ANNIS3: A New Architecture for Generic Corpus Query and Visualization

THOMAS KRAUSE[1] AND AMIR ZELDES[2]

1 Humboldt-Universität zu Berlin
2 Georgetown University

Abstract
This paper is concerned with the data structures, properties of query languages and visualization facilities required for the generic representation of richly annotated, heterogeneous linguistic corpora. We propose that above and beyond a general graph based data-model, which is becoming increasingly popular in many complex annotation formats, a well-defined concept of multiple, potentially conflicting segmentation layers must be introduced to deal with different sources and applications of corpus data flexibly. We also propose a generic solution for specialized corpus visualizations in a Web interface using annotation-triggered style sheets, which leverage the power of modern browsers and CSS for multiple and highly customizable views of primary data. We offer an implementation and evaluation of our architecture in ANNIS3, an open source browser-based architecture for corpus search and visualization. We present three case studies to test the coverage of the system, encompassing core linguistic and digital humanities use-cases including richly annotated newspaper treebanks, multilingual diplomatic and normalized manuscript materials edited in TEI, and analysis of multimodal recordings of spoken language.

1 Introduction

The issue of flexible, generic software and standards for linguistic corpora has received increasing attention in recent years (see Ide & Suderman 2007, Bański & Przepiórkowski 2009, Zipser & Romary 2010 to name a few). The general trend in both corpus/computational linguistics circles and in the digital humanities has been to adopt and reuse TEI subsets (e.g. EpiDoc in the antiquities, cf. Cayless et al. 2009) and other XML formats that can be interpreted as annotation graphs, with the canonization of multiple annotation frameworks in this vein under ISO/TC37/SC4 marking the central position of these approaches.[1]

However, while persistent XML representations and nomenclature have advanced substantially in their coverage power and adaptability to new uses, corpus search systems have lagged somewhat behind. Generally speaking, it is still the case that most projects develop search and visualization software dedicated to a single or narrow number of use cases: there is software for searching and annotating treebanks (TigerSearch, Lezius

2002; PML-TQ, Štěpánek & Pajas 2010; Fangorn, Ghodke & Bird 2012), spoken language corpora (ELAN, Brugman & Russel 2004; EXMARaLDA, Schmidt & Wörner 2009), specific interfaces for historical corpora (e.g. the Corpus or Middle English Prose and Verse, http://quod.lib.umich.edu/c/cme/, or the TXM based corpus of the Queste del saint Graal, http://portal.textometrie.org/txm/) and so on. There has been less work on generic corpus query systems that cover a broad spectrum of applications in the sense of GrAF, TEI at large or other XML formats and data-models. This leads some users, especially of complex TEI encoded corpora, to employ generic XML processing tools using XPath/XQuery, which are however far from optimal for handling linguistic data, in terms of presentation, performance and learnability for non-programmers. Additionally, as we will see, some important features of linguistic data are lost in non-linguistic generic XML representations, which give no special status to elements such as word-forms, parts of words, sentences, alignment, time indices and more (see Section 2).

Some five years ago, in an earlier version of the system to be presented in this paper, ANNIS2 took a first step in the direction of a more or less arbitrary graph based corpus search system not intended for any one type of corpus. The ANNIS Object Model (AOM) used by the system at the time was little different from a general graph of annotated nodes and edges. It was able to represent a wide variety of annotations starting with an underlying token layer (the smallest unit of analysis, generally corresponding to word-forms) and any number of non-terminal structures above those tokens, which were visualized using dedicated modules as syntax trees, coreference annotation etc. Yet as we shall see, an arbitrary graph in which all nodes and edges are alike turns out to be insufficient for some important linguistic applications, including parallel corpora, dialog data, manuscripts with annotations below the word level, and more (see Section 4). At the same time, the strategy of creating dedicated visualizations for every application (constituent syntax trees, dependencies, rhetorical structure, information structure and many more) eventually becomes costly and difficult to maintain, while also limiting the ability to create visualizations to programming experts.

For these reasons, the development of ANNIS3, the successor to ANNIS2, which will be presented in this paper, is based on a more sophisticated data-model, which allows corpus designers and users to define more precise semantics for separate regions of generic annotation graphs. The specific features of these graphs offer the possibility of constructing novel visualizations triggered by user-defined annotations using little more than good knowledge of HTML/CSS, substantially expanding the developer and user communities across the multifaceted fields of linguistics and the digital humanities.

The remainder of this article is structured as follows. Section 2 concentrates on the prerequisites of generic corpus query capabilities, addressing what an abstract data-model not bound to a specific XML format must cover. We introduce and present an implementation of the concept of corpus segmentations, which offer a principled solution to issues of multiple/conflicting tokenizations, annotations of units below the word-form (sometimes called ‘subtokens’), and missing and overlapping data. Section 3 deals with the prerequisites of tailored vs. generic corpus visualizations. We cast the task of generic corpus visualization as the formulation of annotation-triggered style sheets, discuss the limits of the approach and some applications that mandate tailored visualizations. Section 4 applies the concepts from the previous sections to three very different case studies: richly annotated multilayer newspaper treebanks, manuscript-near parallel historical corpora, and multimodal corpora of spoken language. We will aim to show how the same data-model can be applied to all of these scenarios and more within one system. Section 5 draws the conclusion with some suggestions for further work.

2 Generic Corpus Query

In this section we discuss the relationship between querying specific corpora and the underlying data-models involved. We begin by outlining the general graph based notion of corpus representation, briefly review a meta-model based approach to dealing with a variety of concrete formats and introduce the concept of segmentations, which designate particular, meaningful reference components of a generic annotation graph.

2.1 Ontological Requirements

In order for a corpus query system to work in a truly generic fashion that is applicable to a wide variety of unseen corpora, it must abstract away knowledge of specific categories and designations. While standards such as the TEI define a complex vocabulary of annotation categories and interpretations, many widespread linguistic search tools (e.g. the IMS Corpus Work Bench [CWB], Christ 1994, and most tools mentioned in the previous section) are agnostic with respect to the names and values of annotations. At the same time it is clear that corpora have different components, and these are found again and again in very different projects: the possibly (meta-)annotated corpus structure, recursive subcorpora, primary linguistic data at the bottom of the corpus hierarchy, non-primary interpretative elements or recursive groupings of parts of the data (essentially a graph of nodes connected to the primary data) and some annotations for those nodes.[2] A system processing such data must therefore have awareness of:

i. Primary data
ii. A graph with any number of nodes and edges above the primary data
iii. A corpus structure in which documents of primary data are organized
iv. Annotations of all three of the above

To see why these aspects of corpus data are distinct consider the following: primary data (e.g. running text) has a special role in many corpus architectures. It determines precedence (which elements precede which) for all annotation layers. It is used to determine the extent of context displayed within search results (e.g. ±5 words). It also has a primary role in almost any type of visualization: it is conceivable to visualize syntax trees and coreference annotations separately, but both visualizations are certain to show the running text to which the annotations apply. We therefore argue that *primary data has a privileged role* in a corpus architecture and must be distinguished in a graph based corpus model (we will see that this is not always so simple in Section 2.3).

At the same time, corpus structure is distinct from the annotation graph attached to textual data. For example, document division can break the relevance of precedence information. If our corpus contains three subcorpora with texts from different newspapers, it makes little sense to say the last word from subcorpus 1 is *followed* by the first word of subcorpus 2. The order of the subcorpora may be arbitrary. If we search for a sequence of two words, it seems that one would want them to be within the same corpus document. We therefore formulate the following definitions for a special corpus graph that begins above the annotation graph discussed above:

- A corpus is a possibly hierarchically recursive container for any number of subcorpora
- The lowest level of the corpus graph shall be called a *document*
- At the document level of the corpus, the inter-document corpus graph is attached to the intra-document annotation graph
- The search space for a corpus query language cannot exceed the document

The last stipulation means that we cannot search for the last word of one document followed by the first one from the next document. Note that this does not mean that one should be unable to find such consecutive words from two different, ordered texts (e.g. chapters in a book). In such cases, the definition simply mandates that both texts are seen as part of the same technical unit called ‘document’, and the distinction between the texts will be modeled within the document internal annotation graph.

2.2 Data-models and the Meta-Model Approach

When writing a tool for a certain linguistic search task, data-modeling is an essential part. The data-model we use constrains the supported types of annotations, performance characteristics of the search, and the range of possible structures that can be queried. For example, CWB models a corpus as a series of tokens (strings) and positional attributes (e.g. parts-of-speech or lemmas), as well as non-recursive structural attributes for ranges of tokens (e.g. sentence, paragraph) and alignment information for parallel corpora. It also uses bigram tables to make searching faster, but these do not add information to the corpus (see Christ 1994). Such a model can represent a wide range of corpora and benefits from very efficient indexing for large datasets. However it is not designed to deal with hierarchical data like syntax trees, and tools building upon this model cannot handle corpora containing this kind of information.

Modeling corpus data means abstracting from specific corpora, tools and formats. A model describes the concepts and possible structure of supported datasets. Whereas creating a particular corpus means choosing a specific format, designing a generic search-tool ideally means supporting multiple formats, with possibly heterogeneous elements. The crucial factor in supporting multiple formats is the expressivity of the underlying data-model used to represent them. A specific format can be seen as one possible *serialization* of an abstract corpus model (i.e. it may be possible to encode the same corpus data in TEI XML or in GrAF). Not every model can cover all kinds of possible annotation structures, but an immediate advantage of any clearly defined model is in delineating the structural and theoretical boundaries we impose on the data. Practically speaking, it is also easier to compare models of annotation rather than specific formats. Representing data from a more complex format in a 'weaker' model leads to data loss.

In the interest of a maximally generic data model, able to fit different kinds of corpora, we have opted for a graph based data-model for ANNIS. Since the system is meant to support corpora in a variety of source formats, a natural choice was to reuse the data-model Salt, from the freely available converter framework SaltNPepper (Zipser & Romary 2010). Salt is already capable of representing a wide variety of popular linguistic formats, including CWB, EXMARaLDA, ELAN, TigerXML, GrAF, Penn Treebank brackets (cf. Marcus et al. 1993), PAULA XML (Dipper 2005), MMAX2 (Müller & Strube 2006) and more. Data from these formats can then be manipulated in memory and converted into one of the other supported formats. Salt is formally defined using the Eclipse Modeling Framework (EMF, Steinberg et al. 2009), a software architecture used to model data and processes. Though it is based on a general graph model, Salt defines corpus linguistic concepts and semantics, like "token", "annotation", "metadata, "relation between annotated nodes" etc. Figure 1 shows the relationship between Salt and a general "labeled Graph" model (see Zipser & Romary 2010 for more details).

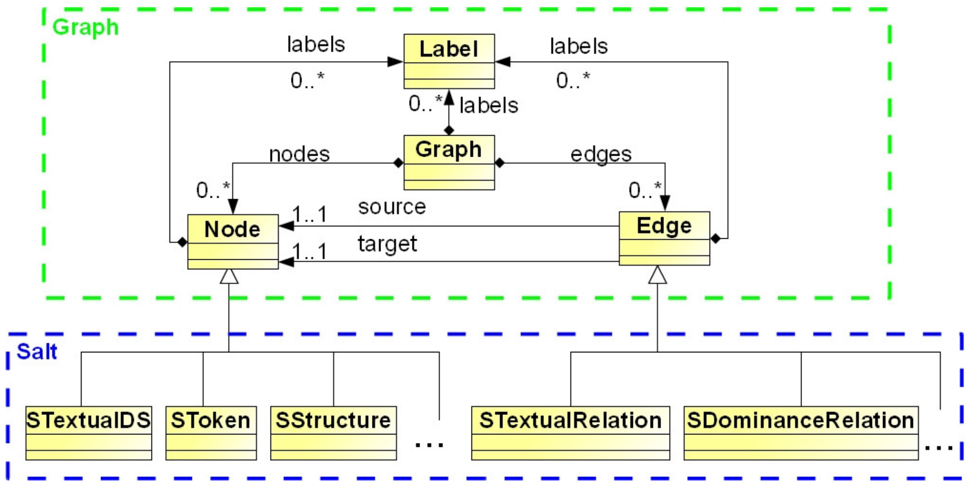


Fig. 1 Excerpt from the graphical representation of the Graph and Salt models

Salt thus defines specializations of general graph nodes and edges with meaningful names that map across formats. For example, nodes of the type "STextualDS" represent textual data sources. The textual content itself is contained in the label "sText". Nodes of the type “SToken” represent tokens, minimal structural units connected to the textual data source by “STextualRelation” edges, which contain labels defining the left and right character borders which tokens cover in a text source. To illustrate the usage of these elements, we can consider a running string of text such as “Mary had a little lamb”. Taken without any information about segmentation, this text would be codified using a node of the type STextualDS. The presence of individual words would be represented by assigning each word to an SToken node with the character positions in the string at which that word begins and ends (the first four characters for ‘Mary’). These may in turn be annotated or joined together to form more complex nodes, annotated as phrases or sentences.

