LAF projects, in principle following similar goals, do currently not involve a database implementation. Instead, these approaches focus on the development of libraries for corpus processing (e.g., Carletta *et al.* (2005)).

Our conversion tools (to and from the pivot format) and the ANNIS system are freely available for research purposes—see the URLs given in Footnotes 3 and 9. In future work, we plan to improve especially the visualization capabilities of ANNIS, which at present are restricted to a straightforward layer-oriented presentation of annotations.

Finally, we wish to draw attention to the methodological issues of multi-layer architectures. As we pointed out, in general they provide new possibilities for an in-depth analysis of linguistic data by allowing multiple independent annotation layers. This will certainly have interesting implications for the qualitative and quantitative analysis of linguistic data, but at the same time it requires thorough research on the particular evaluation possibilities. Technically, it is easily possible to search across annotation layers. But conceptually, annotation layers are often not independent of each other (information structure, for instance, is dependent on certain syntactic configurations) and therefore simple statistical analyses might not always be possible. One

way of exploring these interdependencies is to use multi-dimensional techniques (see, e.g, Moisl (to appear)). And a special use of multi-layer architectures is the annotation of conflicting analyses for the same linguistic level of analysis (such as different part-of-speech tag sets or different syntactic annotations). This will be especially interesting for "non-standard" language (such as historical language, dialects, or learner language) where annotation standards are strongly contested or not yet well developed. To mention just one example, Lüdeling (2008) shows how strongly individual analyses can influence the empirical basis for theory building: different interpretations of the same learner data lead to error rates that differ by 100%. Thus, an important aspect of the meta data in multi-level corpora should be the provenance of the annotations and the possibilities of dependencies, which need to be taken into account when drawing conclusions from the data.

8. References

Albert S., Anderssen J., Bader R., Becker S., Bracht T., Brants S., Brants T., Demberg V., Dipper S., Eisenberg P., Hansen S., Hirschmann H., Janitzek J., Kirstein C., Langner R., Michelbacher L., Plaehn O., Preis C., Pußel M., Rower M., Schrader B., Schwartz A., Smith G., Uszkoreit H., TIGER Annotationsschema, Technical report, Universities of Saarbrücken, Stuttgart, and Potsdam, 2003.

Atwell E., Hughes J., Souter C., "AMALGAM: Automatic Mapping Among Lexico-Grammatical Annotation Models", *in* J. Klavans, P. Resnik (eds), *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, Association for Computational Linguistics, Las Cruces, NM, USA, p. 21-28, 1994.

Baumann S., Brinckmann C., Hansen-Schirra S., Kruijff G.-J., Kruijff-Korbayová I., Neumann S., Teich E., "Multi-dimensional annotation of linguistic corpora for investigating information structure", *Proceedings of the Workshop Frontiers in Corpus Annotation at HLT-NAACL 2004*, Boston, MA, USA, p. 39-46, 2004.

Bird S., Liberman M., "A formal framework for linguistic annotation", *Speech Communication*, vol. 33, n° 1-2, p. 23-60, 2001.

Brants S., Dipper S., Eisenberg P., Hansen S., König E., Lezius W., Rohrer C., Smith G., Uszkoreit H., "TIGER: Linguistic Interpretation of a German Corpus", *Research on Language and Computation*, vol. 2, n° 4, p. 597-620, 2004.

Brants T., Plaehn O., "Interactive Corpus Annotation", *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC-2000)*, Athens, Greece, p. 453-459, 2000.

Carletta J., Evert S., Heid U., Kilgour J., "The NITE XML Toolkit: data model and query", *Language Resources and Evaluation Journal (LREJ)*, vol. 39, n° 4, p. 313-334, 2005.

Carletta J., Evert S., Heid U., Kilgour J., Robertson J., Voormann H., "The NITE XML Toolkit: flexible annotation for multi-modal language data", *Behavior Research Methods, Instruments, and Computers*, vol. 35, n° 3, p. 353-363, 2003a.

