Accessing Heterogeneous Linguistic Data — Generic XML-based Representation and Flexible Visualization

Stefanie Dipper and Michael Götze

University of Potsdam
Dept. of Linguistics, Computational Linguistics
D-14415 Potsdam, Germany
{dipper,goetze}@ling.uni-potsdam.de

Abstract

Annotation of linguistic data increasingly focuses on information beyond the (morpho-)syntactic level. Moreover, annotated data of less-studied languages is growing in importance. To maximally profit from this data, straightforward and user-friendly access has to be provided. In this paper, we describe a linguistic database that is accessed via a web browser and offers flexible visualization of multiply annotated data. Data is internally represented by a generic XML-based format.

1. Introduction

In recent years, interest in linguistic data that is annotated beyond the (morpho-)syntactic level has increased. For instance, text corpora are annotated with semantic and pragmatic information, such as predicate-argument structure and word sense information (PropBank (Kingsbury and Palmer, 2002), FrameNet (Johnson et al., 2003), SALSA (Erk et al., 2003)), discourse and information structure (Penn Discourse TreeBank (Miltsakaki et al., 2004), RST Discourse Treebank (Carlson et al., 2003), Potsdam Commentary Corpus (Stede, 2004), MULI (Baumann et al., 2004)) and dialogue structure (Switchboard SWBD-DAMSL (Jurafsky et al., 1997)).[1]

In many cases, linguistic information is added to already-annotated corpora, as is the case, e.g., with the Penn Treebank, whose syntactic annotation has been supplemented by discourse (RST and Penn Discourse TreeBank) and semantic information (PropBank). Such multi-level annotation allows for research on complex phenomena which operate on several linguistic levels simultaneously.

Moreover, annotated data of less-studied languages is gaining importance, and researchers try to base their work on data of typologically diverse languages.

As a result, we are faced with a considerable amount of rather small corpora of many different languages. They are usually annotated manually (or semi-automatically), are created for different purposes and, accordingly, vary with regard to their annotation.

Such corpora are valuable resources and may serve other users and applications as well if the data is represented in a standardized format and if documentation of the annotation scheme and guidelines is provided, thus allowing for informed interpretation of the annotation.

In addition, in order to manually inspect and interpret the data (as opposed to with automatic applications), straightforward and user-friendly access is a prerequisite. This involves transparent visualization of the text and its annotation as well as an intuitive search facility.

In this paper, we present a linguistic database that addresses these requirements. It can host multiply annotated data, is accessed via a web browser, and offers flexible visualization. Data is internally represented by a generic XML-based format.

The database *ANNIS* is being developed in the Collaborative Research Center SFB 632 on Information Structure.[2] In the context of this research center, data of various languages is collected and annotated at the levels of phonology, morphology, syntax, semantics, and pragmatics—levels that contribute in ways yet to be determined to the information structural partitioning of discourse and utterances.

The paper is organized as follows. We first describe the need for easy data access by different users (sec. 2.). We then present the internal representation format and visualization techniques of our database (sec. 3.) and compare it to related work (sec. 4.). A conclusion and directions for future development follow in sec. 5.

2. Motivation

Three factors emerge from the scenario sketched out above that we consider especially important:

Data heterogeneity results from a number of sources: First, primary data itself is heterogeneous, differing with respect to size (e.g., single sentences vs. journal articles), modality (monologue vs. dialogue, text vs. speech), and language. Second, rich annotations often require data structures of various types (attribute-value pairs, trees, pointers, etc.). And finally, data is often annotated by means of different, task-specific annotation tools (e.g., by tools for syntax, discourse, or co-reference annotation).

Data reuse To maximize profit from available corpora, data must be reusable, even for purposes which are different from the original ones. To this end, it must be possible

[1] In this paper, we focus on text corpora, leaving aside the large field of multi-modal corpora, which raise specific problems of their own.

