**HEIDEN — Progressive**

<1 table, 1 figure>

**Progressive Philology with TXM: From ‘Raw Text’ to ‘TEI Encoded Text’ Analysis and Mining**

Heiden, S.

A typical text analysis workflow combines the following two stages:

• Preparing data. Philological methodology establishes the base text, critical apparatus, and some analytical information like text metadata (author, date, domain, etc.) to be able to compare domains, authors, etc. with respect to the text’s content (word forms or word lemma frequency, n-gram frequency, etc.).

• Processing. Analytic methodology provides the framework to analyze a corpus or sub-corpus of texts with the help of word patterns search and display (concordances) or count (frequency lists) and comparison tools (factorial analysis, clustering).

It is convenient to characterize the relation between the data and the processing stages as contractual. That contract is based on the encoding conventions of the digital representation of texts interpreted by the software to manipulate and display them through interfaces.

Simple conventions (like Unicode character encoding of raw text) can correspond to file formats directly interpreted by the software; more elaborate ones (like XML-TEI encoded text) correspond to guidelines that need tuning to be processable by software. This article presents a text analysis workflow implemented in the TXM analysis software in which the data/processing philological contract complexity can be progressively adapted to the needs of the user, balancing costs of encoding with interests in processing and analysis. It will also introduce the delegation model to call external tools (like ‘natural language processing’ [NLP] software as lemmatizers) to automatically enrich encoded texts on the fly while importing them into the platform for analysis.

**The TXM Analysis Software**

The open-source GPL-licensed TXM software implements an analytic methodology articulating those stages. It provides an import framework that allows reading of corpus sources at various levels of encoding and a classic toolbox for text analysis and mining, composed of a versatile and efficient full text search engine, text reading and browsing, sub-corpus and partition building, co-occurrence analysis, factorial analysis, and clustering. It is available as a desktop application for Windows, Mac, or Linux, as well as a web portal software for a server accessed through a web browser (Heiden, 2010). The TXM platform can be downloaded for free at http://sf.net/projects/txm.

**Importing Corpora into TXM at Progressive Levels of Encoding**

Starting from raw text corpora, the first level of contract is simple and has a low encoding cost barrier: texts may have properties (called metadata) that can be used to compare them or their properties, and words are defined by their character constituents. The text format provides all information needed explicitly to the software.

To analyze a corpus at such a level of encoding, TXM provides the Unicode ‘TXT+CSV’ import module. The CSV in the name expresses the possibility to associate metadata to each text through an external CSV table file. That table encodes the metadata of each text on a separate line, as in the following excerpt of the Brown corpus metadata table defining the ‘type’ and ‘reference’ metadata (Francis and Kucera, 1964):1

|  |  |  |
| --- | --- | --- |
| **id** | **type** | **reference** |
| a01 | PRESS: REPORTAGE | Sample A01 from The Atlanta Constitution |
| a02 | PRESS: REPORTAGE | Sample A02 from The Dallas Morning News, February 17, 1961, sec. 1 |
| a03 | PRESS: REPORTAGE | Sample A03 from Chicago Daily Tribune |
| a04 | PRESS: REPORTAGE | Sample A04 from The Christian Science Monitor, May 11, 1961, p. 1 |
| a05 | PRESS: REPORTAGE | Sample A05 from The Providence Journal |

Table 1. Brown corpus metadata (the ‘id’ column encodes text identifiers, ‘type’ the genre, and ‘reference’ the source).

When it comes to TEI encoded texts, the contract is more complex and fuzzy, especially because the TEI provides guidelines for encoding but not as prescriptive as a format can do. One can typically use several different ways to encode a particular text component. This is needed for scholars to adopt the technology: it provides ways to encode texts we haven’t read yet, for which we haven’t established the base text or the critical apparatus yet. For a given project, the TEI encoding practice should be much more specific and documented because scholars need to share the same conventions to cross-validate their philological work. In such a context, it is possible to tune the software based on the conventions documentation to import the specific TEI sources and build the necessary corpus model to process it.