Carletta J., Kilgour J., O'Donnell T., Evert S., Voormann H., "The NITE Object Model Library for Handling Structured Linguistic Annotation on Multimodal Data Sets", *Proceedings of*

the EACL Workshop on Language Technology and the Semantic Web (3rd Workshop on NLP and XML, NLPXML-2003), Budapest, Hungary, April, 2003b.

Chiarcos C., "An Ontology for Heterogeneous Data Collections", *Proc. of the Int. Conference 'Corpus Linguistics 2006', St. Petersburg*, St. Petersburg University Press, p. 373-380, 2006.

Chiarcos C., "An ontology of linguistic annotations", *GLDV-Journal for Computational Linguistics and Language Technology*, to appear.

Cunningham H., Maynard D., Bontcheva K., Tablan V., "GATE: A framework and graphical development environment for robust NLP tools and applications", *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, Philadelphia, PA, USA, p. 168-175, 2002.

de Cea G. A., Gómez-Pérez A., Álvarez de Mon I., Pareja-Lora A., "OntoTag's Linguistic Ontologies: Improving Semantic Web Annotations for a Better Language Understanding in Machines", *ITCC '04: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04) Volume 2*, IEEE Computer Society, Washington, DC, USA, p. 124-128, 2004.

Dipper S., Faulstich L., Leser U., Lüdeling A., "Challenges in Modelling a Richly Annotated Diachronic Corpus of German", *Workshop on XML-based richly annotated corpora*, Lisbon, Portugal, p. 21-29, May, 2004.

Dipper S., Götze M., Skopeteas S., (eds.), Information Structure in Cross-Linguistic Corpora: Annotation Guidelines for Phonology, Morphology, Syntax, Semantics, and Information Structure, Technical report, University of Potsdam, 2007.

Dybkjær L., Bernsen N. O., Dybkjær H., McKelvie D., Mengel A., The MATE Markup Framework. MATE Deliverable D1.2, Technical report, University of Southern Denmark, 1998.

Eeg-Olofsson M., Word-class tagging: Some computational tools, PhD thesis, Department of Linguistics and Phonetics, University of Lund, Sweden, 1991.

Erjavec T., "MULTEXT-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora", *Fourth International Conference on Language Resources and Evaluation, LREC'04*, Paris, France, p. 1535-1538, 2004.

Erk K., Pado S., "A powerful and versatile XML format for representing role-semantic annotation", *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*, Lisbon, Portugal, 2004.

Farrar S., Langendoen D. T., "A Linguistic Ontology for the Semantic Web", *GLOT International*, vol. 7, p. 97-100, 2003.

Faulstich L. C., Leser U., Lüdeling A., Storing and Querying Historical Texts in a Database, Technical Report n° 176, Institut für Informatik der Humboldt-Universität zu Berlin, January, 2005.

Faulstich L., Leser U., Vitt T., "Implementing a Linguistic Query Language for Historic Texts", *Proceedings of the International Workshop on Query Languages and Query Processing*, München, Germany, p. 601-612, 2006.

Georgiadis H., Vassalos V., "XPath on steroids: exploiting relational engines for XPath performance", *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, ACM, New York, NY, USA, p. 317-328, 2007.

Götze M., Skopeteas S., Roloff T., Stoel R., "Towards a Cross-Linguistic Production Data Archive: Structure and Exploration", *in* B. ten Cate, H. Zeevat (eds), *TbiLLC*, vol. 4363 of *Lecture Notes in Computer Science*, Springer, p. 127-138, 2005.

Greenbaum S., *The ICE tagset manual*, University College London. 1992.

Greene B. B., Rubin G. M., *Automatic grammatical tagging of English*, Department of Linguistics, Brown University, Providence, RI, USA. 1981.

Ide N., Romary L., "A Registry of Standard Data Categories for Linguistic Annotation", *Proceedings of the Fourth International Language Resources and Evaluation Conference (LREC)*, Lisbon, Portugal, p. 135-139, 2004.

