# Corpus Construction Tools

Radovan Garabík
Jazykovedný ústav Ľ. Štúra SAV
813 64 Bratislava, Slovakia
korpus@korpus.juls.savba.sk, http://korpus.juls.savba.sk

## Abstract

Современное развитие вычислительной техники позволяет нам принять участие в раньше невозможных направлениях научного исследования естественного языка. Основной, необходимой базой данных являются корпусы языков, в том числе и репрезентативные большие (национальные) корпусы. Уже широко доступны общие программные средства позволяющее эффективно обрабатывать большие количества текстов, как и средства поиска в корпусах. Всё-таки, создание корпуса с большим количеством данных требует определённый план организации обработки текстов, вместе с структурой программного обеспечения. В докладе представлена общая система позволяющая быстро применить специфические черты обработки данных конкретного языка. Обсуждены необходимые аспекты национального корпуса, как с лингвистической, так и с компьютерной точек зрения. Система использует преимущественно современный объектно-ориентированный язык программирования Python, имеющий превосходные возможности обработки текстовых данных. Разметка текста состоит из двух частей, из лингвистической (внутренней) разметки текста, которая является внутренним свойством лингвистических единиц (слов) в тексте, и из общих данных о документах (метатекстовая, внешняя разметка). Внутренняя разметка текста входит прямо в формат обработанных текстов, в результате использования существующих стандартов репрезентации текстовых данных, как XML (XCES). Внешняя разметка сохраняется в простых текстовых файлах, с реляционной базой данных построенной над этой структурой.

## Introduction

There exists a reasonably extensive literature concerning principles of corpora structure and end-user interaction [1, 2, 3, 4 and many others]. However, technical details of corpora construction are usually left out as uninteresting or too closely tied up with a specific corpus, and therefore not applicable in general. As with every big project, creating and maintaining an extensive (i.e. “national”) corpus of written language requires careful thought up design of data structure and of data manipulation. Consequently, each newly created big corpus ends up reinventing the wheel and implementing the data workflow and manipulation from the scratch. During the Slovak National Corpus construction, we did basically the same thing, but we tried to make our design general and clean, in order to serve as an inspiration for eventual other yet to be created big corpora. This does *not* include end-user information searching by a corpus manager – there are several (thought not many)

different corpus managers available, that could be plugged into our system.

## Premises of big corpora design

First, we have to explain what we mean by a “big” corpus. The first criterion is the size (number of words). Second criterion is a richness of corpus annotation. Obviously, size and annotations by themselves are not a requirement for a complex and careful design, because if the goal is just to collect as much texts as possible (e.g. from the Internet) and make them available in corpus format without the need of any additional non-trivial markup and annotation, we do not need any special data structures or construction plans. Conversely, very carefully annotated corpus of documents, each of them of a great importance (typical example can be a historical corpus of texts in a rare language) has different goals and the data manipulation, conversion and annotation is best left completely to be done by human researchers.

Third criterion is an extensibility of the corpus. For one-time, create-and-forget corpora, even if big in size and with a rich annotation, ad-hoc written tools and creation procedures are adequate.

Therefore, by a “big” corpus we understand a corpus with size exceeding comfortable dimensions for easy human and computer processing (millions of words), with a rich, non-trivial annotation (bibliographic information and linguistic annotation – e.g. lemmatization and grammar tagging) , and with the need to accommodate new texts being added into the corpus.

Since the incoming data can be in many different formats, and because as the final result of text processing we should obtain full linguistic analysis, it is not feasible to run the whole conversion process for all the different input formats. This leads us to a need to deploy a common intermediate format for text storage. This way, we write conversion procedures turning many different incoming text formats into this intermediate format, which is then further analysed. As the XML format gained popularity and became de facto standard for text storage, the Text Encoding Initiative turned from SGML formats to XML, and has modified SGML based CES

format into an XML based XCES [5]. Using XCES format as an intermediate storage has an advantage in being compatible with several other big corpora, and a reasonable high availability of many different XML processing tools.

## Requirements

The authors' programming language of choice is Python [6]. However, the presented system is highly modular, and in order to use it as provided, no special understanding of Python is required.

System needs following libraries and programs installed before:

- *Python*, at least version 2.3, with standard library
- *ElementTree* XML parsing library
- to parse Microsoft Word (.doc) files, *antiword* [7] version at least 0.35
- to parse Rich Text Format (.rtf) files, *rtf2xml* and *transform* [8]
- to search the metadata (recommended), an SQL database is required. Both MySQL and SQLite are supported.
- UTF-8 capable terminal and text editor

Recommended setup consists of a central UNIX server with k3t and MySQL server installed. In order to work with the system, the users (annotators) remotely log into the server. The workstation's operating system is irrelevant, as long as it is able to provide a reliable UTF-8 terminal and ssh or telnet connection to the server[1].

The system is cluster friendly – central server filesystem can be exported to cluster nodes and data conversion can be distributed throughout the cluster.

## Corpus structure

Data are kept in four levels called them Archive, Bank, Corpusoid and Data.

In Archive, original input texts are kept , without any conversion or modification.

In Bank, texts are converted into a carefully chosen subset of the XCES format. One document in the Archive can be divided into several texts in the bank. Apart from

1 part of operating system in MacOsX and most GNU/Linux distributions; provided by many different terminal emulators (e.g. putty) for Microsoft Windows .

the conversion, no other analysis is done here.