To analyze a corpus at such a level of encoding, TXM provides pre-tuned TEI import modules for each documented TEI practice—for example, the ‘XML-TEI BFM’ import module for texts encoded according to the BFM corpus TEI encoding practice (Heiden et al., 2010).

To analyze a corpus encoded with an unknown TEI practice, TXM provides the generic ‘XML/w+CSV’ import module. That module provides a simple way to adapt any XML or TEI encoding through an XSLT transformation stylesheet into the ‘XML-TEI TXM’ format designed for the TXM corpus model.

TXM is available with a library of several XML and TEI adaptation stylesheets: https://sourceforge.net/projects/txm/files/library/xsl.

The ‘XML/w+CSV’ import module is part of the progressive TXM import framework, which is built on a concentric format model (every format is based on the format of lower-outer level): Unicode TXT < XML < TEI < XML-TEI TXM:

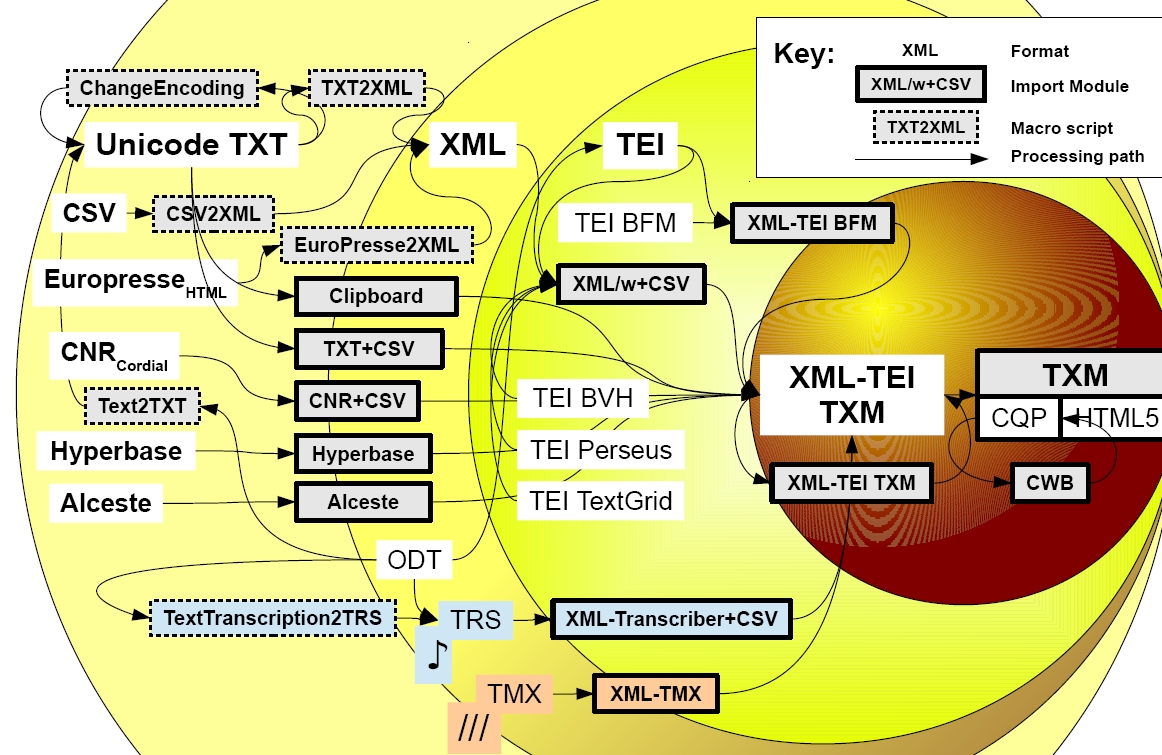


Figure 1. TXM progressive import framework workflow.

In that framework, import modules can take a corpus at any level of encoding passing through the final TXM pivot ‘XML-TEI TXM’ format before compilation by the software (search engine indexes, text edition, etc.).

The corpus model of TXM is composed of the following:

• Each corpus consists of a set of **texts** with properties called ‘metadata’ (author, title, date, genre . . .).

• Each text is composed of nested optional internal **structures** that have properties.