[2] `http://www.sfb632.uni-potsdam.de/`. For more information about ANNIS, see `http://www.sfb632.uni-potsdam.de/annis/` and (Dipper et al., 2004).

text.xml:

```
...</header>
<body>Fürchtet euch nicht !  Die einstige Fußball-Weltmacht zittert vor einem Winzling .  Mit seinem Tor zum
1:0 für die Ukraine stürzte der 1,62 Meter große Gennadi Subow die deutsche Nationalelf...</body>
```

tok.xml:

```
<markList type="token" xml:base="text.xml">
  <mark id="t1" xlink:href="#xpointer(string-range(//body,'',0,8))"/> <!-- Fürchtet -->
  <mark id="t2" xlink:href="#xpointer(string-range(//body,'',9,4))"/> <!-- euch -->
  <mark id="t3" xlink:href="#xpointer(string-range(//body,'',14,5))"/> <!-- nicht -->
  <mark id="t4" xlink:href="#xpointer(string-range(//body,'',20,1))"/> <!-- !  -->
  <mark id="t5" xlink:href="#xpointer(string-range(//body,'',22,3))"/> <!-- Die -->
  ...
```

Figure 1: Source text (*text.xml*) and `<mark>` tags specifying tokens (*tok.xml*)

inf_status.xml:

```
<featList type="inf_status" xml:base="tok.xml">
  ...
  <!-- euch: brand-new -->
  <feat xlink:href="#xpointer(id('t2'))" value="type_istat.xml#type2"/>
  <!-- Die einstige Fußball-Weltmacht: unused -->
  <feat xlink:href="#xpointer(id('t5')/range-to(id('t7')))" value="type_istat.xml#type3"/>
  ...
```

type_istat.xml:

```
<typeList type="inf_status">
  <type id="type1" name="discourse-new" descr="The referent is new in the discourse.">
    <type id="type2" name="brand-new" descr="The referent is new to the hearer."/>
    <type id="type3" name="unused"    descr="The referent is known to the hearer."/>
    ...
  </type>
  <type id="type4" name="discourse-old" descr="The referent is old in the discourse.">
    ...
```

Figure 2: Annotation of information status: anchoring to token markables (*inf_status.xml*) and definition in a type hierarchy (*type_istat.xml*)

to focus on specific annotations (e.g., by filtering out irrelevant ones) or to present them in different ways, tailored to specific tasks.

Data accessibility Accessibility of the corpus data for the end user—i.e., linguists interested in the data—is a major aim of a variety of visualization and query tools. However, current tools either do not provide functionality that is sufficiently flexible to cope with the heterogeneous data and varying purposes as described above.[3] Or else, they address users with at least basic programming skills in languages such as Java or XML/XSLT, who then have to adapt pre-existing programs or stylesheets to their needs or, in case of highly user-specific requirements, start from scratch.[4] Data access with user-adaptable functionality is thus restricted to expert users, which we believe to be an unsatisfying state of the art.

3. A Generic Approach

Our approach to representing heterogeneous data is based on a generic XML format, which serves as a kind of interlingua. Outputs of different tools are mapped to this standardized, abstract XML format. Customizable stylesheets read the XML representations and generate different views of the data.

3.1. XML Representation Format

In our approach, data is internally represented by a generic XML format, similar to the Generic Mapping Tool (GMT) proposed by (Ide and Romary, 2001). In contrast to initiatives such as XCES[5] or MATE (Dybkjær et al., 1998), our format does not propose actual tags such as `<s>` (for "sentence") or `<dys>` ("dysfluencies"). It rather provides generic XML elements like `<mark>` (markable), `<feat>` (feature), and `<struct>` (structure), which indicate which data type the annotation conforms to. These tags allow us to represent all kinds of linguistic annotation in a uniform way. The representations standardized this way then serve as the input to the visualization module (see next section).

We assume that primary data is stored in a file that optionally specifies a header, followed by a tag `<body>`, which contains the source text. Annotations are stored in separate files (stand-off), which allows for overlapping hierarchies and competing, alternative annotations. We distinguish three different types of annotations:

(i) Markables `<mark>` tags specify text positions or spans of text (or spans of other markables) that can be an-

[3] Examples of such tools are EXMARaLDA, TIGERSearch, or RSTTool.

[4] The NITE XML Toolkit is an instance of such a tool.