It is important to stress that Salt provides a general model that is not limited to specific theories or formats. As an abstract in-memory data-model it does not need to take technical considerations into account like specific formats (e.g. limitations imposed by XML technology). Its concepts are however inspired by existing formats, including relations like ‘(syntactic) dominance’, ‘parallel alignment’ etc. The corresponding concepts are mirrored by EMF-generated Java classes that contain the resulting corpus structure. The model is implemented in ANNIS3 at three different points:

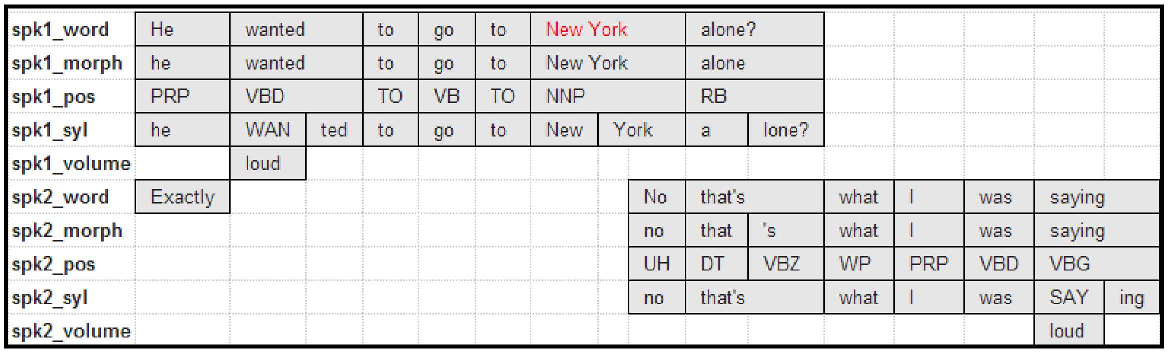
1. Model object storage in a database (implemented in the relational DB PostgreSQL)
2. An ANNIS readable serialization (in text files) for archival and import into the DB
3. Search results representation (articulated using Salt in Java)

Salt already provides a Java object model, meaning 3 is solved by re-using the EMF-based implementation described by Zipser & Romary (2010). 1 and 2 are more difficult, since the ideas of Salt have to be transferred into a relational database format. We call this format relANNIS (see Rosenfeld 2010). As a serialized, persistent format, relANNIS loses the abstract flexibility of Salt and is optimized and extended with redundant data structures in order to optimize import performance and faster query execution. However it is still possible to map Salt representations of corpora to the relANNIS format and vice versa, and Salt representations are used to represent linguistic graphs at run time.

2.3 Introducing Segmentations

The discussion so far has assumed that primary data is fairly straightforward: textual data is seen as a stream of text, and parts of that text may be annotated. However, what are the units of analysis? To find two ‘adjacent’ words, for example, we must ignore spaces between them, effectively defining word-forms in the text as the basic unit of analysis, often referred to as tokens (see Schmid 2008:527-540 on tokenization). However in some situations annotations need to apply to segments that are smaller than, or do not coincide with, the borders of the standard unit of reference, i.e. a word. Specifically, the following scenarios need to be dealt with:

- **Subtokenization:** Parts of words should be annotated or there are annotations that encompass only parts of words. For example in historical corpora, annotation of a single letter as an illuminated initial capital, or numbered line annotations where the beginning of a word is on one line and the end is on another, mean that an annotation border must be drawn within the word-form. In spoken language data, some phonetic or acoustic properties of parts of words might be annotated in a similar way (cf. the annotation of loud syllables in Figure 2).
- **Multiple tokenizations:** There is more than one sensible division of the text into basic units that needs to be taken into account. This can happen because data from two annotators or tools differs (e.g. a tagger tags "New York" together, but a parser treats the input as two syntactic tokens, and similarly for contracted word-forms like "that's" in Figure 2), or because there are two levels of analysis (e.g. diplomatic and normalized transcriptions of an ancient manuscript may have different word divisions based on differing orthography).
- **Overlapping dialogue data:** two or more speakers produce utterances simultaneously, but words from all speakers should be treated equally as basic units in the text of the conversation. In conversational data, speakers beginning words in the middle of other speakers' words is commonplace, and again requires divisions within the other speaker's words (cf. Figure 2, where speaker 2 says "No" in the middle of the word "York", at a time point corresponding to no other annotation border).



spk1_word	He	wanted		to	go	to	New York			alone?						
spk1_morph	he	wanted		to	go	to	New York			alone						
spk1_pos	PRP	VBD		TO	VB	TO	NNP			RB						
spk1_syl	he	WAN	ted	to	go	to	New	York		a	lone?					
spk1_volume		loud														
spk2_word	Exactly								No	that's		what	I	was	saying	
spk2_morph									no	that	's	what	I	was	saying	
spk2_pos									UH	DT	VBZ	WP	PRP	VBD	VBG	
spk2_syl									no	that's		what	I	was	SAY	ing
spk2_volume															loud	

Fig. 2 Grid view of conflicting segments in dialog data

The scenarios above, and others like them create several difficulties for a corpus search system. Firstly **searching for primary text** data becomes non-trivial. To find all occurrences of a word like 'become', one does not wish to query whether speaker 1 said 'become', or speaker 2, or 3, and in fact, in a complex corpus one may have no idea how many speakers are in a conversation or how annotation borders have caused words to be split up. It must be possible to search for text on 'all textual levels' in some scenarios, but also for text coming only from one or more specific sources (e.g. words from any speaker or just a specific speaker in a dialogue). We will call such textual layers *segmentation layers*, though there can also be non-textual segmentation layers (see below).

Secondly, the aforementioned **context problem** becomes egregious if we do not know which subdivided units are word-forms. A context of plus/minus *n* tokens (e.g. 5 tokens to the left and right for a key-word-in-context or ‘KWIC’ view) becomes unreliable if these may correspond to 5 words or only 1 word which happens to be divided into 5 subunits because of word internal annotation borders. In other words, if the smallest units are potentially less than a word, context will behave erratically, unpredictably showing parts of words, and more or less context depending on whether the search result happens to include ‘subtokenized’ (subdivided) positions. It is also conceivable that units larger than word-forms could be useful for context size definitions, e.g. sentence boundary annotations, which are not themselves part of the text, and again these can conflict between simultaneous speakers. It is therefore necessary to be able to designate any annotation layer as a unit for segmentation: word-forms, syllables, paragraphs or whatever else makes sense for a specific corpus.

Thirdly, the definition of **adjacency** in a corpus with one of the scenarios above becomes problematic without explicit segmentation layers. If there are multiple layers of ‘basic text’ (possibly unannotated, but tokenized raw data), it is difficult to decide when two elements are adjacent. Two consecutive elements on a tokenization layer may be interrupted by elements on another tokenization layer. It is therefore necessary to have separate search facilities for ‘absolute’ adjacency (nothing in between) and ‘relative’ adjacency, in terms of some particular segmentation layer. For example, we might want to annotate one speaker coughing between two words of another speaker. Whether those two words are adjacent depends on whether we ask “what is the next thing that speaker 1 said?” or “did anything else whatsoever happen between those words?”. Segmentation layers allow us to make this distinction explicitly wherever it is deemed sensible.

Finally, it has already been mentioned that **correct visualization** of key-words-in-context depends on the recognition of words or a similar unit of reference. This actually applies to many visualizations which assume one unambiguous token layer as part of their display (e.g. syntax trees above word-forms, coreference edges, and more). Facilities are required to define which layer should be treated as the base segmentation for a visualization, as well as allowing to switch between available segmentations (view a syntax tree for each speaker ignoring the other one, view diplomatic manuscript ‘words’ in context or normalized ‘words’ in modern orthography, etc.).

The examples in this section have concentrated on historical and dialogue data, but it should be made clear that the intention is to create a conceptually sound model for generic corpus representation, the applications of which may be very broad: for example language learner corpora have very similar problems, since one may be interested in viewing original learner language or rather some form of correction of the data (a grammatical, stylistic, or other ‘target hypothesis’, see Reznicek et al. 2010 and Section 3.3) as a point of reference. The same can be said for viewing a foreign language in its original script or a transliteration which may or may not use the same dividing units,[3] and so on. With these very broad applications of corpus modeling in mind, we now turn to consider the form of a query language for generic corpus data with support for the concept of segmentations.

2.4 Implementing Generic Search: AQL

To achieve the same generality of our data-model, the corresponding query language must be expressive enough to fit unforeseen linguistic phenomena in a wide range of corpus encodings. Still, the language must be simple enough to learn for non-experts. The design of the ANNIS Query Language (AQL) is therefore based on existing query engines like NQL (Evert & Voormann 2003) and TigerSearch, which are particularly oriented towards tree and graph data. It is also somewhat similar to CQP, the query language of the IMS CWB mentioned above, and some closely related variants such as CQL (used by SketchEngine, Kilgarriff et al. 2014) and Poliqarp (Janus & Przepiórkowski 2007) in its use of a simple sequence of attribute value pairs.[4]

The basic design approach for AQL is to introduce as few language concepts as possible, while making the single concepts configurable and combinable. New kinds of linguistic queries are created by recombining existing concepts instead of adding new ones. An AQL query consists of the descriptions of the query nodes being searched for and constraints applying between these nodes. The simplest search:

```
(1)    Node
```

defines a single query node as a result, which has no constraints at all. Search nodes can also be any annotation name (with or without a value), e.g.:

```
(2)    cat="S"
```

returns all subgraphs containing an annotation with the name "cat" and the value "S". It is also possible to search for annotations within a certain namespace (using ":" as a separator, e.g. "tiger:cat") or search for values by regular expressions, using slashes (/…/) instead of double quotes. Multiple constraints are separated by the "&" character. If multiple query node constraints are defined they have to be connected by some relational constraints. Thus the following is invalid AQL:

```
(3)    lemma="sleep" & pos="VB"
```

since there is no relation between the "lemma" annotation and "pos" (do they apply to the same word? Does one follow or dominate the other?). Relations between query nodes are defined by operators, e.g.:

```
(4)    lemma="sleep" & pos=/NP.*/ & #1 ->dep #2
```

uses the "->" arrow operator, which signifies pointing relations, to connect the first query node (#1, the lemma 'sleep') with the second (#2, a proper noun tag beginning with 'NP'). The operator only selects nodes that are connected by an edge of the type given after the arrow, here "dep" (for syntactic dependency between 'sleep' and a proper noun). There is no predefined set of allowed names for edge types. An extension of this operator is the additional constraint on edge annotations which are defined in square brackets, e.g. for a subject function annotation:

```
(5)    lemma="sleep" & pos=/NP.*/ & #1 ->dep[func="subj"] #2
```

Again, the list of annotation names and values is open. It is also possible to query for annotations spans containing, overlapping or dominating other annotations. For example:

```
(6)    lemma="sleep" & speaker="mother" & #2 _i_ #1
```

searches for cases where the lemma 'sleep' is included (_i_) within the same area covered by a speaker annotation with the value 'mother'.