• Each text is composed of a sequence of **words** that can have properties. Words can be embedded in any textual structure.

Each text has an HTML edition built during the import process. Some **textual planes** can be built on demand, such as separating comments or notes from content body, etc.

For each import encoding level, some corpus model elements can be built, or not:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **TXT** | **XML** | **TEI** |
| **Text units** | files | files | files |
| **Metadata** | CSV | CSV | teiHeader |
| **Structures** | n/a | all | TEI specific |
| **Words** | raw | <w>? | <w>? |
| **Textual planes** | n/a | front XSL | TEI specific |

For each import level, a file corresponds to a text unit. TXT level cannot provide either internal structures encoding for texts or planes and tokenizes words from their ‘raw’ character constituents. XML level provides TXT level processing and can optionally encode some or all words with a <w> tag; all other tags are interpreted as text structures, and a front XSL can filter out some elements on the fly. TEI level provides XML level processing, text metadata are taken from the TEI header, and every tag has specific semantics related to structures or planes.

Within such an import framework, the scholar can adapt the encoding effort to the level best suited for the elements she wants to be able to manipulate in the analysis software. A typical progression is to start with raw text (TXT) and progressively encode more information (sometimes based on findings made with the analysis tools). When entering the XML level, the first elements encoded are very often some specific structures (e.g., sentence, paragraph, or section) or words (e.g., compounds or entities) and page breaks to control text edition pagination with respect to the sources and concordance references (e.g., page number).

**NLP Annotation at Word Level**

As any level of encoding supposes the processing of words (implicitly or explicitly encoded), TXM integrates the possibility to call external NLP tools working on words to automatically enrich the sources through a delegation model. In that model, tools and their parameters are first declared once. Then, for each import module, they can be automatically called as many times as needed on a representation of the texts compatible with their processing (texts are converted to the format needed by each tool), and the result of their processing is reinjected back into the sources. As an example, TXM automatically annotates words with their lemma and morphosyntactic category with the TreeTagger software. In that model, the quality of the automatic encoding is delegated to external tools.

In the case of XML type sources, if a word is already encoded with some properties (typically by using a <w> element), the NLP annotation is just added to the initial encoding.

Adding systematic word level annotation can be difficult to keep compatible with original XML encoding. This is why TXM provides stand-off and inline ways to store word annotations into a TXM-specific TEI extension scheme called ‘XML-TEI TXM’. That representation can be exported to other tools or re-imported in any TXM.

**Conclusion**

We presented the TXM progressive import framework allowing the scholar to choose the right level—and effort—of encoding needed for the analysis of its sources. TEI level encoding is not mandatory to use the tool, simple raw text can be analyzed, and XML encoded sources provide a lot of useful services for analysis, but TEI provides the best reference target level of encoding as a language to establish a contract between philological and analytic work.

**Note**

1. The sample BROWN corpus can be downloaded from the TXM software website, http://sourceforge.net/projects/txm/files/corpora/brown.

**References**

**Francis, W. N. and Kucera, H.** (1964). A Standard Corpus of Present-Day Edited American English, for Use with Digital Computers. Department of Linguistics, Brown University, Providence, RI. Revised, 1971; revised and amplified, 1979. See also https://sourceforge.net/projects/txm/files/corpora/brown.

**Heiden, S.** (2010). The TXM Platform: Building Open-Source Textual Analysis Software Compatible with the TEI Encoding Scheme. In Otoguro, R., Ishikawa, K., Umemoto, H., Yoshimoto, K. and Harada, Y. (eds), *24th Pacific Asia Conference on Language, Information and Computation*, November 2010, Sendai, Japan, Institute for Digital Enhancement of Cognitive Development, Waseda University, pp. 389–98.

**Heiden, S., Guillot, C., Lavrentiev, A. and Bertrand, L.** (2010). Manuel d’encodage BFM / XML-TEI, Version 4.0, BFM–Base de Français Médiéval [En ligne]. Lyon: ENS de Lyon, Laboratoire ICAR, http://bfm.ens-lyon.fr/IMG/pdf/Manuel\_Encodage\_TEI.pdf.