[5] `http://www.cs.vassar.edu/XCES/`

const.xml:

```
<structList type="const">
  ...
  <struct id="c2" > <!-- Die einstige Fußball-Weltmacht -->
    <rel id="r4" xlink:href="tok.xml#t5" type="undir"/> <!-- Die -->
    <rel id="r5" xlink:href="tok.xml#t6" type="undir"/> <!-- einstige -->
    <rel id="r6" xlink:href="tok.xml#t7" type="undir"/> <!-- Fußball-Weltmacht -->
  </struct>
  ...
```

syntax.xml:

```
<featList xml:base="const.xml">
  ...
  <feat xlink:href="#c2" type="cat" value="type_cat.xml#type5"/> <!-- Die einstige Fußball-Weltmacht: NP -->
  <feat xlink:href="#r4" type="func" value="type_func.xml#type8"/> <!-- Die: NK -->
  <feat xlink:href="#r5" type="func" value="type_func.xml#type8"/> <!-- einstige: NK -->
  <feat xlink:href="#r6" type="func" value="type_func.xml#type8"/> <!-- Fußball-Weltmacht: NK -->
  ...
```

Figure 3: Syntactic annotation: anchoring to constituent markables (*const.xml*) and annotation by `<feat>` tags (*syntax.xml*)

notated by linguistic information. For instance, `<mark>` tags might indicate tokens by specifying ranges of the source text, cf. fig. 1.

(ii) Features `<feat>` tags specify information annotated to markables, which are referred to by xlink attributes. The type of information (e.g., "information status") is encoded by an attribute "type".

We adopt the idea of (Carletta et al., 2003) by assuming that admissible feature values (such as "discourse-new", marking information new in the discourse, or "NP", noun phrase) may be complex types and are organized in a type hierarchy. For instance, "unused" and "brand-new" can be subtypes of the more general type "discourse-new". `<feat>` tags point to some type in the hierarchy (which is stored separately), thus specifying the value of the annotated property, cf. fig. 2.[6]

This approach supports underspecified annotation by referring to less specific supertypes. Moreover, it allows us to neglect the distinction between attributes and their values—a distinction which is not always clear-cut. For instance, some tagset might comprise an attribute "information_structure" with values "topic" and "focus". In other tagsets, however, "topic" might be used as an attribute, with values such as "aboutness-topic" or "frame-setting_topic".

(iii) Structures `<struct>` tags are special types of markables. Similar to `<mark>` tags, they specify objects that then can serve as anchors for annotations. Whereas `<mark>` tags define simple types of anchors (flat spans of text or markables), a `<struct>` tag represents a complex anchor involving relations between arbitrarily many markables (including `<struct>` elements). Relations (`<rel>`) can be undirected or directed (= pointers).

Syntax trees are encoded by `<struct>` tags, each of which specifies a local subtree. For instance, the `<struct>` element in fig. 3 defines a subtree consisting of a mother node (`<struct id="c2">`) and three daughters (= the markables referenced by the xlink attributes). Further properties of the subtree, like category of the mother node, "cat", or functions of the daughters, "func", are encoded stand-off by `<feat>` tags, cf. fig. 3.[7]

`<struct>` elements may also be used to represent multiple analyses of ambiguous input. Competing analyses would be stored in separate files and an additional layer would mark these analyses as alternatives by means of `<struct>` elements.[8]

3.2. Visualization

Our implementation is a Java servlet application, which is accessed via a web browser. Its input consists of data in our generic XML format plus a format file specifying the appearance of the annotation. Its output is an HTML page, displaying the primary data and its annotation.

Visualization modes The system supports the concurrent visualization of different types of annotation. In fig. 4, data from a newspaper text—whose annotations were created using three different annotation tools[9]—is visualized

[6] For ease of readability, we simplify the presentation of the type hierarchy—type hierarchies do not necessarily form a tree, hence we actually represent them by `<struct>` elements, see below. The stand-off architecture allows the user to experiment with different hierarchies.

Type hierarchies have to be defined by the user or they may be derived from annotation schemes that incorporate hierarchies, cf. the schemes used by the annotation tool MMAX. In case no hierarchy is defined, the features are organized in a flat list.

[7] The examples in fig. 3 and 4 follow the annotation schemes of STTS (Schiller et al., 1999) and the TIGER project (`http://www.ims.uni-stuttgart.de/projekte/TIGER/`).

