Requirements of a User-Friendly, General-Purpose Corpus Query Interface

Jan-Philipp Soehn1, Heike Zinsmeister2, Georg Rehm1

1 Tübingen University
Sonderforschungsbereich 441
Nauklerstraße 35
72074 Tübingen, Germany

2 Konstanz University
Department of Linguistics
Fach D 185
78457 Konstanz, Germany

Abstract

This article reports on a survey that was conducted among 16 projects of a collaborative research centre to learn about the requirements of a web-based corpus query interface. This interface is to be created for a collection of corpora that are heterogeneous with respect to their languages, levels of annotations, and their users’ research interests. Based on the survey and a comparison of three existing corpus query interfaces we compiled a set of requirements. In the context of sustainable strategies of corpus storage and accessibility we point out how to design an interface that is general enough to cover multiple corpora and at the same time suitable for a wide range of users.

1. Introduction

Immense amounts of corpus data have been created in recent years. The process of building a language resource is expensive, time-consuming, and it includes aspects such as corpus sampling and linguistic annotation on multiple levels. There is an urgent need to ensure that researchers are able to access data collections such as these beyond the lifetime of the project that created the resource. Issues of sustainability and preservation are increasingly important to the community; see, for example, Bird and Simons (2003), Trilsbeek and Wittenburg (2006), Dipper et al. (2006) as well as efforts such as OLAC (http://www.language-archives.org), E-MELD (http://emeld.org), and metadata aggregators such as the Digital Repository Infrastructure for European Research (http://www.driver-repository.eu).

One major aspect of sustainability is perpetuating access to corpora independently of project duration, availability of the researchers who built the resource, and development cycles of operating systems, tools, and applications. There is a great danger of a language resource turning into an expensive data graveyard if the tools for accessing, displaying, and searching the resource become obsolete or if there is no proper documentation available for the respective data collection (Bird and Simons, 2003; Schmidt et al., 2006).

A straightforward way out of this problem is to adhere to a particular annotation and encoding standard so that only one common interface needs to be supported for accessing a whole range of resources (Lehmberg and Wörner, In print; Rehm et al., 2007; Rehm et al., 2008a; Rehm et al., 2008b; Witt et al., 2007; Zinsmeister et al., In print). The availability of such an interface would lead to two new challenges. First, due to the diversity of information that needs to be accessed, the interface must be general enough to cover multiple corpora with heterogeneous annotation and it must be specific enough to enable users to find the information that they are looking for. Second, due to the diversity of potential users, the query interface has to be designed to favour high acceptability. Such a user interface should assist users who cannot be expected to be experts in composing queries in, for example, a formal query language that is based on first-order logic. At the same time the interface should do justice to the experienced user and support efficient data access. Thus, alternative approaches have to be explored to facilitate accessing and querying linguistic resources for a heterogeneous group of users.

The goal of this article is twofold. On the one hand we outline a set of general requirements for a sustainable corpus query interface, on the other we report on ongoing work of implementing such a general-purpose linguistic query interface for a set of heterogeneous corpus resources. Both efforts build upon a survey conducted among 16 projects of the German collaborative research centre 441 at Tübingen University supplemented by a qualitative analysis of three existing corpus interfaces which we take to be prototypical representatives of specific types of corpus interfaces. It is worth pointing out that we do not discuss query languages such as such but take it for granted that a user-friendly interface is independent of the underlying query language. For surveys on the expressiveness of query languages see, for example, Lai and Bird (2004) or Dipper et al. (2007).

This article is structured as follows: In Section 2 the survey is reported. We present the results by aggregating the answers given to us by the project staff. Section 3 presents three existing corpus query interfaces, comparing and summarising their respective functions. In Section 4, we outline some of the requirements for the query interface that we collected based on the survey as well as from our analyses of the query interfaces. Section 5 gives a detailed overview of a corpus query interface that is currently under development. Its design is guided by the results of our studies from Sections 2 and 3. Finally, Section 6 rounds off this paper with a conclusion and an outlook on future work.