AQL has a wide range of operators for querying the different types of relations that can be found in the Salt data-model and many can be parameterized. Most interesting in the context of the previous section is the precedence dot-operator, '.':

```
(7)    "the" & "house" & #1 . #2
```

The query above searches for two textual units (tokens or other text segmentations) which have values "the" and "house". #1 directly precedes #2 on the minimal token level. For all binary operators like `_i_` and `.` above, it is also possible to position the operator directly between search terms, so that an alternative short-hand for (7) would simpy be: `"the" . "house"`. However when multiple constraints apply to the same node, the longer notation referencing nodes by number is required.[5] The precedence operator can be parameterized by defining the distance of the nodes, e.g.:

```
(8)    "the" & "house" & #1 .3,10 #2
```

finds all occurrences of "the" followed by "house" within 3 to 10 tokens. The AQL precedence operator can be bound to any segmentation level. Thus the next query searches for the two adjacent words on the "speaker1" level, even if they are interrupted by another unrelated annotation (e.g. in spoken data coughing, background noises, or in written data a diagram, page number at a page break, etc.).

```
(9)   "the" & "house" & #1 .speaker1 #2
```

For more AQL examples, see Section 4. For a complete list of AQL operators, see http://www.sfb632.uni-potsdam.de/annis/aql.html.

3 Generic Corpus Visualizations

In this section we approach the generation of flexible visualizations directly conditioned on the content of corpus annotations, as opposed to specialized views that require deep knowledge of the data. We discuss which cases can be handled by a generic approach, how it can be implemented using contemporary browser-based techniques and offer some potential problems and solutions for the interaction of segmentations with visualizations.

3.1 Dedicated versus Generic Visualization

Traditionally, corpus interfaces have focused on key-word-in-context views, which display search results in a dedicated concordance with left and right context (see for example interfaces for CWB such as CQPweb, Hardie 2012). However more generally speaking, an ideal generic visualization component would simply be a configurable one-to-one translation from annotation structures to visual elements. Some sort of styling instruction would translate the presence of particular annotations into corresponding styles, and various layouting options should be available. An obvious candidate language for representing annotation structures in a Web interface is some form of HTML/CSS, which is very flexible and supported across operating systems without installing additional software.

In some cases, a mapping from the data structure to the desired visual structure can be more or less straightforward, but in others complex program logic must be used that exceeds a simple CSS style sheet. We will therefore begin by discussing complex cases that defy generic approaches, determine what might still be made generic within them, and then move on to a large class of cases that do allow generic solutions. In essence, generic solutions for textual data are most readily available when annotations are translatable into graphical mark-up of areas of a running text. This is also the type of annotation that immediately lends itself to the sort of annotation grid view seen in Figure 2, which was adapted for ANNIS based on the EXMARaLDA annotation software. Because of its neutrality and general applicability, it is the default visualization of choice for annotations in ANNIS.

A large class of visualizations that tend to require dedicated solutions involves cases where a complex graph must be visualized using a layouting algorithm, which determines the optimal positions, sizes, distances etc. of the visualized nodes. This applies to most types of syntax trees (dependency and constituency trees), which bring special constraints with them. For example in Figure 3, the dependency syntax tree in Panel A and the constituent tree in Panel B both retain the linear order of some German word-forms from the Potsdam Commentary Corpus (PCC, Stede 2004). The tree in Panel A needs to calculate the required height of the arch-like edges based on the distance between connected words, the length of the words, and the presence of other edges covering the same space. The tree in Panel B must determine the width and height of branch brackets, the optimal position for labels, etc. Panel C, by contrast, shows a non-linear dependency tree of New Testament Classical Greek (from PROIEL, Haug et al. 2009), in which words are arranged in a hierarchy. Here too, optimal positioning and spacing of nodes are non-trivial, and some special symbols (arrow and node types) have been used.

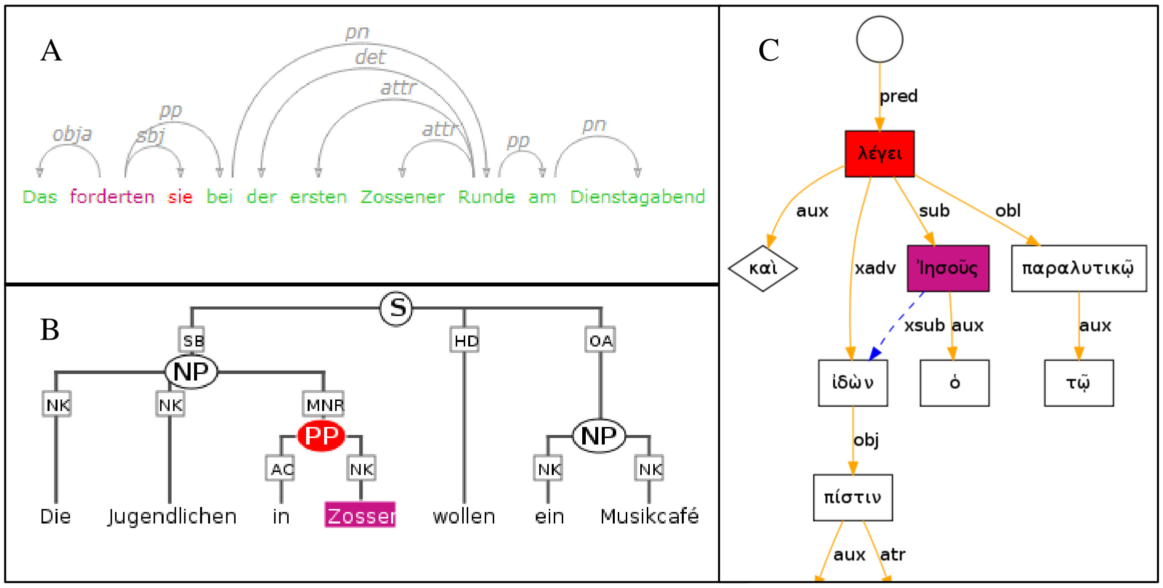


Fig. 3 Several syntax tree modules in ANNIS3

While some configurability is still possible within such visualizations (e.g. colors, arrow and node types, names of the annotations producing node and edge labels), it is not realistic to expect a generic styling mechanism to realize such results, as complex layout calculation exceeds the capacity of most style sheet languages (for some more examples of dedicated visualization for uses other than syntax, see Section 4.1).

In ANNIS3 we have therefore taken the approach of enabling some modularity of dedicated solutions by allowing the inclusion of visualization plugins, which are fed the same data-model (Salt), while the names of specific annotations corresponding to nodes or edges in a tree etc. can be configured separately. Plugins can also make use of existing standards and visualization technologies: for example, some of the tree visualizations, including the one in Panel C above, are realized in ANNIS using multipurpose layouting software provided by Graphviz (http://www.graphviz.org/), a framework offering implementations of some useful algorithms not specifically targeted at linguistic applications. With the problems of graph layouting in mind we now turn to cases that can be addressed using simple HTML and style sheets, which will turn out to include a wide variety of complex scenarios.

3.2 Strategies for Generic Visualization

Visualizing annotation graphs from a model like Salt in a flexible way is similar to styling an HTML page: there is a data-model which contains certain elements, to every occurrence of which some conditional formatting can be applied. We therefore reduce the definition of a generic visualization to two components: a configuration file, which determines which annotations generate which structural elements (HTML), and a style sheet which determines how browsers render those elements (CSS). The configuration file is a simple text file with three columns: a generation-instruction, an element to be generated, and some content to fill the element with. As an example, consider the following configuration file, which is used to visualize a corpus from a Nigerian film in the Hausa language in the form of a 'movie script' dialogue ('#' precedes comments; cf. Figure 4 for the resulting visualization):

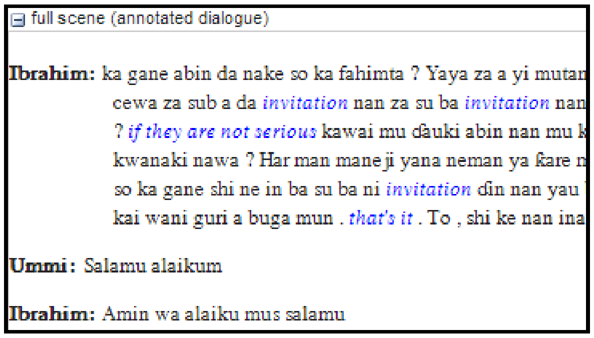
```
tok         span; style="word"              value       #output all tokens
POS="FOC"   span; style="foc"                           #highlight focus particles
info        t:title; style="info"           value       #put info annotation in @title
speaker     div:speaker; style="spk"        value       #create div for speaker change
lang        span:lang; style="code-switch"  value       #highlight code-switching
```

The first instruction tells the visualizer to output the value of each token and surround it by a span with the style 'word'. This gives the basic textual content of what each speaker says. Next, if the part-of-speech annotation (POS) has the value 'FOC', a span styled 'foc' should be generated to highlight focus particles. Note that this will only happen if focus particles are present in the text. The third entry codes 'info' annotations, which give extralinguistic information about the scene, and places them into an element 't' with an attribute 'title', whose value is the value of the annotation. Since 'title' attributes are rendered as tooltips by browsers, hovering over a stretch of dialogue will show any annotations about the events in the scene. An HTML fragment generated by these annotations will look like this (sentence: *Ibrahim ne* 'It's Ibrahim!'):

(10)
```
<t title="Hajiya, in a dream">
 <span style="word">Ibrahim</span>
 <span style="foc"><span style="word">ne</span></span>
</t>
```

The information that this is a 'dream sequence' is given as a tooltip in the 'title' attribute, while each word-form is styled as a word. The word *ne* additionally receives the 'foc' style, since it carries the corresponding part-of-speech annotation, whereas *Ibrahim* does not. This style, as well as all others referred to here, is defined in an accompanying CSS file, which determines how individual tokens and annotations will be rendered.

The final two instructions in the configuration file create a <div> whenever a new 'speaker' annotation begins, giving the name of the speaker in a 'speaker' attribute, and style spans containing code-switching (use of languages other than Hausa) together with an attribute naming the language used, which comes from an annotation called 'lang'. The visualization can be seen in action in Figure 4, where code-switching has been highlighted in blue italics and the speaker <div>'s are given a hanging indent and bold speaker names followed by colons.



Ibrahim: ka gane abin da nake so ka fahimta ? Yaya za a yi mutan
cewa za sub a da *invitation* nan za su ba *invitation* nan
? *if they are not serious* kawai mu ɗauki abin nan mu k
kwanaki nawa ? Har man mane ji yana neman ya ƙare n
so ka gane shi ne in ba su ba ni *invitation* ɗin nan yau
kai wani guri a buga mun . *that's it* . To , shi ke nan ina

Ummi : Salamu alaikum

Ibrahim: Amin wa alaiku mus salamu

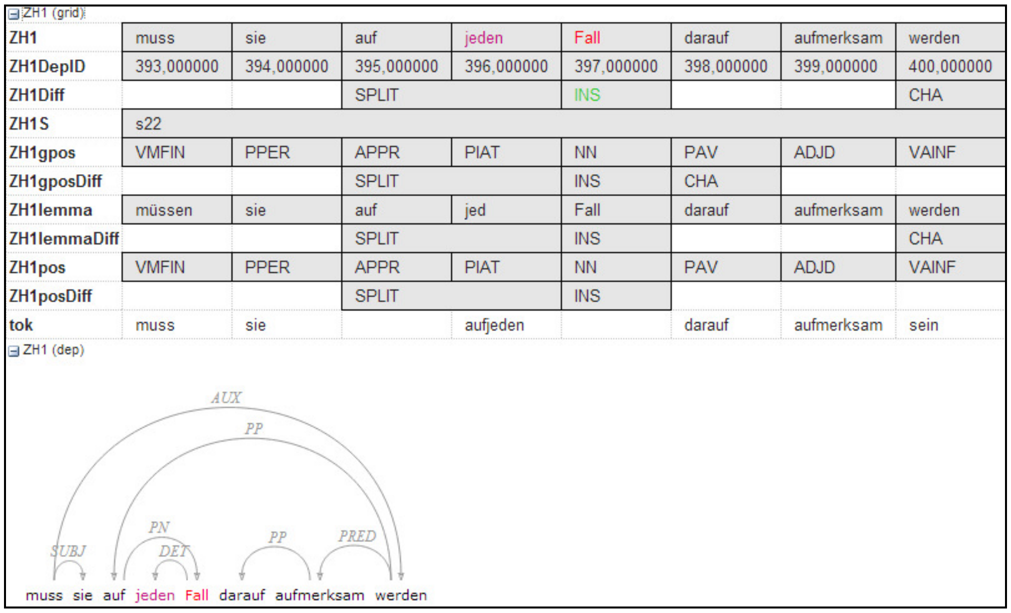
Fig. 4 Generic HTML visualization adapted to a 'movie script' scenario, with English code-switching highlighted within a Hausa dialogue

The range of visualizations that can be created in this way is limited primarily by the corpus designer's imagination (and the limitations of HTML). The entire visualization in Figure 4 is defined in a few lines of code, yet its utility compared to a plain text output without clear demarcation of speaker alternations and highlighting of relevant information is much greater for researchers. For different research questions, multiple views of the same data can be generated relatively easily, without major changes to the system (see Section 4 for more examples).

