Usage of XSL Stylesheets for the annotation of the Sámi language corpora

Saara Huhmarniemi
University of Tromsø
saara.huhmarniemi@helsinki.fi

Sjur N. Moshagen
Norwegian Sámi Parliament
sjur.moshagen@samediggi.no

Trond Trosterud
University of Tromsø
trond.trosterud@hum.uit.no

Abstract

This paper describes an annotation system for Sámi language corpora, which consists of structured, running texts. The annotation of the texts is fully automatic, starting from the original documents in different formats. The texts are first extracted from the original documents preserving the original structural markup. The markup is enhanced by a document-specific XSLT script which contains document-specific formatting instructions. The overall maintenance is achieved by system-wide XSLT scripts.

1 Introduction

Corpus building for a specific language is considered to require much human effort and time. To overcome this difficulty, there is a recent development of applications for automatic corpus building using often the Web as a resource e.g. (Baroni and Bernardini eds., 2006; Sharoff, 2006). For minority languages, the resources for building a text corpus are often limited. Automatic tools for building corpus database specifically for the minority languages are developed e.g. by (Ghani et al., 2005; Scannell, 2004).

The requirement to have the corpus building process automatized as much as possible was also central in the Sámi language corpora project. However, the collection of texts is done in a ”traditional” manner: the files are gathered and classified manually. For North Sámi, there are texts available in electronic form which can be exploited in a corpus database, mainly administrative and newspaper texts. The small amount of those texts forced us to take into account a wide variety of sources and formats, and also to include texts that were of low technical quality. That introduced problems for the automatic processing of the texts. The solution to this problem was the document-specific processing instructions that were implemented in XSLT.

2 The Project

The corpus described here is the first structurally annotated text corpus for any Sámi language. The corpus database was developed in parallel with the spell checker and the syntactic analyzer projects for North and Lule Sámi[1]. The new texts became test material for these two applications as soon as they were added to the corpus database. The requirements for the markup were constantly being re-evaluated during the project. The infrastructure was designed flexible so that it would accomodate to the different needs of the two projects in different phases of the application development.

At the moment, the corpus database consists of almost 6 million words for North Sámi and some 240 000 for Lule Sámi. Even though the system was primarily designed for the Sámi languages, there are no strictly language-dependent sections in the system; it has already been tested with Norwegian and Finnish, among others.

One of the main applications of the text corpus database is the syntactically annotated and fully disambiguated corpus database for Sámi languages. The syntactic annotation is done automatically us-

[1] http://www.divvun.no/, http://giellatekno.uit.no/

Proceedings of the Linguistic Annotation Workshop, pages 45–48,
Prague, June 2007.

ing the tools developed in the syntactic analyzer project, but the process is out of the scope of this paper. There is also some parallel texts with Norwegian, and plans for extending parallel text corpora to different Sámi languages and Finnish and Swedish. The corpus database is freely available for research purposes. There will be a web-based corpus interface for the syntactically annotated corpus and a restricted access to the system for examining the corpus data directly.

3 XSLT and corpus maintenace

Flexibility and reusability are in general the design requirements of annotated text corpora. XML has become the standard annotation system in physical storage representation. XML Transformation Language (XSLT) (Clark ed., 1999) provides an easy data transformation between different formats and applications. XSLT is commonly used in the contemporary corpus development. The power of XSLT mainly comes from its sublanguage XPath (Clark and DeRose eds., 1999). XPath provides an access to the XML structure, elements, attributes and text through concise path expressions.

In the Sámi language corpora, XSLT is used in corpus establishment and maintenance. The raw structural format is produced by text extraction tools and coverted to a preliminary XML-format using XSLT. The markup is further enhanced by document specific information and a system-wide processing instruction, both implemented in XSLT.

4 The Sámi corpus database

4.1 Overall architecture

The corpus database is organized so that the original text resources, which are the documents in various formats (Word, PDF, HTML, text) form the source base. The text is extracted from the original documents using various freely available text extraction tools, such as *antiword* and *HTML Tidy*. They already provide a preliminary structural markup: antiword produces DocBook and HTML Tidy provides output in XHTML. There are XSLT scripts for converting the different preliminary formats the to an intermediate document format.

The intermediate format is further processed to the desired XML-format using XSLT-scripts. The result is the final XML-document with structural markup, see Fig. 1.