[8] Such extensive use of xlinks might be considered problematic for performance. In our context, however, the XML format mainly serves data visualization. That is, processing of the XML data concerns only parts of a corpus (e.g., one newspaper article out of a collection of articles) rather than entire corpora. Hence, corpus size does not directly influence the performance of the visualization module. In contrast, for data retrieval, data is translated into Java objects and searched by a Java-implemented query engine.

[9] Information structure was annotated with EXMARaLDA, co-reference with MMAX, and (morpho-)syntax with Annotate.

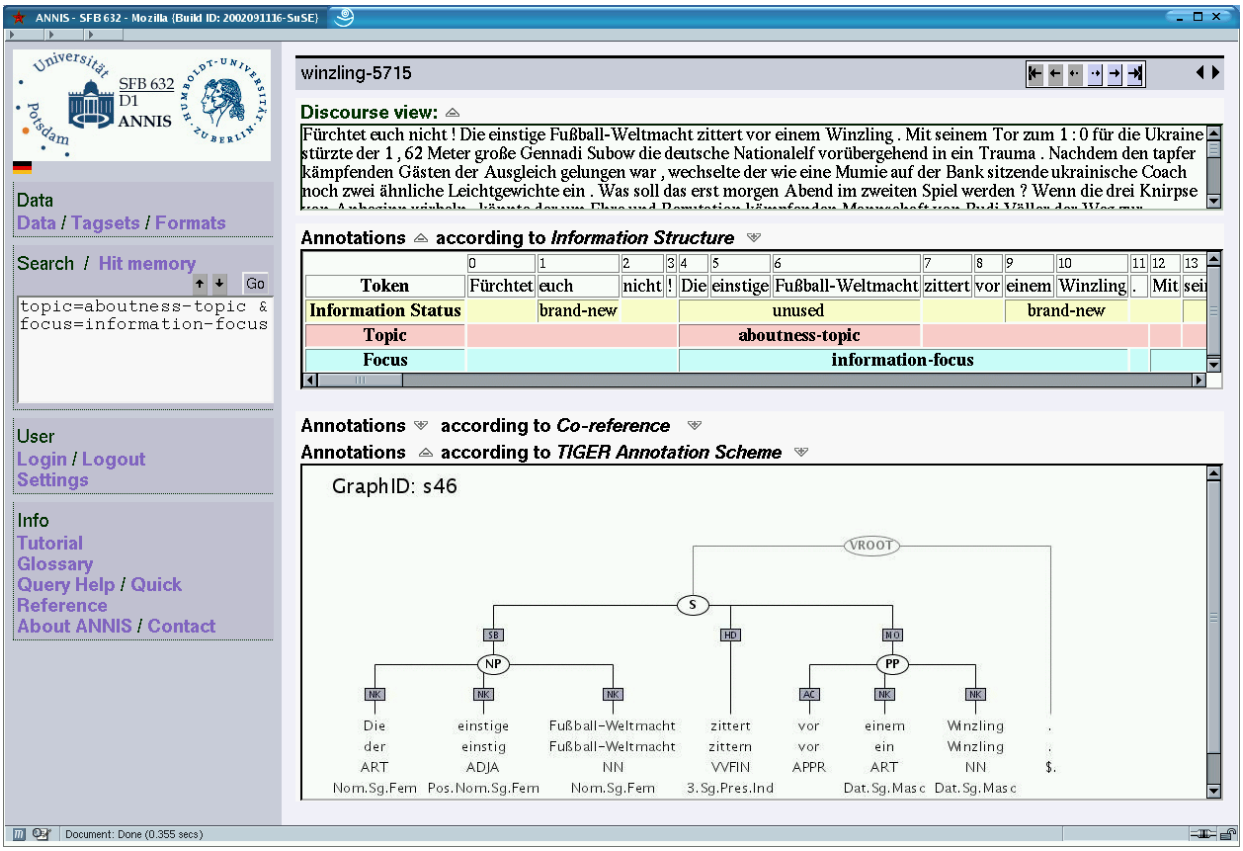

Figure 4: The ANNIS user interface: user-adaptive visualization

in three different modes. At the top, the *discourse view* gives an overview on the discourse. Immediately below this representation, the *table view* enables easy access to annotations consisting of simple attribute-value pairs, characterizing various dimensions of information structure. The *tree view* at the bottom shows the syntactical structure of the sentence that is currently active.[10] In addition to these modes, the system provides a *dialogue view* (not shown in the figure) and integrates a search facility and various export functions (for a description of the search and export functions, see (Dipper et al., 2004)).