3.3 Implementing Segmentations for Textual Data Visualization

The visualizations explored above are all more or less anchored in the idea of a sequence of words being styled and laid out in certain ways. But as we have seen, the notion of the word-form brings some limitations on corpus design with it, limitations which we have sought to overcome using a mechanism of multiple, privileged segmentations. Just as segmentations are necessary for correct query behavior, they must also be taken into account in visualization. In order to do so, it is necessary to allow different visualizers to 'listen' to different annotation layers in lieu of tokens, the smallest units of the analysis. This means that, for example, a dependency tree no longer connects tokens with annotation edges, but rather some other arbitrary unit defined by the corpus designer.

This is illustrated in Figure 5, taken from the error-annotated learner corpus of German, Falko (Reznicek et al. 2010). In the corpus, several target hypotheses are formed for potentially ungrammatical learner utterances, reflecting what annotators believe the learner is trying to say. The target hypotheses can then be contrasted with actual learner utterances to study different kinds of errors. While the learner language is often difficult to parse correctly, syntactic structures on the target hypothesis level can be analyzed with greater ease. However, the base text produced by the learner and each alternative hypothesis form different segmentations, meaning that the syntactic analysis must be able to refer to the hypothesis annotation as its basis, rather than the actual primary text. Thus the dependency tree in Figure 5 has the segmentation layer ZH1 (*Zielhypothese1*, 'target hypothesis 1') as its leaves, whereas the actual learner utterance segmentation can be seen on the line labeled 'tok' in the grid. Some of the leaves necessary for the syntactic analysis are not present in the original segmentation: these are annotated on the layer ZH1Diff (differences between ZH1 and the text) for example as insertions (INS) or splits (SPLIT) of word-forms produced by the learner.


ZH1 (grid)

ZH1	muss	sie	auf	jeden	Fall	darauf	aufmerksam	werden
ZH1DepID	393,000000	394,000000	395,000000	396,000000	397,000000	398,000000	399,000000	400,000000
ZH1Diff			SPLIT		INS			CHA
ZH1S	s22							
ZH1gpos	VMFIN	PPER	APPR	PIAT	NN	PAV	ADJD	VAINF
ZH1gposDiff			SPLIT		INS	CHA		
ZH1lemma	müssen	sie	auf	jed	Fall	darauf	aufmerksam	werden
ZH1lemmaDiff			SPLIT		INS			CHA
ZH1pos	VMFIN	PPER	APPR	PIAT	NN	PAV	ADJD	VAINF
ZH1posDiff			SPLIT		INS			
tok	muss	sie		aufjeden		darauf	aufmerksam	sein

ZH1 (dep)

Fig. 5 A dependency tree visualizer refers to an alternative segmentation layer for its 'word-form' leaves in the Falko learner corpus

Thus a truly generic corpus architecture must allow multiple reference levels in the sense of segmentation layers on all levels of the architecture: search distance, context size, and visualization. With the data-model described so far, we can now consider some complex case studies that test the limits of our approach for different types of language data.

4 Case Studies

4.1 Multilayer Treebanking

A classic test case for a search system representing richly annotated corpora is dealing with syntactic annotations, i.e. with so-called treebanks. State-of-the-art treebanks have grown considerably in complexity since the publication of the Penn Treebank (PTB, Marcus et al. 1993), which has been used as a starting point for many later corpora. Contemporary projects include much richer annotations: parts-of-speech, lemmas and morphology, as in the German Tiger Treebank (Brants et al. 2002); named entities and coreference relations (e.g. the PTB-based OntoNotes corpus for English, Hovy et al. 2006), or coreference and topological field annotations in the German TüBa-D/Z (Telljohann et al. 2009), as well as information structure and rhetorical structure annotations in PCC, mentioned above. In this section we therefore test ANNIS on these treebanks and more, seeing how different annotation schemes map onto the generic query and visualization components.

The data-model of corpora annotated for syntactic constituents mandates primarily the possibility of hierarchically nested nodes all carrying the same annotation type: the phrasal category such as NP, PP, a sentence or clause node, etc. In English treebanks such as the PTB, many grammatical functions are not annotated directly, but rather identified using their position in the tree (cf. Figure 6A). In German treebanks, functions are typically given to each node in the tree by means of edge annotations. This is directly represented in ANNIS and AQL using a typed and labeled dominance relation (cf. the labels in Figure 6B-C, showing a query for nominal appositions).

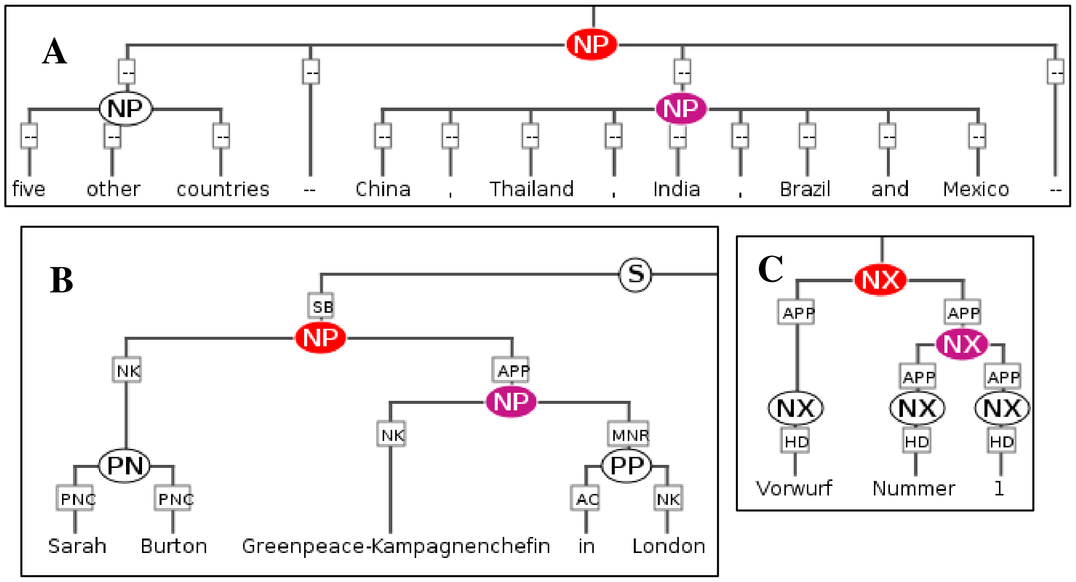


Fig. 6 Similar trees for appositions in OntoNotes (A), Tiger (B, 'Sarah Burton Greenpeace campaign chief in London') and TüBa-D/Z (C, 'accusation number 1').

The figure shows that, although annotation schemes differ across corpora substantially (names of nodes, edge labels if present, preference for deep/binary branching versus flatter constituents), they are all representable as labeled graphs of nodes and edges. The queries producing these results in AQL vary in content depending on the annotation names, but are structurally similar:[6]

(11) (OntoNotes [A]): `cat="NP" & cat="NP" & #1 > #2`
(12) (Tiger [B]): `cat="NP" & cat="NP" & #1 >[func="APP"] #2`
(13) (TüBa-D/Z [C]): `cat="NX" & cat="NX" & #1 >[func="APP"] #2`

The nominal phrase category is called NP in Tiger and OntoNotes, but NX in TüBa-D/Z, and what exactly receives this annotation differs from corpus to corpus as well (e.g. no unary derivations and flat NP/PP structures in Tiger, leading to no NP node above 'London' in Figure 6B). This means that users must be familiar with the corpora they are searching in, and ANNIS facilitates this somewhat by providing example queries and a list of available annotations for each corpus.[7] However from a technical standpoint, the different nomenclature is unproblematic, since unlike some XML formats, such as TEI XML, there are no predefined annotation names. The visualization used in Figure 6 can be configured to render any node or edge annotations.

Note also that in OntoNotes, there is no edge annotation allowing to search for appositions. The AQL query above will find any NP dominating an NP (e.g. possessives like "John's dog"). Appositions can still be searched for using the coreference annotation available in the corpus, which includes a special apposition edge type. The appropriate query is:

(14) `node & node & #1 ->APPOS_relation #2`

The query searches for any two nodes that are connected by a pointing relation of the type APPOS_relation. In this case, the dominance operator is not used, since the first node does not necessarily *consist* of the second node. Whereas dominance implies inherited coverage of the same text (an NP literally consists of its constituents) a coreference relation merely links two nodes as being somehow related, without implying constituency, and indeed many coreference relations apply across sentences, meaning relevant nodes are in separate syntax trees. Figure 7 gives an appropriate dedicated visualization of the reference relationships in ANNIS for the document containing the sentence from Figure 6A.

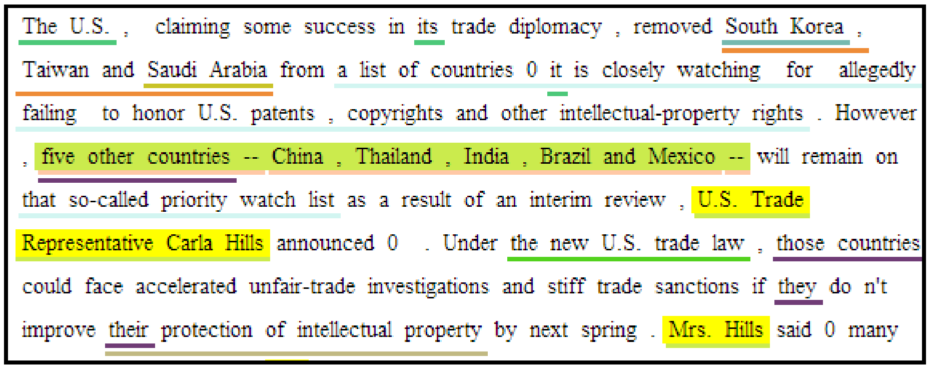
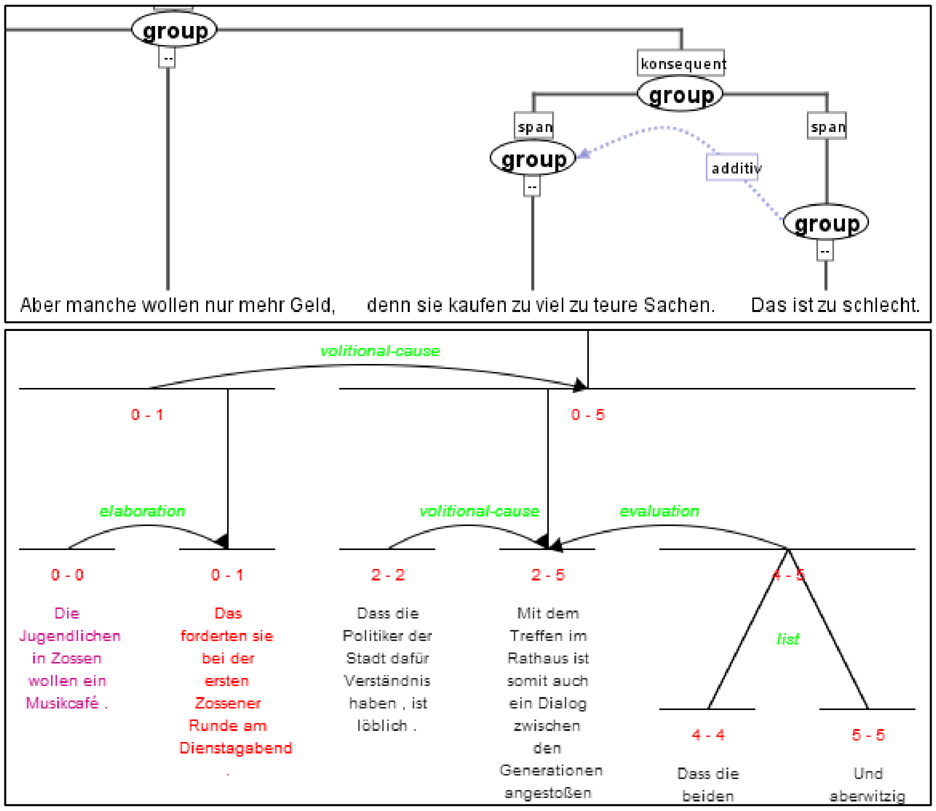