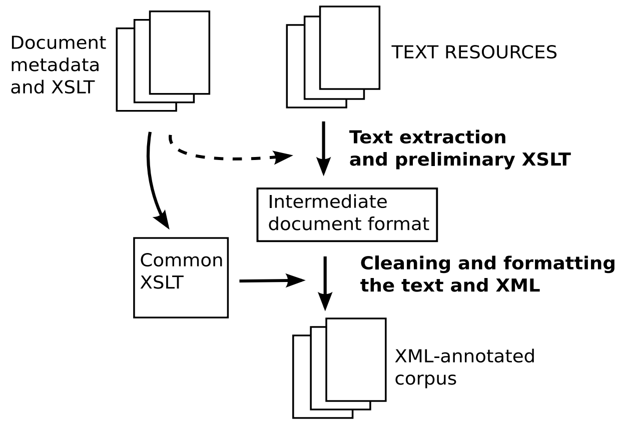

Figure 1: The overall architecture of the conversion process.

The conversion of a document always starts from the original file, which makes it possible to adapt for the latest versions of the text extraction tools and other tools used in the process as well as the changes in XML-markup.

The annotation process is fully automatic and can be rerun at will. Some documents may contain errors or formatting that are not taken into account by the automatic tools. On the other hand, the automatized annotation process does not allow manual correction of the texts, nor manual XML-markup. Those exceptions can be taken into account by document-specific processing instructions, which are implemented using XSLT. The script can be used for adding XML-annotation for specific parts of the document, fixing smaller errors in the document, or even to rescue a corrupted file that would be otherwise unusable. This is a useful feature when building a corpus for a minority language with diverse and often limited text resources.

4.2 XML-annotation

In the Sámi language corpora, markup of running text is simple, containing no more structural information than what is generally available in the original text. The body text can contain sections and paragraphs and each section can contain sections and paragraphs. There are four paragraph types: title, text, table and list. The paragraphs are classified whenever the information is available in the original document. Lists and especially tables contain incomplete sentences and in many cases numeric data. When conducting e.g. syntactic analysis, it might be better to leave tables and even lists or titles out, whereas for e.g. terminological work the tables are highly relevant. Tagging for paragraph type makes it possible to include or exclude the relevant paragraph types at will.

Inside a paragraph, there is a possibility to add emphasis markup and other span information, such as quotes. The sentence-level and word-level markup is not included in the text corpus. The markup is added when the text corpus is moved to the syntactically annotated corpus database.

The XML-annotation does not follow any standardized XML-format, but it is, in essence, a subset of the XCES (Ide et al., 2000) format. Furthermore, the system is designed so that changing the XML-annotation and moving to a standardized format is a straightforward process.

4.3 XSLT processing

Each original document in the corpus database is paired with an XSLT script. The document-specific XSLT script contains processing instructions that are applied to the document during the conversion from the preliminary document format to the final XML-format (see Fig. 1.). The XPath expressions are powerful tools for accessing portions of text in a document and modifying the XML-markup without editing the XML-file itself. The usage of the XPath expressions entails that the XML-structure of a document does not change, which poses some restrictions to the intermediate format of the document.

The XSLT script contains the document metadata and status information, among other relevant data. The document metadata is stored in variables in the document-specific XSLT script, and the system-wide XSLT scripts access these variables and convert them to the required format.

The system-wide XSLT script contains functions and templates that can be called from the document-specific XSLT script. There is for example a string-replacement function for correcting errors that are of technical origin, such as wrongly converted Sámi characters that were missed by the automatic detection of wrongly encoded characters.

Another example of a template that can be called from the document-specific XSLT script is the string-replacement, that can be used for marking spelling errors in the text. Due to the variety of conventions of writing Sámi, the texts tend to contain lot of strings that are classified as spelling errors. The errors disturb the testing of the analyzer, but are on the other hand interesting from the point of view of the spell checker project. When a spelling error is discovered in the text, the erroneous strings and their corrections are added to the document-specific metafile from where they are picked by the conversion process. The errors are thus preserved in the XML-format but with a correction which can be used instead of the erroneous string. This is achieved by a special markup:

```
<error correct="text">tetx</error>
```

In this way the original documents stay intact and the information is preserved also when the file is reconverted. If the error is not just a single word but involves more context, it is possible to add the context to the error string.

In addition, the document-specific XSLT script contains variables that may be used already in the text extraction phase. An example would be the text alignment information of a pdf-file.