2. Survey of Requirements

This contribution reports on a survey we conducted to learn about the requirements of a web-based corpus query interface. This interface is to be created for a collection of corpora that are diverse with respect to their languages, levels of annotations, and research interests of the users, who, furthermore, come from several communities, each with their own standards and traditions (Witt et al., 2007; Rehm et al., 2007; Rehm et al., 2008a). Based on a questionnaire (Lehmberg et al., 2007, describe a related approach), we interviewed the research staff of 16 projects based in the
The respondents' areas of expertise:

- Computational Linguistics: 6 (30%)
- German Language: 3 (15%)
- Romance Languages: 3 (15%)
- Slavic Languages: 3 (15%)
- General Linguistics: 2 (10%)
- English Language: 1 (5%)
- Psycholinguistics: 1 (5%)
- Tibetan Language: 1 (5%)

Among them with a specialisation in:
- Language Acquisition: 2
- Semantics: 1

Table 1: Demographic characteristics of the questionnaire respondents

<table>
<thead>
<tr>
<th>Programming skills</th>
<th>yes: 45%</th>
<th>no: 55%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data creation</td>
<td>involved: 75%</td>
<td>not involved: 25%</td>
</tr>
<tr>
<td>Age</td>
<td>&lt;30: 30%</td>
<td>30–40: 45%</td>
</tr>
<tr>
<td>Sex</td>
<td>female: 65%</td>
<td>male: 35%</td>
</tr>
</tbody>
</table>

Table 1 contains demographic information about the questionnaire respondents.

1. What kind of information will be requested by the user of your corpus (please give examples)?
2. Please give examples of frequent queries.
3. What is the input format of the query (text, XML, specialised query language, . . .)?
4. What are your requirements on a query form (beyond a simple text-field and a search button)? Are there any online tools you consider suitable?
5. What will be the format in which search results are displayed? Are there existing websites that use this format?

The respondents took two dimensions into account. First, they referred to the specific annotation, metadata and requirements of the corpora created in their respective projects. Most of them did not generalise with regard to the questions on adequate formats of search results or the query interface. Second, they considered their research interests and their formal background as well as their computer literacy. The answers to the survey are extremely heterogeneous, ranging from rather short to very detailed answers. To illustrate their broad range consider, for example, the following two answers to question 2 on example queries. On the one hand we got:

FSQ-query for subject wh-movement: (E y (& (cat y D) (E z (& (cat z W-Pron) (≥ y z))) (E x (& (cat x Trace) (mor x nom) (move x y)))))

and on the other:

Find all accented adjectives!
Find an activity verb in stative passive!

Table 2 contains a summary of the answers we received.

### 3. Existing Corpus Query Interfaces

In addition to the questionnaire we compare and summarise the functions of three corpus query interfaces that have been mentioned by respondents as suitable tools. In this way we can identify their features and components. These features were integrated into a requirements document (Rehm and Schonefeld, 2008) that specifies properties and functional areas of the query interface that is currently under development in the project Sustainability of Linguistic Data, a joint initiative of the Universities of Hamburg, Potsdam and Tübingen. The query interfaces that we examined as a complement to the questionnaire are COSMAS II, TIGERSearch, and ELAN, that can be conceptualised as three different types of corpus user interface. COSMAS II represents the general interface to query large amounts of textual data which takes into account positional, i.e., word-based annotation only. Other instances of this kind of interface are, for example, the web interface of the Corpus del Español (Davies, 2005), XSara the search tool accompanying the British National Corpus (http://www.oucs.ox.ac.uk/rts/xaera/), or the WordSmith tool (Scott, 2004).

TIGERSearch goes beyond positional information and allows the user to query and display hierarchical annotation and distributional relations. Other examples of this kind of interface include the fsq tool (Kepser, 2003) and the Linguist’s Search Engine (Resnik and Elkiss, 2005). ELAN is taken as a prototypical interface to multiple-layered annotated corpora which are organised according to a reference line. Related interfaces are provided by EXAKT (http://www.exmaralda.org/exakt.html) the search tool of EXMARaLDA (Schmidt, 2004).

The three example interfaces are all parts of highly accepted and widely used tools in their respective research communities. Only COSMAS II is implemented as a genuine online
1. Information requested by the user
Words/lemmas, strings, patterns (regular expressions), part-of-speech tags, morphological/prosodic annotation, syntactic structures, metadata (about source, date, etc.), specific elements and attributes in the XML structure

2. Examples of frequently used queries
Only project-specific responses were given ranging from structural dependencies (“cat1 dominates word1 & pos1”) over regular expressions (“[^ZJ\d\ae\ou\ym\?i]”) to very abstract natural-language queries (“find an activity verb in stative passive”)

3. Input format of the query
Text, graphical query interface (cf. TIGERSearch), macros or example queries as templates, FSQ

4. a. Requirements on a query form
Display frequent queries, features: save and name queries, drop-down menus of all categories that can be searched for (this feature should be hideable)

4. b. Existing online tools
Examples: COSMAS II (http://www.ids-mannheim.de/cosmas2/), CQP-Online (http://www.ims.uni-stuttgart.de/projekte/CQPDemos/Bundestag/frames-cqp.html), Corpus del Español (http://www.corpusdelespanol.org)

5. Display format of search results
The following options should be available: text (with links to tree graphs or audio files), KWIC with hideable/adjustable context, syntactic structure (constituents in brackets), cross-sentence discourse structure, search history, structured text (XML, spreadsheet), export to HTML, etc.