The U.S. , claiming some success in its trade diplomacy , removed South Korea , Taiwan and Saudi Arabia from a list of countries 0 it is closely watching for allegedly failing to honor U.S. patents , copyrights and other intellectual-property rights . However , five other countries -- China , Thailand , India , Brazil and Mexico -- will remain on that so-called priority watch list as a result of an interim review , U.S. Trade Representative Carla Hills announced 0 . Under the new U.S. trade law , those countries could face accelerated unfair-trade investigations and stiff trade sanctions if they do n't improve their protection of intellectual property by next spring . Mrs. Hills said 0 many

Fig. 7 Coreference annotation in OntoNotes

The apposition between the ‘five other countries’ and ‘China…’ is marked with an underline and a same colored background that can be switched on or off. This is especially useful for quick identification of longer distance coreference chains, such as the chain highlighted in yellow, linking occurrences of U.S. Trade Representative Carla Hills across the text.

As already noted in Section 3.1, visualizing relational information such as syntactic dominance or coreference is harder to do in a generic way than simply highlighting annotation spans, since layouting considerations come into play. At the same time, some measure of customizability and reusability can be afforded. The coreference view in Figure 7 has been used for multiple corpora (including TüBa-D/Z and PCC), and the syntax tree has also been extended to several special cases. Specifically, the same tree layouting has proven easy to adapt for right-to-left languages like Arabic and Hebrew (by reversing the order of word nodes at the bottom), and multiple instances of the tree visualizer have been used to represent parallel treebanks such as SMULTRON (Volk et al. 2010). As for other languages, the entire system supports Unicode, and languages such as Chinese or Japanese are therefore supported out of the box, though some visualizations, such as syntax trees, impose word separation on such languages, which otherwise leave no spaces between words.[8]

However, reusability does have its limits, even for data that can be modeled as a tree. For example, Rhetorical Structure Theory trees (RST, Mann & Thompson 1988), which represent the rhetorical relationship between sentences (e.g. ‘elaboration’, ‘evidence’), are representable using the same visualization as above, but the result is not very easy to read if relations in a whole text are to be analyzed (Figure 8, top): very little of the entire text fits into a window when words are arranged in a row. For this reason, it was decided to build a dedicated RST visualization for ANNIS (Figure 8, bottom), which offers a more convenient layout for reading a tree of long sentences, rather than short phrases.


group
konsequent
group
span
span
group
additiv
group
Aber manche wollen nur mehr Geld,
denn sie kaufen zu viel zu teure Sachen.
Das ist zu schlecht.
volitional-cause
0 - 1
0 - 5
elaboration
volitional-cause
evaluation
0 - 0
0 - 1
2 - 2
2 - 5
4 - 5
list
Die Jugendlichen in Zossen wollen ein Musikcafé .
Das forderten sie bei der ersten Zossener Runde am Dienstagabend .
Dass die Politiker der Stadt dafür Verständnis haben , ist löblich .
Mit dem Treffen im Rathaus ist somit auch ein Dialog zwischen den Generationen angestoßen
4 - 4
5 - 5
Dass die beiden
Und aberwitzig
Fig. 8 Visualizing rhetorical structure with a more generic tree visualization (top) versus a highly dedicated one (bottom)

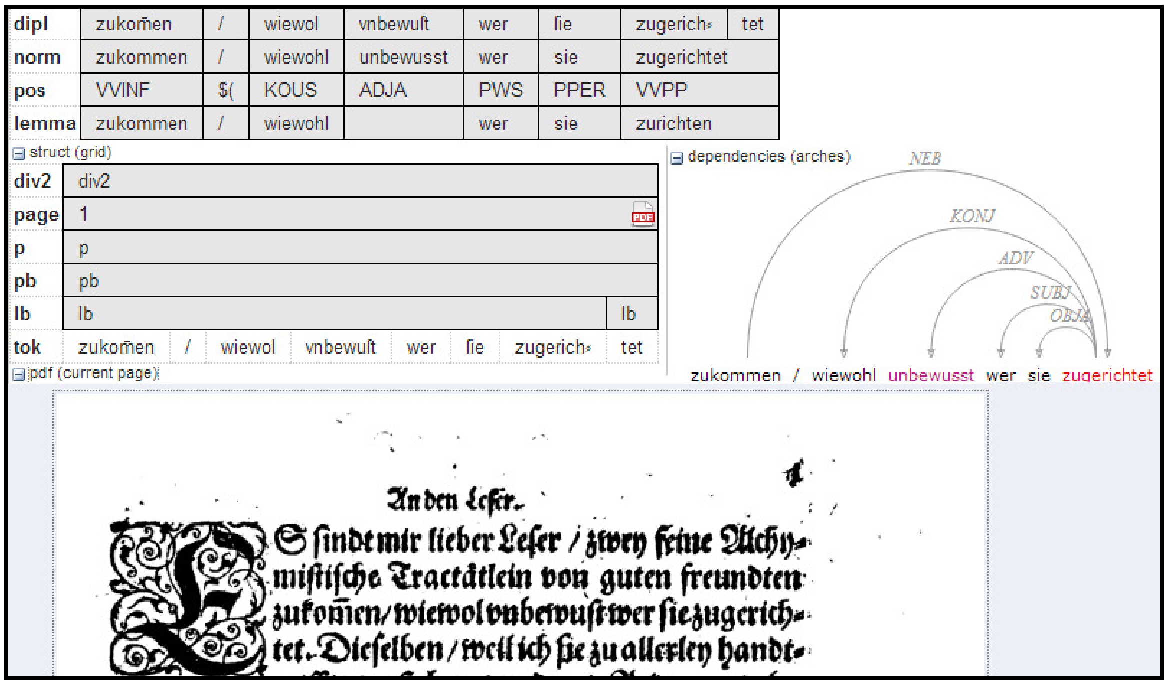
Finally note that no attempt is made to visualize ‘everything at once’. The PCC corpus containing the RST analysis above also has constituent syntax trees, information structure annotation, coreference and entity annotation, and all can be queried in tandem. But for the sake of convenience, different aspects of the annotation graph are picked out by different visualizations, making the forest of information easier to navigate for users.

4.2 Historical Corpora

Historical corpora offer a rather different, but substantial set challenges than modern newspaper texts (cf. Claridge 2008). They typically contain non-standard, inconsistent orthography, and their usage involves a tension between the desire to represent physical objects such as manuscripts and codices accurately (including line breaks, pages, repetitions and errors) versus a modern, text-based view which seeks to capture the semantic content of a document in a consistent way that is easy to search through. Because of this tension, historical corpora offer a natural scenario for the implementation of alternative ‘word’ levels and so-called sub-tokenization, i.e. the definitions of annotation units smaller than the standard reference word.

As a first example, we may consider the representation of textual data in the RIDGES corpus (Register In Diachronic GErman Science), which was constructed to study the development of German as a language of science between the 16th and 19th centuries

(Krause et al. 2012). The diplomatic annotation of the underlying documents was done in TEI XML and contains page and line-breaks that split up word-forms, non-standard orthography, and a variety of non-textual, structural elements such as figures, paragraph marking, marginalia, etc. The corpus was manually normalized to Modern German orthography, then automatically part-of-speech tagged and lemmatized, followed by manual correction and further semantic annotation of scientific concepts such as definitions, technical terms and more.[9] It is in principle unproblematic to represent all of these annotations concurrently on an annotation grid, by defining as many minimal units as necessary and spreading the annotations to overlap correctly, as seen in Figure 9. The orthographic layer 'dipl' is interrupted at the final word *zugerich⸗tet* by a line break, but this doesn't affect the 'norm' or 'lemma' layers. In the structural annotation grid in the middle of the figure, the disruptive 'lb' (line-break) annotation can be seen. The corresponding position in the text can be seen in the PDF view at the bottom, which is aligned to the 'page' annotation and rendered by a standard PDF browser plugin.



dipl	zukom̃en	/	wiewol	vnbewuſt	wer	ſie	zugerich⸗	tet
norm	zukommen	/	wiewohl	unbewusst	wer	sie	zugerichtet	
pos	VVINF	$(	KOUS	ADJA	PWS	PPER	VVPP	
lemma	zukommen	/	wiewohl		wer	sie	zurichten	

struct (grid)

div2	div2							
page	1							
p	p							
pb	pb							
lb	lb							lb
tok	zukom̃en	/	wiewol	vnbewuſt	wer	ſie	zugerich⸗	tet

dependencies (arches)

pdf (current page)

An den Leſer.
ES findet mir lieber Leſer / zwey feine Alchy-
miſtiſche Tractätlein von guten freundten
zukom̄en/ wiewol vnbewuſt wer ſie zugerich-
tet. Dieſelben / weil ich ſie zu allerley handt-

Fig. 9 The RIDGES corpus in ANNIS, with conflicting 'norm' and 'dipl' segmentation layers

The 'norm' layer is defined as a segmentation layer, meaning that searching for `"zugerichtet"` ('prepared') as a normal textual unit is possible, as well as searching within a context of ±*n* 'norm' units. Also note the dependency syntax annotation on the right of the figure: dependency syntax applies between standard word-forms, and therefore the visualization is configured to operate on the 'norm' layer, as discussed in Section 2.3.

Finally note that it is possible for annotations to require a gap in the text. Unary elements like TEI <figure/> cannot be interpreted as encompassing a span of text: they stand *between* portions of text. Therefore the ‘figure’ annotation in Figure 10, corresponding to a ‘Drawing of two cut-out boards’, causes a gap in the ‘norm’ layer.