Corpusoid corresponds one-to-one to the Bank, with additional linguistic information (such as lemmatization and morphological tagging).

In Data, texts are converted into format suitable for the corpus manager used (typically vertical text).

Documents in the Archive, Bank, Corpusoid and Data are are organized in a tree directory structure. There is a fixed number of subdirectory levels for Archive, and another (bigger) number of levels for Bank, Corpusoid and Data. The exact meaning of the levels is not rigidly specified, but is left to a corpus designer to reflect a logical structure of the corpus here. For example, the structure deployed in the Slovak National Corpus has 4 levels for the Archive, corresponding to the year, month and day of data acquisition, and the fourth level number enumerates the documents acquired in one day. Bank contains one additional level, used for numbering multiple document in the Bank coming from the same document in the Archive.

## File formats conversions

Under typical circumstances, input texts coming into the corpus are in many different formats.

Texts produced by different authors often differ in subtle details, e.g. in the usage of typographic quotes, in using different conventions for various types of hyphens and quotes, in header and section formatting. There is also the question of many varieties of html files, acquired from the Internet and forming a substantial part of a corpus.

While all these differences and formats could be easily dealt with during manual conversion, it is desirable to have an automated conversion for as much data as possible. This is exceptionally useful in case of a periodical publications, where we can have many thousands of different text units contained in one archive document (e.g. newspaper). Archive conversion tools look for a subdirectory called *.convert*, for each archive unit, and execute an executable called *convert* contained in this

directory. This executable accepts two parameters, the first is archive Id of the document, and the second is the full path to the archive directory. The script is responsible for converting all the desired texts belonging to this archive ID into the Bank. It can be written in any programming language, but in case of using Python, there is a broad range of helper functions for different conversions available. This script is also responsible for eventual “levelling” of all typographical differences for different archive documents, and for creating correct (possibly empty, with the exception of Id and Sourceid key) Bank annotation.

## Tokenization and segmentation

The system contains a tokenization module, and  amodule for grouping the tokens into bigger structures (sentences). A sample rule-based tokenization module suitable (with minor modifications) for most alphabetic languages is provided, as well as a sample segmentation module suitable for the Slovak language.

## User interface

Nowadays, user interaction with a computer is carried out primarily via Graphical User Interface (GUI), even if Command Line Interface (CLI) has still many supporters, mostly among technically more experienced users, especially because of its expressive powers and usage speed unmatched by GUI.

We found out that teaching basics of GUI interaction to people with little or no computer training is extremely time consuming and difficult, since using GUI requires a lot of assumed user knowledge still not universally widespread. On the other end, it is relatively easy to teach the users a few necessary commands, and as our experience shows that even complete computer novices can very quickly become proficient in working with the system. Therefore, presented system is (rather unusually for modern times) heavily CLI oriented. Adding an optional GUI layer on top of it is rather straightforward and easy, but it was not necessary for our purposes.

## Drawbacks and limitations

First, perhaps the most obvious drawback lies in the use of XML formats, as opposed to plain unmarked texts. Parsing XML is usually a CPU intensive task, and when combined with using an interpreted programming language and several separated stages of text processing, it leads to more computer time needed. However, it is compensated by a versatility and clarity of the whole system.

Another design insufficiency is a strict separation of input data path into exactly four levels. Sometimes, it means that too much of logical processing is going to be crammed into the third level (corpusoid), since, by design, all linguistic processing should be carried out here.

## Software availability and status

The system described herein is released under the GNU General Public License version 2 and is freely downloadable from the Slovak National Corpus WWW page (http://korpus.juls.savba.sk/download/). At the time of writing, the status of the software would be best described as "late alpha" – it works for the authors, but it is still rather complicated to install and use, and there are numerous small bugs and imperfections. Also the documentation is very scarce and unfinished, but the system is provided to the general public in hope that it could be at least partially useful.

## Literature

1. Jarošová, A.: *Korpus textov slovenského jazyka.* In: Slovenská reč 2 (1993) 89–95
2. Šimková, M.: *Počítačové spracovanie prirodzeného jazyka a Slovenský národný korpus*. Budmerice: Počítačová podpora prekladu (2003)
3. Zakharov, V.:Russian Corpus of the 19th Century. In: Proceedings of the 6th International Conference TSD 2003. České Budějovice, Czech Republic (2003)
4. Przepiórkowski, A.: *The IPI PAN Corpus preliminary version.* Instytut Podstaw Informatyki PAN, Warsaw, Poland (2004)
5. Ide, N., Bonhome, P., Romary, L.: *XCES: An XML-based Encoding Standard for Linguistic Corpora*. In: Proceedings of the Second International Language

Resources and Evaluation conference. Paris: European Language Resources Association (2000)
6. http://www.python.org
7. http://www.winfield.demon.nl/index.html
8. http://rtf2xml.sourceforge.net
9. Brants, T.: TNT – Statistical Part-of-Speech Tagging. http://www.coli.uni-saarland.de/~thorsten/tnt/
10. http://www.textforge.cz

Príspevok odznel na konferencii MegaLing'05, Meganom, Ukrajina, 27. 6 – 2. 7 2005 a bol publikovaný v zborníku.
Abstrakt bol publikovaný v zborníku abstraktov.