<table>
<thead>
<tr>
<th>4.1. Input Options</th>
</tr>
</thead>
</table>

For the search function a text-field should be provided that supports Unicode encoding, given the need to accommodate non-Latin (e.g., Russian or Tibetan) scripts. Alternatively, it would be advantageous if the user interface contained a graphical tool to assemble a query based on predefined graphical objects that represent linguistic concepts. These building blocks should range, for example, from part-of-speech categories such as different types of nouns (“proper name”, “inanimate object”), verbs (“ditransitive verb”), and prepositions, to grammatical functions (“genitive object”), or simply terminal and non-terminal nodes of a hierarchical structure, as well as to relations such as

4. General Requirements
In the following subsections, we outline requirements for a general query interface based on our findings on the questionnaire (Section 2) as well as from our analyses of existing query interfaces (Section 3).

4.1. Input Options
For the search function a text-field should be provided that supports Unicode encoding, given the need to accommodate non-Latin (e.g., Russian or Tibetan) scripts. Alternatively, it would be advantageous if the user interface contained a graphical tool to assemble a query based on predefined graphical objects that represent linguistic concepts. These building blocks should range, for example, from part-of-speech categories such as different types of nouns (“proper name”, “inanimate object”), verbs (“ditransitive verb”), and prepositions, to grammatical functions (“genitive object”), or simply terminal and non-terminal nodes of a hierarchical structure, as well as to relations such as
dominance and precedence. This requisite is reported by our informants in their answers 1 and 3 in Table 2. Users of TIGERSearch and COSMAS II are used to this twofold way of formulating queries; which of the two modes is most appropriate depends on the user’s preferences as well as on the type of query that is conducted.

4.2. Search Functions

The search function should be able to address primary data, multiple levels of annotation, and metadata. Frequent queries should be available as examples, represented both in a graphical and textual way, so that users who are not familiar with corpus query languages can use and modify them in order to explore the system capabilities as well as to arrive quickly at queries that are useful for their own research questions. This is further supported by a mapping of graphical queries into the textual query language syntax. In addition, a query form would be desirable for experienced users who would like to edit the underlying query formula directly. Though the interface is independent of a specific query language, we suggest to use XQuery, a language for finding and extracting elements and attributes from XML data, analogous to what SQL is for relational databases. XQuery is built on XPath expressions and standardised by the World Wide Web Consortium. It is rather easy to learn for an XML-experienced user and deployable in a broader range of applications. Moreover, the possibility to manipulate XQuery queries most directly meets the requirement to search for specific elements and attributes in the XML structure. Thus, XQuery is the obvious choice when it comes to picking a query formalism for XML-based linguistic resources.

Furthermore, a search history and a function to save and load queries (i. e., a kind of bookmark function) should be available just as in TIGERSearch (see row 4 a in Table 2). Lastly, a summary of all available search criteria and constraints, displayed via drop-down menus or similar means, would help the user in composing a query. For example, in COSMAS II, search operators with an intuitive description are displayed prominently within the search window and allow users to drag them into the search pane.

4.3. Visualisation

The query interface should cover linguistic patterns in a large and heterogeneous set of language resources. For the purpose of querying and visualising a corpus, all resources should be mapped onto abstract corpus types for type-specific query and visualisation methods. For example, the results for one specific corpus type are displayed as hyperlinked matches in a KWIC format, for another type as matrix of annotation layers, or as hierarchical tree structures. There should be functions that allow the user to include or exclude several layers of information in the display, such as complete sentences, information on words, or cross-sentence discourse annotation. In addition, the amount of visible context to the left and to the right should be customisable and there should be an option of enlarging the match up to a whole paragraph with cross-sentence annotation. Detailed tree structures that provide clickable nodes, and secondary/tertiary edges should be available where appropriate in suitable formats (e. g., SVG). Appropriate export formats (ODF, Excel, TXT, XML, HTML, etc.) are demanded by the researchers who participated in our questionnaire, both for the query results and for user-specified subsets of a corpus. An ID list of hits would be a useful feature to locate a particular result quickly. Statistical functions (frequencies, co-occurrences, mean utterance length, type-token ratio) analogous to COSMAS II complete the desirable functionality of the query interface.

This concludes our overview of the basic requirements. Certainly, we did not do justice to all of the features of COSMAS II, TIGERSearch, and ELAN but focused on the main properties relevant for a general query interface.