Ide N., Romary L., de la Clergerie E., "International Standard for a Linguistic Annotation Framework", *Proceedings of HLT-NAACL'03 Workshop on the Software Engineering and Architecture of Language Technology*, Edmonton, Canada, p. 25-30, 2003.

Ide N., Suderman K., "GrAF: A Graph-based Format for Linguistic Annotations", *Proceedings of The Linguistic Annotation Workshop (LAW) 2007*, Prague, Czech Republic, p. 1-8, 2007.

König E., Lezius W., "A description language for syntactically annotated corpora", *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, Saarbrücken, Germany, p. 1056-1060, 2000.

Laprun C., Fiscus J. G., Garofolo J., Pajot S., "A practical introduction to ATLAS", *Proceedings of LREC 2002*, Las Palmas, Spain, 2002.

Leech G., Wilson A., *EAGLES Recommendations for the Morphosyntactic Annotation of Corpora*, Istituto di Linguistica Computazionale, Pisa, Italy. 1996.

Lüdeling A., "Mehrdeutigkeiten und Kategorisierung. Probleme bei der Annotation von Lernerkorpora", *in* M. Walter, P. Grommes (eds), *Fortgeschrittene Lernervarietäten*, Niemeyer, Tübingen, p. 119-140, 2008.

Marcus M., Santorini B., Marcinkiewicz M. A., "Building a large annotated corpus of English: the Penn Treebank", *Computational Linguistics*, vol. 19, n° 2, p. 313-330, 1993.

Moisl H., "Exploratory multivariate analysis", *in* A. Lüdeling, M. Kytö (eds), *Corpus Linguistics. An International Handbook*, Mouton de Gruyter, Berlin, Germany, to appear.

Müller C., Strube M., "Multi-level annotation of linguistic data with MMAX2", *in* S. Braun, K. Kohn, J. Mukherjee (eds), *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*, Peter Lang, Frankfurt, Germany, 2006.

O'Donnell M., "RSTTool 2.4 – A Markup Tool for Rhetorical Structure Theory", *Proceedings of the International Natural Language Generation Conference (INLG'2000)*, Mitzpe Ramon, Israel, p. 253-256, 2000.

Orasan C., "PALinkA: a highly customisable tool for discourse annotation", *Proceedings of the 4th SIGdial Workshop on Discourse and Dialogue*, Sapporo, Japan, p. 39-43, 2003.

Pustejovsky J., Meyers A., Palmer M., Poesio M., "Merging PropBank, NomBank, TimeBank, Penn Discourse Treebank and Coreference", *Proceedings of ACL Workshop on Frontiers in Corpus Annotation 2005*, Ann Arbor, MI, USA, 2005.

Ramshaw L. A., Marcus M. P., "Text Chunking using Transformation-based Learning", *Proceedings of the Third ACL Workshop on Very Large Corpora*, Cambridge, MA, USA, 1995.

Sampson G., *English for the Computer. The SUSANNE Corpus and Analytic Scheme*, Clarendon, Oxford, 1995.

Santorini B., *Part-of-speech tagging guidelines for the Penn Treebank Project*, Department of Computer and Information Science, University of Pennsylvania. 1990.

Schiller A., Teufel S., Stöckert C., Thielen C., Guidelines für das Tagging deutscher Textcorpora mit STTS (Kleines und großes Tagset), Technical report, Universities of Stuttgart and Tübingen, 1999.

Schmidt T., "EXMARaLDA – ein System zur computergestützten Diskurstranskription", *in* A. Mehler, H. Lobin (eds), *Automatische Textanalyse. Systeme und Methoden zur Annotation und Analyse natürlichsprachlicher Texte*, Verlag für Sozialwissenschaften, Wiesbaden, Germany, p. 203-218, 2004.

Schmidt T., "EXMARaLDA und die Datenbank "Mehrsprachigkeit" — Konzepte und praktische Erfahrungen", *in* S. Dipper, M. Götze, M. Stede (eds), *Heterogeneity in Focus: Creating and Using Linguistic Databases*, ISIS (Interdisciplinary Studies on Information Structure), Working Papers of the SFB 632, Universitätsverlag Potsdam, 2005.