4.4 Language identification

Most documents in the Sámi corpus database contain sections of text that are not in the document's main language. Those sections are marked at paragraph level, using the attribute *xml:lang*.

The language identification is done using the *TextCat* tool (van Noord, 1997). Since the different Sámi languages and the close relative Finnish resemble each other significantly (the same is true for the Scandinavian languages), the search space was reduced at the document level. The information of the document languages was stored to the document-specific XSLT script.

Since the Sámi texts contain lot of quotations from other languages, especially from the majority language (Norwegian, Swedish or Finnish), the quoted text fragments are analysed separately using *TextCat* and marked with a corresponding *xml:lang* attribute. For example:

```
<span type="quote" xml:lang="nob">
"Arbeidet med fylkesplanene"
</span>
(bargu fylkkaplánaiguin).
```

When a sentence that contains a quotation in a foreign language is given to the syntactic analyzer, the quotation can be considered as a syntactic unit and that way carried through the analysis.

4.5 Other processing

Character set conversion may be a central task when a corpus is built for minority languages, due to a large repertoire of non-standardized 8-bit character sets, e.g. (McEnery et al., 2000; Trosterud, 1996). In the Sámi corpus database, the text extraction tools often produced wrongly-utf8 -encoded output, due to erroneous codepage IDs and font specifications. There is a specific module for guessing the documents' original code-page, and for fixing errouneous utf8-conversion.

There are a couple of other scripts that are applied to the final XML-documents. For example, real hyphenation marks in the document are preserved for testing of the hyphenator. The hyphen-tags are marked automatically, taking into account some language specific cues and information of e.g list context.

5 Conclusion

The system is flexible and reusable since the central XSLT processing allows for changes in the XML-structure as well as the introduction of new structural information. The intermediate XML-formats which are produced by the text extraction tools are straightforward to convert to a format that conforms to the project's DTD using XSLT processing. Instead of trying to predict the future uses of the corpus database in the beginning of the project, the infrastructure was set up so that it evolves throughout the project.

The main problem in the heavy usage of XSLT is that the syntax of the XSLT is quite restricted although XSLT/XPath 2 brings some improvements. The lack of regular expressions is one of the restrictions, so some of the string-replacement functions had to be implemented by other means. In the future, these could probably be replaced with XPath 2 functions.

Fully-automated, XSL/XML-based conversion has made it possible to build a corpus of decent size for small languages. After the initial infrastructure is created, adding new documents does not require much resources. The system does not involve any strictly language-dependent processing, so it is portable to other languages. The result is a clean, classified and XML-annotated corpus which can be used in research and different language technology applications.

References

Marco Baroni and Silvia Bernardini (eds.). 2006. *Wacky! Working papers on the Web as Corpus*. http://wacky.sslmit.unibo.it/.

James Clark (ed.). 1999. *XSL Transformations (XSLT) 1.0*. W3C Recommendation. http://www.w3.org/TR/xslt.

James Clark and Steve DeRose (eds.). 1999. *XML Path Language (XPath) 1.0*. W3C Recommendation. http://www.w3.org/TR/xpath.

Rayid Ghani, Rosie Jones, and Dunja Mladenic. 2005. *Building Minority Language Corpora by Learning to Generate Web Search Queries*. *Knowledge and Information Systems*, 7(1):56–83.

Nancy Ide, Patrice Bonhomme, and Laurent Romary. 2000. *XCES: An XML-based Encoding Standard for Linguistic Corpora*. *Proceedings of the Second Language Resources and Evaluation Conference (LREC)* 825–830.

Anthony McEnery, Paul Baker, Rob Gaizauskas, Hamish Cunningham. 2000 *EMILLE: Building a corpus of South Asian languages*. *Vivek, A Quarterly in. Artificial Intelligence*, 13(3): 23–32.

Kevin P. Scannell. 2004. *Corpus Building for Minority Languages.* http://borel.slu.edu/crubadan/.

Serge Sharoff. 2006. *Open-source corpora: using the net to fish for linguistic data International Journal of Corpus Linguistics*, 11(4): 435-462.

Trond Trosterud. 1996 *Funny characters on the net. How information technology may (or may not) do support minority languages*. *Arbete människa miljö & Nordisk Ergonomi*, 3:114–125.

Gertjan van Noord. 1997. *TextCat Language Guesser*. http://www.let.rug.nl/∼vannoord/TextCat/.