The annotation views work independently from each other and can be clicked away, if wished. In our example, the annotation view of the annotation level "Co-reference" is suppressed that way. The individual views are synchronized, i.e., clicking onto a certain markable in one of the displays causes the other displays to switch to the respective markable.

Customizing the display The appearance of the table view can be customized via a format file: the user can specify features such as the coloring and ordering of annotation tiers. Irrelevant annotations can be similarly filtered out. For instance, a user who is interested in referring expressions might want to focus on "discourse-new" NPs. In this case, "discourse-old" annotations would not be displayed. Moreover, the emergence of annotations displayed on one or more tiers, in the form of bubbles, etc., can be determined by the user. This is illustrated by fig. 5, which specifies the appearance of one tier, labeled "Information Status". This tier displays information that is specified in the file *inf_status.xml*: The tier segments correspond to the markables that the *feat* elements in *inf_status.xml* are anchored to. They are labeled according to the attribute *value*, e.g., by "unused". In addition, on mouse over, a bubble emerges, displaying the description of the respective value, as specified in *type_istat.xml* ("The referent is known to the hearer").

```
<tier label="Information Status"
      xml:base="inf_status.xml">
  <feat value="id(feat/@value)/@name"
                          display="label"/>
  <feat value="id(feat/@value)/@descr"
                          display="bubble"/>
</tier>
```

Figure 5: Specifying the display of a tier

4. Related work

Our generic XML format is inspired by the Generic Mapping Tool proposed by (Ide and Romary, 2001) in defining abstract XML elements and attributes that can represent different annotation types. Our format, however, is more restrictive in that it strictly separates purely structural from content information. In GMT, for instance, categorial information like "NP" may be specified as attributes of `<struct>` elements, whereas in our format, this information must be encoded stand-off by `<feat>` elements. Similarly, our format does not allow

[10] The table view has been created by a modified version of EXMARaLDA stylesheets, whereas the tree view has been generated by means of TIGERSearch, which provides an SVG export of data represented in the TIGER XML format.

for recursively-embedded elements to encode hierarchical structures. Instead, it makes use of xlink attributes to define trees and graphs.

Another project closely related to ours is NITE (Carletta et al., 2003; Carletta et al., 2002). Common features include support of multi-level annotation, flexible visualization by stylesheets, and use of XML/XSLT and Java. However, there are also important differences between our projects:

Flexible input NITE allows for various kinds of input formats, including inline and stand-off representations. The user has to modify the stylesheets that trigger the display according to (i) the format of the input data, and (ii) the targeted display format.

In contrast, we require the input to conform to our generic XML format. We are thus able to provide general and multi-purpose stylesheets that can cope with any kind of data (represented in our format). Of course, the burden now lies with the mapping to this format, and people not skilled in XML/XSLT will need support.[11] However, we think that a facility for flexible, customizable visualization must rely on a fixed input: otherwise, nitemodifications of visualization do require subsequent adjustment for many different input formats.

Validation In NITE, the user must list the elements and attributes used in the annotation and must specify their data types. In return, NITE validates the input data with respect to the specification. In our approach, validation must be performed prior to import into the system. This reflects the fact that our internal standardized format does not serve annotation but visualization only and, hence, need not be sensitive to specific DTDs or XML schemas.

Annotation In NITE, data can be edited and annotations can be modified. Our database (in its current version) specializes on data visualization (and retrieval) and does not allow for data editing. That is, to modify data, the user has to export the data, perform the modifications by external tools (e.g., EXMARaLDA), and reimport the modified data into the database.

5. Conclusion and Future Work

In our user scenario, criteria such as data heterogeneity and data accessibility are of particular importance—an aspect which is not captured by most of the currently available databases.