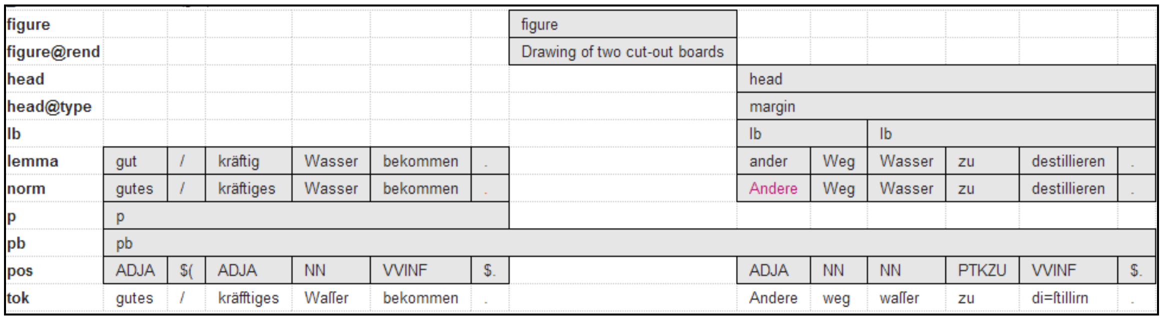


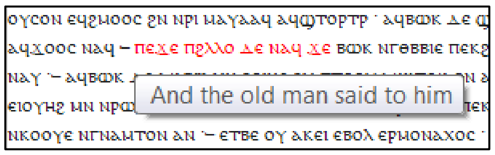
figure							figure						
figure@rend							Drawing of two cut-out boards						
head								head					
head@type								margin					
lb								lb		lb			
lemma	gut	/	kräftig	Wasser	bekommen	.		ander	Weg	Wasser	zu	destillieren	.
norm	gutes	/	kräftiges	Wasser	bekommen	.		Andere	Weg	Wasser	zu	destillieren	.
p	p												
pb	pb												
pos	ADJA	$(	ADJA	NN	VVINF	$.		ADJA	NN	NN	PTKZU	VVINF	$.
tok	gutes	/	kräfftiges	Waſſer	bekommen	.		Andere	weg	waſſer	zu	di=ſtillirn	.

Fig. 10 The ‘norm’ segmentation layer is interrupted by a ‘figure’ annotation, which covers none of the primary textual data

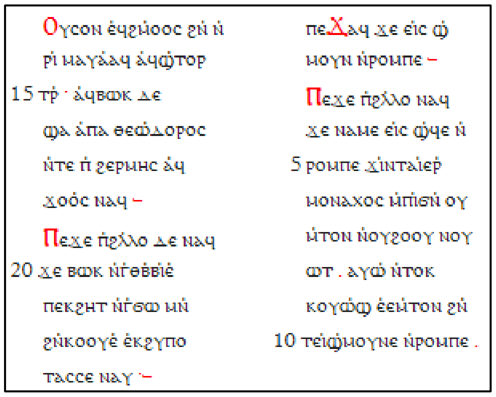
However, since ‘norm’ is a segmentation layer, it is possible to search for consecutive items while ignoring this gap. For example, we can search for adjectives following sentence ending punctuation (`pos="$."`) to find sentence initial adjectives, even though sentence boundaries are not annotated in the corpus. In the case depicted in Figure 10, the first query below will not retrieve the period and following adjective *Andere* ‘other’, but the second query will, since it specifies that the two search nodes are adjacent *within the norm segmentation*:

```
(15)    pos="$." & pos="ADJA" & #1 . #2
(16)    pos="$." & pos="ADJA" & #1 .norm #2
```

A second consideration in representing historical corpora is producing faithful yet readable full-text visualizations. Here again it is often necessary to offer a diplomatic and a normalized view. While PDF facsimiles as in Figure 9 are useful in attaining the former goal, these are often insufficient, since facsimiles can be very difficult to read for students and researchers not trained in paleography. The diplomatic view should therefore be similar to a digital diplomatic edition. The Coptic SCRIPTORIUM corpora[10] follow this approach using the generic HTML visualization technique described in Section 3.2. The approach is particularly well suited for diplomatic layouting, since unlike the syntactic annotations and other graphs described in the previous section, the information to be presented is non-relational. Instead, areas of the text are rendered according to the presence or absence of some annotations at different positions. This is illustrated in Figure 11 for a Sahidic Coptic corpus from the Apophthegmata Patrum (Sayings of the Desert Fathers), which shows normalized and diplomatic views of the same manuscript.


ⲟⲩⲥⲟⲛ ⲉϥϩⲙⲟⲟⲥ ϩⲛ ⲛⲣⲓ ⲙⲁⲩⲁⲁϥ ⲁϥϣⲧⲟⲣⲧⲣ · ⲁϥⲃⲱⲕ ⲇⲉ ϣ
ⲁϥϫⲟⲟⲥ ⲛⲁϥ – ⲡⲉϫⲉ ⲡϩⲗⲗⲟ ⲇⲉ ⲛⲁϥ ϫⲉ ⲃⲱⲕ ⲛⲅⲑⲃⲃⲓⲉ ⲡⲉⲕϩ
ⲛⲁⲩ ·– ⲁϥⲃⲱⲕ ⲛ ⲁ
ⲉⲓⲟⲩⲏϩ ⲙⲛ ⲛⲣⲱ ⲧⲉⲕ
ⲛⲕⲟⲟⲩⲉ ⲛⲅⲛⲁⲙⲧⲟⲛ ⲁⲛ ·– ⲉⲧⲃⲉ ⲟⲩ ⲁⲕⲉⲓ ⲉⲃⲟⲗ ⲉⲣⲙⲟⲛⲁⲭⲟⲥ ·

And the old man said to him

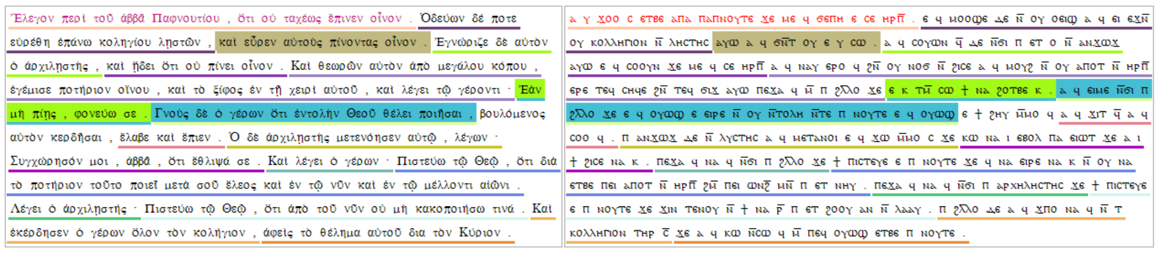

Ⲟⲩⲥⲟⲛ ⲉ̀ϥϩⲙ̀ⲟⲟⲥ ϩⲛ̀ ⲛ̀
ⲣ̀ⲓ ⲙⲁⲩⲁ̀ⲁϥ ⲁ̀ϥϣ̀ⲧⲟⲣ
15 ⲧⲣ̀ · ⲁ̀ϥⲃⲱⲕ ⲇⲉ
ϣⲁ ⲁ̀ⲡⲁ ⲑⲉⲱ̀ⲇⲟⲣⲟⲥ
ⲛ̀ⲧⲉ ⲡ̀ ϩⲉⲣⲙⲏⲥ ⲁ̀ϥ
ϫⲟⲟ̀ⲥ ⲛⲁϥ –
Ⲡⲉϫⲉ ⲡ̀ϩⲗⲗⲟ ⲇⲉ ⲛⲁϥ
20 ϫⲉ ⲃⲱⲕ ⲛ̀ⲅⲑⲃ̀ⲃⲓ̀ⲉ̀
ⲡⲉⲕϩⲏⲧ ⲛ̀ⲅϭⲱ ⲙⲛ̀
ϩⲛ̀ⲕⲟⲟⲩⲉ̀ ⲉ̀ⲕϩⲩⲡⲟ
ⲧⲁⲥⲥⲉ ⲛⲁⲩ ·–

ⲡⲉϪⲁϥ ϫⲉ ⲉ̀ⲓⲥ ϥ̀
ⲙⲟⲩⲛ ⲛ̀ⲣⲟⲙⲡⲉ –
Ⲡⲉϫⲉ ⲡ̀ϩⲗⲗⲟ ⲛⲁϥ
ϫⲉ ⲛⲁⲙⲉ ⲉ̀ⲓⲥ ϥ̀ϥⲉ ⲛ̀
5 ⲣⲟⲙⲡⲉ ϫ̀ⲓⲛⲧⲁ̀ⲓⲉⲣ̀
ⲙⲟⲛⲁⲭⲟⲥ ⲙ̀ⲡⲓ̀ϭⲛ̀ ⲟⲩ
ⲙ̀ⲧⲟⲛ ⲛ̀ⲟⲩϩⲟⲟⲩ ⲛⲟⲩ
ⲱⲧ . ⲁⲩⲱ̀ ⲛ̀ⲧⲟⲕ
ⲕⲟⲩⲱ̀ϣ ⲉ̀ⲉⲓ̀ⲧⲟⲛ ϩⲛ̀
10 ⲧⲉ̀ⲓϣⲙⲟⲩⲛⲉ ⲛ̀ⲣⲟⲙⲡⲉ .

Fig. 11 Two views of a Coptic manuscript: normalized (left) and diplomatic (right)

The normalized visualization on the left is focused on conveying textual content: it does not contain diacritic markings, the division of the text into lines wraps depending on the browser window size, and a translation is shown when the text is hovered over. The visualization on the right, by contrast, retains line and column breaks (EpiDoc TEI ‘lb’ and ‘cb’ annotations), gives line numbers in the manuscript, and retains all orthographic idiosyncrasies and rendering details of the text (e.g. ink colors, oversized letters). Both these visualizations are generated using the same generic module, in this case by applying a formatted absolutely positioned HTML <div> to render lines, or using an HTML title attribute and conditional formatting to create the hovering translation tooltip.

Needless to say, the same annotations can also be used in other visualizations: for example Figure 12 shows the ‘norm’ annotations being represented in the dedicated ‘highlight and underline’ visualization that was used for coreference in Figure 7. In this case, the visualization has been configured to co-highlight alignment edges between Greek and Coptic versions of the Apophthegmata Patrum, texts which deviate from and complement each other at many points.


Ἔλεγον περὶ τοῦ ἀββᾶ Παφνουτίου , ὅτι οὐ ταχέως ἔπινεν οἶνον . Ὁδεύων δέ ποτε
εὑρέθη ἐπάνω κοληγίου λῃστῶν , καὶ εὗρεν αὐτοὺς πίνοντας οἶνον . Ἐγνώριζε δὲ αὐτὸν
ὁ ἀρχιλῃστὴς , καὶ ᾔδει ὅτι οὐ πίνει οἶνον . Καὶ θεωρῶν αὐτὸν ἀπὸ μεγάλου κόπου ,
ἐγέμισε ποτήριον οἴνου , καὶ τὸ ξίφος ἐν τῇ χειρὶ αὐτοῦ , καὶ λέγει τῷ γέροντι · Ἐὰν
μὴ πίῃς , φονεύω σε . Γνοὺς δὲ ὁ γέρων ὅτι ἐντολὴν Θεοῦ θέλει ποιῆσαι , βουλόμενος
αὐτὸν κερδῆσαι , ἔλαβε καὶ ἔπιεν . Ὁ δὲ ἀρχιλῃστὴς μετενόησεν αὐτῷ , λέγων ·
Συγχώρησόν μοι , ἀββᾶ , ὅτι ἔθλιψά σε . Καὶ λέγει ὁ γέρων · Πιστεύω τῷ Θεῷ , ὅτι διὰ
τὸ ποτήριον τοῦτο ποιεῖ μετὰ σοῦ ἔλεος καὶ ἐν τῷ νῦν καὶ ἐν τῷ μέλλοντι αἰῶνι .
Λέγει ὁ ἀρχιλῃστής · Πιστεύω τῷ Θεῷ , ὅτι ἀπὸ τοῦ νῦν οὐ μὴ κακοποιήσω τινά . Καὶ
ἐκέρδησεν ὁ γέρων ὅλον τὸν κολήγιον , ἀφεὶς τὸ θέλημα αὐτοῦ δια τὸν Κύριον .