Souter C., A Short Handbook to the Polytechnic of Wales Corpus, Technical report, ICAME, Norwegian Computing Centre for the Humanities, Bergen University, Norway, 1989.

Sperberg-McQueen C. M., Bernard L. (eds), *Guidelines for Electronic Text Encoding and Interchange (TEI P3)*, Text Encoding Initiative, Chicago, Oxford, 1994.

Stede M., "The Potsdam Commentary Corpus", *Proceedings of the ACL Workshop on Discourse Annotation*, Barcelona, Spain, p. 96-102, 2004.

Trißl S., Zipser F., Leser U., "Applying GRIPP to XML Documents containing XInclude and XLink Elements", *Proceedings of the XML Tage*, Berlin, Germany, 2007.

Witt A., Goecke D., Sasaki F., Lüngen H., "Unification of XML Documents with Concurrent Markup", *Literary and Linguistic Computing 2005*, vol. 20, p. 103-116, 2005.

Witten I. H., Frank E., *Data mining: Practical machine learning tools and techniques*, 2nd edn, Morgan Kaufman, San Francisco, 2005.

Wittenburg P., "Preprocessing multimodal corpora", *in* A. Lüdeling, M. Kytö (eds), *Corpus Linguistics. An International Handbook*, Mouton de Gruyter, Berlin, Germany, to appear.

ANNEXE POUR LE SERVICE FABRICATION
A FOURNIR PAR LES AUTEURS AVEC UN EXEMPLAIRE PAPIER
DE LEUR ARTICLE ET LE COPYRIGHT SIGNE PAR COURRIER
LE FICHIER PDF CORRESPONDANT SERA ENVOYE PAR E-MAIL

1. ARTICLE POUR LA REVUE :

Traitement Automatique des Langues. Volume 49 – n° 2/2008

2. AUTEURS :

Christian Chiarcos[*] *— Stefanie Dipper*[**] *— Michael Götze*[*]
Ulf Leser[***] *— Anke Lüdeling*[****] *— Julia Ritz*[*] *— Manfred Stede*[*]

3. TITRE DE L'ARTICLE :

A Flexible Framework for Integrating Annotations from Different Tools and Tag Sets

4. TITRE ABRÉGÉ POUR LE HAUT DE PAGE MOINS DE 40 SIGNES :

Framework for Integrating Annotations

5. DATE DE CETTE VERSION :

April 27, 2010

6. COORDONNÉES DES AUTEURS :

– adresse postale :
[*] Institut für Linguistik, Universität Potsdam
Karl-Liebknecht-Str. 24-25 – D-14476 Golm
(chiarcos|goetze|julia|stede)@ling.uni-potsdam.de

[**] Sprachwissenschaftliches Institut, Ruhr-Universität Bochum
Universitätsstr. 150 – D-44801 Bochum
dipper@linguistics.rub.de

[***] Institut für Informatik, Humboldt-Universität zu Berlin
Unter den Linden 6 – D-10099 Berlin
leser@informatik.hu-berlin.de

[****] Institut für deutsche Sprache und Linguistik, Humboldt-Universität zu Berlin
Unter den Linden 6 – D-10099 Berlin
anke.luedeling@rz.hu-berlin.de
– téléphone : 00 00 00 00 00
– télécopie : 00 00 00 00 00
– e-mail : guillaume.laurent@ens2m.fr

7. LOGICIEL UTILISÉ POUR LA PRÉPARATION DE CET ARTICLE :

LaTeX, avec le fichier de style `article-hermes.cls`,
version 1.23 du 17/11/2005.

SERVICE ÉDITORIAL – HERMES-LAVOISIER
14 rue de Provigny, F-94236 Cachan cedex
Tél. : 01-47-40-67-67
E-mail : revues@lavoisier.fr
Serveur web : http://www.revuesonline.com