We use a generic XML format as an interlingua: it serves both as the target format for import scripts, and as the source format for rendering different annotation displays in the user interface. The fact that we represent complex multi-level annotations in a uniform way distinguishes our approach from related initiatives (such as TASX or NITE).

Future work includes developing a suitable visualization of inter-sentential annotation like co-reference. We also want to simplify customization of the display by offering an input mask for specifying the display features (similar to the format-table dialogue in EXMARaLDA) and to extend it to other modes of visualization. The input will then be processed by parametrized XSLT-stylesheets, which facilitates reuse in other applications—in contrast to the proprietary Java-based approach we are using currently.

[11] We provide importers for data in the following formats: EXMARaLDA, TIGER XML, MMAX, RSTTool.

6. References

Baumann, Stefan, Caren Brinckmann, Silvia Hansen-Schirra, Geert-Jan Kruijff, Ivana Kruijff-Korbayova, Stella Neumann, Erich Steiner, Elke Teich, and Hans Uszkoreit, 2004. The MULI project: Annotation and analysis of information structure in German and English. In *Proceedings of LREC 2004*. Lisbon, Portugal.

Carletta, Jean, Jonathan Kilgour, Tim O'Donnell, Stefan Evert, and Holger Voormann, 2003. The NITE object model library for handling structured linguistic annotation on multimodal data sets. In *Proceedings of the EACL Workshop on Language Technology and the Semantic Web (NLPXML-2003)*.

Carletta, Jean, David McKelvie, Amy Isard, Andreas Mengel, Marion Klein, and Morten Baun Møller, 2002. A generic approach to software support for linguistic annotation using XML. In G. Sampson and D. McCarthy (eds.), *Corpus Linguistics: Readings in a Widening Discipline*. London and NY: Continuum International.

Carlson, Lynn, Daniel Marcu, and Mary Ellen Okurowski, 2003. Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. In Jan van Kuppevelt and Ronnie Smith (eds.), *Current and New Directions in Discourse and Dialogue*. Kluwer Academic Publishers.

Dipper, Stefanie, Michael Götze, Manfred Stede, and Tillmann Wegst, 2004. ANNIS: A linguistic database for exploring information structure. In Shinichiro Ishihara, Michaela Schmitz, and Anne Schwarz (eds.), *Interdisciplinary Studies on Information Structure (ISIS)*, volume 1. Potsdam, Germany: Universitätsverlag Potsdam, pages 245–279.

Dybkjær, Laila, Niels Ole Bernsen, Hans Dybkjær, David McKelvie, and Andreas Mengel, 1998. The MATE markup framework. MATE Deliverable D1.2.

Erk, Katrin, Andrea Kowalski, Sebastian Pado, and Manfred Pinkal, 2003. Towards a resource for lexical semantics: A large German corpus with extensive semantic annotation. In *Proceedings of ACL 2003*. Sapporo, Japan.

Ide, Nancy and Laurent Romary, 2001. Standards for language resources. In *Proceedings of the IRCS Workshop on Linguistic Databases*. Philapdelphia.

Johnson, Christopher R., Miriam R. L. Petruck, Collin F. Baker, Michael Ellsworth, Josef Ruppenhofer, and Charles J. Fillmore, 2003. Framenet: Theory and practice. Version 1.1.

Jurafsky, Daniel, Elizabeth Shriberg, and Debra Biasca, 1997. Switchboard SWBD-DAMSL shallow-discourse-function annotation. Coders manual, draft 13. Technical Report 97-02, University of Colorado.

Kingsbury, Paul and Martha Palmer, 2002. From TreeBank to PropBank. In *Proceedings of LREC 2002*. Las Palmas,Spain.

Miltsakaki, Eleni, Rashmi Prasad, Aravind Joshi, and Bonnie Webber, 2004. The Penn Discourse TreeBank. In *Proceedings of LREC 2004*. Lisbon, Portugal.

Schiller, Anne, Simone Teufel, Christine Stöckert, and Christine Thielen, 1999. Guidelines für das Tagging deutscher Textcorpora mit STTS (kleines und großes Tagset). Technical report, University of Stuttgart and University of Tübingen.

Stede, Manfred, 2004. The Potsdam Commentary Corpus. In *Proceedings of the ACL-04 Workshop on Discourse Annotation*. Barcelona, Spain.