ⲁ ⲩ ϫⲟⲟ ⲥ ⲉⲧⲃⲉ ⲁⲡⲁ ⲡⲁⲡⲛⲟⲩⲧⲉ ϫⲉ ⲙⲉ ϥ ϭⲉⲡⲏ ⲉ ⲥⲉ ⲏⲣⲡ̄ . ⲉ ϥ ⲙⲟⲟϣⲉ ⲇⲉ ⲛ̄ ⲟⲩ ⲟⲉⲓϣ ⲁ ϥ ⲉⲓ ⲉϫⲛ̄
ⲟⲩ ⲕⲟⲗⲗⲏⲅⲓⲟⲛ ⲛ̄ ⲗⲏⲥⲧⲏⲥ ⲁⲩⲱ ⲁ ϥ ϭⲛ̄ⲧ ⲟⲩ ⲉ ⲩ ⲥⲱ . ⲁ ϥ ⲥⲟⲩⲱⲛ ϥ̄ ⲇⲉ ⲛ̄ϭⲓ ⲡ ⲉⲧ ⲟ ⲛ̄ ⲁⲛϫⲱϫ
ⲁⲩⲱ ⲉ ϥ ⲥⲟⲟⲩⲛ ϫⲉ ⲙⲉ ϥ ⲥⲉ ⲏⲣⲡ̄ ⲁ ϥ ⲛⲁⲩ ⲉⲣⲟ ϥ ϩⲛ̄ ⲟⲩ ⲛⲟϭ ⲛ̄ ϩⲓⲥⲉ ⲁ ϥ ⲙⲟⲩϩ ⲛ̄ ⲟⲩ ⲁⲡⲟⲧ ⲛ̄ ⲏⲣⲡ̄
ⲉⲣⲉ ⲧⲉϥ ⲥⲏϥⲉ ϩⲛ̄ ⲧⲉϥ ϭⲓϫ ⲁⲩⲱ ⲡⲉϫⲁ ϥ ⲙ̄ ⲡ ϩⲗⲗⲟ ϫⲉ ⲉ ⲕ ⲧⲙ̄ ⲥⲱ ϯ ⲛⲁ ϩⲟⲧⲃⲉ ⲕ . ⲁ ϥ ⲉⲓⲙⲉ ⲛ̄ϭⲓ ⲡ
ϩⲗⲗⲟ ϫⲉ ⲉ ϥ ⲟⲩⲱϣ ⲉ ⲉⲓⲣⲉ ⲛ̄ ⲟⲩ ⲛ̄ⲧⲟⲗⲏ ⲛ̄ⲧⲉ ⲡ ⲛⲟⲩⲧⲉ ⲉ ϥ ⲟⲩⲱϣ ⲉ ϯ ϩⲏⲩ ⲙ̄ⲙⲟ ϥ ⲁ ϥ ϫⲓⲧ ϥ̄ ⲁ ϥ
ⲥⲟⲟ ϥ . ⲡ ⲁⲛϫⲱϫ ⲇⲉ ⲛ̄ ⲗⲩⲥⲧⲏⲥ ⲁ ϥ ⲙⲉⲧⲁⲛⲟⲓ ⲉ ϥ ϫⲱ ⲙ̄ⲙⲟ ⲥ ϫⲉ ⲕⲱ ⲛⲁ ⲓ ⲉⲃⲟⲗ ⲡⲁ ⲉⲓⲱⲧ ϫⲉ ⲁ ⲓ
ϯ ϩⲓⲥⲉ ⲛⲁ ⲕ . ⲡⲉϫⲁ ϥ ⲛⲁ ϥ ⲛ̄ϭⲓ ⲡ ϩⲗⲗⲟ ϫⲉ ϯ ⲡⲓⲥⲧⲉⲩⲉ ⲉ ⲡ ⲛⲟⲩⲧⲉ ϫⲉ ϥ ⲛⲁ ⲉⲓⲣⲉ ⲛⲁ ⲕ ⲛ̄ ⲟⲩ ⲛⲁ
ⲉⲧⲃⲉ ⲡⲉⲓ ⲁⲡⲟⲧ ⲛ̄ ⲏⲣⲡ̄ ϩⲙ̄ ⲡⲉⲓ ⲱⲛϩ̄ ⲙⲛ̄ ⲡ ⲉⲧ ⲛⲏⲩ . ⲡⲉϫⲁ ϥ ⲛⲁ ϥ ⲛ̄ϭⲓ ⲡ ⲁⲣⲭⲏⲗⲏⲥⲧⲏⲥ ϫⲉ ϯ ⲡⲓⲥⲧⲉⲩⲉ
ⲉ ⲡ ⲛⲟⲩⲧⲉ ϫⲉ ϫⲓⲛ ⲧⲉⲛⲟⲩ ⲛ̄ ϯ ⲛⲁ ⲣ̄ ⲡ ⲉⲧ ϩⲟⲟⲩ ⲁⲛ ⲛ̄ ⲗⲁⲁⲩ . ⲡ ϩⲗⲗⲟ ⲇⲉ ⲁ ϥ ϫⲡⲟ ⲛⲁ ϥ ⲛ̄ ⲧ
ⲕⲟⲗⲗⲏⲅⲓⲟⲛ ⲧⲏⲣ ⲥ̄ ϫⲉ ⲁ ϥ ⲕⲱ ⲛ̄ⲥⲱ ϥ ⲙ̄ ⲡⲉϥ ⲟⲩⲱϣ ⲉⲧⲃⲉ ⲡ ⲛⲟⲩⲧⲉ .

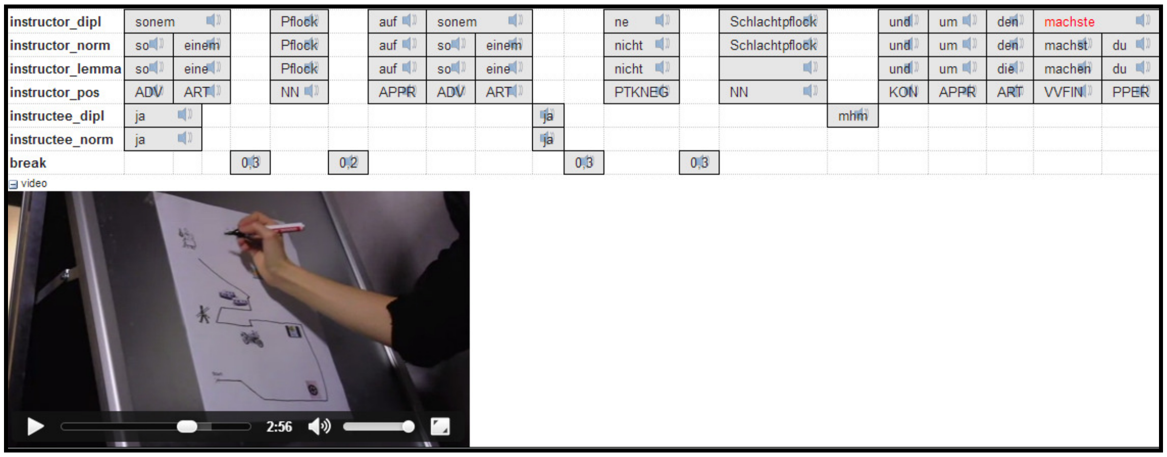
Fig. 12 Sentence alignment of a Greek and Coptic text

Since the views discussed here are dynamically generated from the data-model, new ‘digital editions’ can be created to fulfill different users’ needs without intensive development of new software components. The only requirement is annotations encoding the distinctions that are to be visualized and a sensible translation from data-model to elements in the browser.

4.3 Multimodal Spoken Language Corpora

As a final challenge for a generic corpus architecture, we now turn to consider multi-modal, spoken dialogue data. The challenges in modeling dialogue data have already been alluded to in Section 2.3: simultaneous speakers immediately lead to conflicting token streams and challenge the idea that adjacency means 'what happens at the next token position'. By using segmentation layers to represent text from multiple speakers, it becomes possible to speak of 'absolute adjacency' (the next thing that happens in the corpus timeline) versus 'layer adjacency', meaning the next thing some speaker says (although annotations and text belonging to other speakers may occur between two layer-adjacent events). The same applies to distance-based searches (within *n* to *m* units)

A further challenge lies in integrating a timeline on which all utterances and possibly extralinguistic events (e.g. noise, visual stimuli, etc.) are aligned. Although the data-model Salt has a built-in timeline concept, the database representation of temporal events in ANNIS saves time-alignment information just like any other annotation: start and end times are simply labels applied to positions in the graph. When result sets are retrieved, however, these annotations are interpreted as time alignment information in Salt, e.g. for the purpose of navigating in an A/V player plugin. By making the time annotations accessible to visualizer modules, it is possible to click on an annotation in one visualization and 'jump' to a corresponding point in the A/V stream. Figure 13 illustrates the mechanism in BeMaTaC (Berlin Map Task Corpus),[11] a corpus of spoken German dialogues in a map task setting, where one speaker, the instructor, guides another, the instructee, using a set of two nearly identical maps (cf. Anderson et al. 1991).



instructor_dipl	sonem				Pflock		auf	sonem				ne		Schlachtpflock		und	um	den	machste	
instructor_norm	so	einem			Pflock		auf	so	einem			nicht		Schlachtpflock		und	um	den	machst	du
instructor_lemma	so	eine			Pflock		auf	so	eine			nicht				und	um	die	machen	du
instructor_pos	ADV	ART			NN		APPR	ADV	ART			PTKNEG		NN		KON	APPR	ART	VVFIN	PPER
instructee_dipl	ja									ja					mhm					
instructee_norm	ja									ja										
break				0,3		0,2					0,3		0,3							

Fig. 13 Dialogue data from BeMaTaC with an aligned A/V stream

As the figure shows, two speakers, the instructor and the instructee, are speaking at the same time (the instructee is 'back-channeling', saying *ja* 'yes' etc., to signal following the instructor's explanation). The speakers' words may not overlap exactly (e.g. the instructee's first word *ja* on the left lasts well into the second word of the instructor), but these cases can be queried using appropriate operators in AQL. For example, the overlap operator `_o_` retrives the case above:

(17) `instructor_lemma="so" _o_ instructee_dipl="ja"`

It is also possible for multiple normalized word-forms to fuse phonetically, as shown in the form *machste* ‘you do’ (marked red on the right), which results from a fusion of *machst* + *du* (‘do’ + ‘you’). Both forms belong to a segmentation layer, so that both phonetic and normalized word representations can be chosen as a reference for context, search and visualization purposes. It is also possible to define layers smaller than the reference word form (e.g. queries for adjacent syllables with their own precedence layers in AQL, i.e. an operator like `.syl` for adjacent syllables), but this has not been done in this corpus. Layers not associated with any speakers, such as the ‘break’ layer are also possible, and do not disrupt layer adjacency for consecutive utterances of the same speaker. In this way, the problems involved with representing historical texts and dialogue data turn out to be similar, and the same solution using segmentation layers leads to the desired behavior of the digital corpus representation.

5 Conclusion

This paper has argued for a generic approach to language corpus search and visualization similar to the direction taken by recent corpus format standardization efforts. We have laid out some of the requirements and typical characteristics of underlying data-models that can realize this goal. By adopting a format-independent meta-model that is semantically generic yet suited to representing language data (Section 2), it has been possible to use the same ANNIS architecture for a very broad range of language data coming from different formats (Section 4), without having to design a myriad of systems for spoken data, historical corpora, parallel corpora, learner corpora, etc. This is increasingly being enabled by using the highly configurable nature of style sheet-based visualization methods (Section 3.2) and the extension of the SaltNPepper framework to cover additional corpus formats.

At the same time, a generic approach cannot be successful if it compromises on the type of accessibility that different types of data and research questions demand. It is therefore necessary to offer not only a high degree of configurability to suit individual corpus needs, but also to be willing to develop dedicated solutions where necessary, and embed these within a pluggable context that allows the development of individual modules to be encapsulated from the other services offered by the system (e.g. a special view is needed for rhetorical structure annotations, but these annotations can be searched using the same query engine without modification). A research avenue we are currently working on in this context is the pluggability of different graphical query builders to allow an individualized generation of AQL queries in a manner that is easier and better suited to individual research communities. Corpus query languages are likely to become a locus of much debate in future work on corpus standardization and interoperability (cf. Bański & Witt 2011).