5. Our Corpus Query Interface

We are currently developing a corpus query interface for a sustainability web platform (see Section 1). The development process is completely guided by and based upon requirements that we collected in a survey (Section 2) and that we extracted from the feature sets of several existing and widely used corpus query tools (Section 3). Initially, we made a design decision and introduced a basic distinction that separates between querying for corpus metadata and querying for corpus data, i. e., corpus contents, so that we can tailor and fine-tune the respective functions.

A user has to log in first. From here, the user can either go to the saved queries area or explore the available metadata records. There are several different options how the metadata can be displayed, sorted, and searched (for example, by corpus type, by organisation or project, by properties such as number of tokens, or by the respective research question a corpus was created for). The implementation of this part of the interface is based on Java Server Pages and operates on a relational database due to performance and security considerations (Rehm et al., 2008b).

As soon as the user has decided upon one or more resources, the corpus contents of these collections can be queried using an intuitive graphical query interface that generalises as much as possible from the underlying data structures and querying methods actually used. The system employs Ajax technologies (Asynchronous JavaScript and XML) so that a dynamic, interactive, drag-and-drop-enabled query interface can be provided. An ontology of linguistic annotations (Rehm et al., 2008a) enables us to provide abstract representations of linguistic concepts (e. g., noun, verb, preposition etc.) that may have a specific set of features; operands can be used to glue together the linguistic concepts by dragging and dropping these graphical representations onto a specific area of the screen, building a query step by step. We also provide several output and visualisation modules for query results, e. g., queried corpus subsets that contain syntactic trees can be visualised as trees, and data that is modelled using a timeline-based approach is displayed in a tabular fashion.

Among other functions, the interface provides a graphical tree fragment query editor that allows the user to submit complex queries for retrieving those particular syntactic structures from the currently selected resources that match the tree fragment query. Queries are interpreted and translated into XQuery internally. When the interface is in tree
fragment query mode (see Figures 1 and 2), the user can drag and drop components of a query onto an assembly pane, so that queries can be constructed in a step-by-step fashion. Currently, nodes can be combined by dominance, precedence, and secondary edge relations. The structures defined by these graphs mirror the structures to be found by the XQuery engine of the native XML database that we use. A node may contain one or more conditions linked by boolean connectives that help to refine the node classes a specific query is supposed to match. Tree fragment queries are not the only type of queries allowed by the front-end. It also supports plain text and regular expression queries. Experienced users can formulate their queries in XQuery directly, or they can fine-tune queries initially generated graphically. Our aim is to give the user a variety of options for viewing and exploring results. Four different major display modes are already implemented: plain text view, XML view, graphical tree view and timeline view (see Figures 3 and 4). It should be noted that figures 1 to 4 do not represent the final look of the graphical query interface. The environment is still work in progress – its design will be finalised in the autumn of 2008. Rehm et al. (2008a) provide a detailed description of the corpus query interface and several related components such as the interaction between the XQuery engine and the ontology.

6. Concluding Remarks and Future Work
In this article, we presented requirements of a corpus query interface which have been compiled based on two sources: a survey among linguists that regularly consult corpora and also create corpora themselves and an analysis of existing applications for corpus querying. This approach turned out to be a suitable and effective way to accumulate a number of important and useful requirements for our own query interface. We consider it an additional advantage that users of established software will recognise some popular features in our interface and will not be confronted with entirely new paradigms and metaphors.

The survey and analysis presented here is associated with the project “Sustainability of Linguistic Data” which is still work in progress. We want to highlight some of the aspects that we plan to put into effect by the end of 2008. In addition to the ongoing corpus normalisation and metadata transformation work (Rehm et al., 2008b), most rele-
vant for the results of our survey is the continuous implementation of the metadata exploration interface and of the graphical visualisation and querying front-end (Rehm et al., 2008a). We plan to upgrade and enhance several aspects of the GUI. Next to a substantial design overhaul of the interface in order to improve its usability, we will integrate graphical query templates and saved searches that act like bookmarks in a web browser. For their representation we will use an XML-based format to store all necessary data in one place. Moreover, we will integrate functions for multi-layer querying as well as for the visualisation of multi-layer annotations, and we will finalise the ontology-based query expansion component. We plan to finish work on the GUI as well as on the whole platform by September.

Acknowledgments
The research presented in this paper was supported by a grant from Deutsche Forschungsgemeinschaft within the project Nachhaltigkeit linguistischer Daten. The authors would like to thank Hanan Bechara and Johannes Dellert (Tübingen University) for implementing significant parts of the user interface and Lucas Ogden for proofreading.

7. References