There are also a number of caveats and limitations to the system, which we would like to concentrate on for future development. One limitation is the current inability to negate operators with no implication of node existence. While it is possible to search for nodes dominated by something other than a cat="PP" (using AQL cat!="PP"), it is not possible to look for nodes not dominated by a PP, including cases where a node is not dominated by anything at all. This operation can be considerably expensive in computation time and would at present not perform well in large corpora if the remainder of the query is not very restrictive.

A further limitation for use in the digital humanities is the context of ANNIS as primarily a search system. For many purposes, it would be interesting to use the ANNIS data model to allow close reading of annotated documents, either for discourse studies or outside of narrow linguistics and in the broader humanities environment. This might mean that ANNIS documents should be readable not only by going through the search interface, but by embedding generic ANNIS visualizer output in websites created for other projects. We are therefore currently working on API access to allow document visualizations to be embedded in pages outside of the full search interface, giving researchers easy access to the richly structured corpora.

The resources described in this paper are almost all freely available and open source, in particular both ANNIS and SaltNPepper. It is our hope that making resources available will lead more researchers to reuse these tools and reduce the hurdles involved in creating complex, multilayer corpora. Lessening the need to develop corpus software and conversion tools means more time and freedom from technological restrictions to make richer corpora. A generic multilayer architecture offers the promise of better annotation schemes, of exploring the interdependence between parallel or multiple annotations coming from different researchers, theories or tools, and of representing data in a way that is appropriate for its domain, without compromising on the needs of machine readable data in the information age.

Notes

[1] Most prominent is perhaps the standardization of GrAF XML as part of the Linguistic Annotation Framework LAF (ISO 24615), though the morphological, syntactic and semantic frameworks (MAF, SynAF and SemAF) also represent important milestones; cf. Ide & Suderman (2007) on GrAF and Stührenberg (2012) for a concise overview. Several initiatives in Europe within the framework of CLARIN (http://clarin.eu/) are also focusing increasingly on interchangeable and generic data models (for example pipelining corpus data in WebLicht (https://weblicht.sfs.uni-tuebingen.de/).

[2] There are numerous terminologies when referring to graphs with some differences, but for the present purpose we will regard a graph as a collection of elements called nodes (sometimes referred to as

'vertices') that may be linked to each other via directed links referred to as 'edges'. In our case, annotation graphs are always anchored to primary data (there are no disconnected groups of nodes) and there are no cyclical sets of edges (i.e. a set of directed edges creating a circle or 'cycle') which belong to the same edge type (though indirect cycles resulting from multiple annotation layers are supported).

[3] A good example is Sanskrit, in which sandhi assimilations and the Devanagari script lead to single letters belonging to two words, whereas linguistic transcription usually normalizes text into separate word forms.

[4] However one key feature that is supported by CQP and related languages and is not currently implemented in ANNIS is regular expressions over search nodes, e.g. finding a range of between 3-4 occurrences of a search term etc. Users must currently specify all variants of their search explicitly (either 3 occurrences, or 4, or…). This limitation does not apply to searching within a certain distance (e.g. within 3-4 tokens) or edge depth (dominance of 3-4 'generations').

[5] It is also possible to name nodes for convenience rather than use numbers. Thus for sentence 'sent' starting with two prepositional phrases 'pp', we can write: `PP1#pp . PP2#pp & sent > #PP1 & sent > #PP2`. This indicates that we are looking for two adjacent nodes with an annotation named pp, which have been named PP1 and PP2 for this search, and a node annotated with an annotation named sent, which dominates the other two nodes. For more detailed documentation on AQL operators and notation, see the ANNIS website.

[6] The annotation name 'cat' has been used for phrases in all three corpora, following TigerSearch conventions. The PTB format originally specified no annotation name for phrases, but only values and positions in the bracket structure. For a different approach to representing and naming treebanks with specific reference to TüBa-D/Z, see Krause et al. (2011).

[7] An anonymous reviewer has pointed out that lack of familiarity with the underlying structure of a corpus can be a substantial barrier for new users. We couldn't agree more with this assessment, and indeed in our experience corpora which take advantage of the example query facility, offer linked documentation and even tutorials with linked queries to ANNIS are better received than those that do not. On some level, we believe that this difficulty is inevitable, and facilities that try to eschew this difficulty, for example by correcting users' queries (depending on the correction) or trying to guess their intentions, risk doing more harm than good. A complex corpus is, at the end of the day, complex, and this underscores the importance of good documentation.

[8] Note that in either case, separation of word forms or tokens is up to the corpus designer in the respective format being worked with. ANNIS does not tokenize data on the fly, but respects tokenizations and other segmentations coming from a variety of formats and tools.

[9] The corpus and its full documentation are freely available from http://korpling.german.hu-berlin.de/ridges/ under a Creative Commons license.

[10] Available under a Creative Commons license from http://coptic.pacific.edu/.

[11] Available at: https://www.linguistik.hu-berlin.de/institut/professuren/korpuslinguistik/forschung/bematac.

References

Anderson, A. H., Bader, M., Bard, E. G., Boyle, E., Doherty, G., Garrod, S., Isard, S., Kowtko, J., McAllister, J., Miller, J., Sotillo, C., Thompson, H., and Weinert, R. (1991). The HCRC Map Task Corpus. *Language and Speech* 34: 351–366.

Bański, P., and Przepiórkowski, A. (2009). Stand-off TEI Annotation: The Case of the National Corpus of Polish. In *Proceedings of the Third Linguistic Annotation Workshop, ACL-IJCNLP 2009*. Suntec, Singapore, pp. 64–67.

Bański, P., and Witt, A. (2011). Do Linguists Need a Corpus Query Lingua Franca? In *ISO TC37 Meeting in Seoul, South Korea, 13 June 2011.*

Brants, S., Dipper, S., Hansen, S., Lezius, W., and Smith, G. (2002). The TIGER Treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories, September 20-21 (TLT02)*. Sozopol, Bulgaria, pp. 24–42.

Brugman, H., and Russel, A. (2004). Annotating Multimedia/Multi-modal resources with ELAN. In *Proceedings of LREC 2004, Fourth International Conference on Language Resources and Evaluation*. Paris: ELRA, pp. 2065–2068.

Cayless, H., Roueché, C., Elliott, T., and Bodard, G. (2009). Epigraphy in 2017. *Digital Humanities Quarterly* 3(1). http://www.digitalhumanities.org/dhq/vol/3/1/000030/000030.html (accessed 10.9.2013).

Christ, O. (1994). A Modular and Flexible Architecture for an Integrated Corpus Query System. In *Proceedings of Complex 94. 3rd Conference on Computational Lexicography and Text Research*. Budapest, pp. 23–32.

Claridge, C. (2008). Historical Corpora. In A. Lüdeling, and M. Kytö (eds), *Corpus Linguistics. An Internation Handbook*. Vol. 1. Berlin: Mouton de Gruyter, pp. 242–259.

Dipper, S. 2005. XML-based stand-off representation and exploitation of multi-level linguistic annotation. In *Proceedings of Berliner XML Tage 2005 (BXML 2005)*. Berlin, 39–50.

Evert, S., and Voormann, H. (2003). *NQL – A Query Language for Multi-Modal Language Data*. Technical report, IMS, University of Stuttgart.

Ghodke, S., and Bird, S. (2012). Fangorn: A System for Querying Very Large Treebanks. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING '12)*. Mumbai, India.

Haug, D. T., Eckhoff, H. M., Majer, M., and Welo, E. (2009). Breaking Down and Putting Back Together: Analysis and Synthesis of New Testament Greek. *Journal of Greek Linguistics* 9(1): 56–92.

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2006). OntoNotes: The 90% Solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. New York: Association for Computational Linguistics, pp. 57–60. http://www.aclweb.org/anthology/N/N06/N06-2015 (accessed 10.9.2013).

Ide, N., and Suderman, K. (2007). GrAF: A Graph-based Format for Linguistic Annotations. In *Proceedings of the Linguistic Annotation Workshop 2007*. Prague, pp. 1–8.

ISO 24612 (2012). *Language Resource Management – Linguistic Annotation Framework (LAF)*.

Janus, D., and Przepiórkowski, A. (2007). Poliqarp: An Open Source Corpus Indexer and Search Engine with Syntactic Extensions. In *Proceedings of the ACL 2007 Demo and Poster Sessions*. Prague, pp. 85–88.

Kilgarriff, A., Baisa, V., Bušta, J., Jakubíček, M., Kovář, V., Michelfeit, J., Rychlý, P., and Suchome, V. (2014). The Sketch Engine: Ten Years On. *Lexicography* 1(1), pp. 7–36.

Krause, T., Lüdeling, A., Odebrecht, C., and Zeldes, A. (2012). Multiple Tokenizations in a Diachronic Corpus. In *Exploring Ancient Languages through Corpora*. Oslo.

Krause, T., Ritz, J., Zeldes, A., and Zipser, F. (2011). Topological Fields, Constituents and Coreference: A New Multi-layer Architecture for TüBa-D/Z. In H. Hedeland, T. Schmidt, and K. Wörner (eds), *Multilingual Resources and Multilingual Applications. Proceedings of the Conference of the German Society for Computational Linguistics and Language Technology (GSCL) 2011*. (Working Papers in Multilingualism 96.) Hamburg: Universität Hamburg, pp. 259–262.

Lezius, W. (2002). *Ein Suchwerkzeug für syntaktisch annotierte Textkorpora*. PhD Thesis, Institut für maschinelle Sprachverarbeitung Stuttgart.

Mann, W. C., and Thompson, S. A. (1988). Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text* 8(3): 243–281.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Special Issue on Using Large Corpora, Computational Linguistics* 19(2): 313–330.

Müller, C., and Strube, M. (2006). Multi-level annotation of linguistic data with MMAX2. In S. Braun, K. Kohn, and J. Mukherjee (eds.), *Corpus Technology and Language Pedagogy*. Frankfurt: Peter Lang, 197–214.

Reznicek, M., Walter, M., Schmid, K., Lüdeling, A., Hirschmann, H., Krummes, C., and Andreas, T. (2010). *Das Falko-Handbuch. Korpusaufbau und Annotationen. Version 1.0.1*.

Schmid, H. (2008). Tokenizing and Part-of-Speech Tagging. In A. Lüdeling, and M. Kytö (eds), *Corpus Linguistics. An International Handbook*. Vol. 1. Berlin: Mouton de Gruyter, pp. 527–551.

Schmidt, T., and Wörner, K. (2009). EXMARaLDA – Creating, Analysing and Sharing Spoken Language Corpora for Pragmatic Research. *Pragmatics* 19(4): 565–582.

Stede, M. (2004). The Potsdam Commentary Corpus. In B. Webber, and D. K. Byron (eds), *Proceeding of the ACL-04 Workshop on Discourse Annotation*. Barcelona, Spain, pp. 96–102.

Steinberg, D., Budinsky, F., Paternostro, M., and Merks, E. (2009). *EMF: Eclipse Modeling Framework 2.0*. Upper Saddle River, NJ: Addison-Wesley.

Štěpánek, J., and Pajas, P. (2010). Querying Diverse Treebanks in a Uniform Way. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC-2010)*. Valletta, Malta, pp. 1828–1835.

Stührenberg, M. (2012). The TEI and Current Standards for Structuring Linguistic Data. An Overview. *Journal of the Text Encoding Initiative* 3. http://jtei.revues.org/523 (accessed 10.9.2013).

Telljohann, H., Hinrichs, E. W., Kübler, S., Zinsmeister, H., and Beck, K. (2009). *Stylebook for the Tübingen Treebank of Written German (TüBa-D/Z).*

Volk, M., Göhring, A., Marek, T., and Samuelsson, Y. (2010). *SMULTRON (version 3.0) — The Stockholm MULtilingual parallel TReebank.* Institute of Computational Linguistics, University of Zurich.

Zipser, F., and Romary, L. (2010). A Model Oriented Approach to the Mapping of Annotation Formats using Standards. In *Proceedings of the Workshop on Language Resource and Language Technology Standards, LREC-2010*. Valletta, Malta, pp. 7–18.