The TXM Platform: Building Open-Source Textual Analysis Software Compatible with the TEI Encoding Scheme*

Serge Heiden

ICAR Laboratory, University of Lyon,
15 parvis René Descartes - ENS de Lyon, 69342 Lyon, France
slh@ens-lyon.fr

Abstract. This paper describes the rationale and design of an XML-TEI encoded corpora compatible analysis platform for text mining called TXM.

The design of this platform is based on a synthesis of the best available algorithms in existing textometry software. It also relies on identifying the most relevant open-source technologies for processing textual resources encoded in XML and Unicode, for efficient full-text search on annotated corpora and for statistical data analysis.

The architecture is based on a Java toolbox articulating a full-text search engine component with a statistical computing environment and with an original import environment able to process a large variety of data sources, including XML-TEI, and to apply embedded NLP tools to them.

The platform is distributed as an open-source Eclipse project for developers and in the form of two demonstrator applications for end users: a standard application to install on a workstation and an online web application framework.

Keywords: xml-tei corpora, search engine, statistical analysis, textometry, open-source.

1 Introduction

Textometry is a textual data analysis methodology born in France in the 80s in relation with the data analysis framework designed by (Benzecri et al., 1973a,b). A first synthesis of the methodology can be found in (Lebart et al, 1997). It somewhat differentiates from text mining by the fact that it tries to always combine various statistical analysis techniques, like factorial correspondence analysis or hierarchical ascendant classification, with full-text search techniques like kwic concordances to be able to always get back to the precise original editorial context of any textual event participating to the analysis. It tries to always relate golden nuggets found in corpora to the context of their original data source. Thus it involves more an interaction with corpora than a distillation process of them.

* The work reported in this paper was supported by French ANR grant #ANR-06-CORP-029.

In 2007 a project joining textometry teams from four French universities[1] started a four year project to build an original open-source software framework for a new generation of application software for textometry[2].

The main goals of the project are:

- synthesize the best available algorithms for textometry analysis with the most up to date annotated corpora models
- implement them with the best available and sustained open-source components
- be compatible with Unicode (2006), XML and TEI (2008) standard encoded data sources
- be able to efficiently analyze corpora of several hundred million tagged words
- be compatible with usual NLP software (like taggers and lemmatizers)
- distribute a framework toolbox for developers to build new applications
- demonstrate a Windows and Linux prototype application for the end researcher users from the humanities and social sciences
- demonstrate an equivalent web based client/server prototype application
- package the applications for easy install and deployment
- document the toolbox framework, the applications and the development process publicly to build a community driven development network

This paper will first discuss the specification and the conception model of the new open-source textometry platform. It will then present the data workflow chosen to import corpora into the platform. After the presentation of the software design and its architecture it will detail the open-source software components chosen to implement the first alpha version of the core toolbox and the two first prototype applications: one local rich client and one web based.

2 Specification and conception of the platform

The textometry functionalities and data model of the platform have been designed by a synthesis work on a representative selection of existing software made by members of the project. The software analyzed were:

- Astartex, (J-M. Viprey)
- DTM (L. Lebart)
- Hyperbase (E. Brunet)
- Lexico (A. Salem)
- Sato (F. Daoust)
- Weblex (S. Heiden)
- Xaira (L. Burnard)

2.1 Functional specifications

The project organized the functionalities in the following categories (Pincemin et al., 2010):

- Data
    - Initialization: the corpus import process, the analysis parameters
    - Builders: the capacity to add new properties to the corpus nodes and the ability to select any group of them
    - Current view: the structure (parts for contrastive analysis, parallelism and alignment), the node properties used, location: the way to define bibliographic references (for concordances for example), the current leaf nodes of the corpus

[1] Actually the project involves also colleagues from the universities of Oxford (UK) and UQAM (CA).
[2] See `http://textometrie.ens-lyon.fr`

(like lexical units) and the focus (build from a selection of nodes and some of their property values)

- Reading
  - Edition of the texts composing the corpus
  - Concordances
  - Contexts extraction
- Synthesis
  - Analytical lists
    - Vocabulary hierarchical index
    - Overall statistics
  - Positions
    - Progression of textual events along an internal textual axis (like bursts analysis)
    - Distribution of events in different parts (like specificity analysis)
    - Development along an external axis (like chronological analysis)
  - Associations
    - Sequence of units : recurring sequence of strings
    - Cooccurrency of units : attraction between textual events
    - Analogy (like factorial analysis)
    - Correlation between property values along a textual axis
- Analysis
  - Visualizations : tables, graphics…
  - Organization of results : sorting, grouping, synoptic visualization
  - Annotation : bookmarks and node editing
  - Report : Journal of the analysis, report editing

Those categories not only synthesize functionalities available in previous software but also add several innovative ones and build a new comprehensive set. For example, the analytical tools have access to a new enriched view of corpus elements. Researchers are thus able to explore corpora with the same set of analysis functions applied to every lexical level made available by NLP preprocessing (graphical forms, pos tags, lemmas, etc. and combinations of them).

2.2 Conceptual model

The project designed a first simple conceptual and operational model to implement the previous functionalities (Heiden et al., 2010).
The main concepts involved in the model are:

- a root corpus and several sub-corpora
- an annotation hierarchy
- any number of typed properties on terminal and non terminal nodes of the hierarchy
- a search engine for :
  - focus expression (the result set of the search engine is a list of sequences of contiguous tokens)
  - subcorpora construction
  - partitions construction (for contrastive analysis)
- the node properties are used to construct constraints on focus and to construct the nature of the events analyzed (searched for or counted for in statistics)

In that model, the search engine is a key innovative component to access corpora at any granularity level starting from the lexical units level. For example, one can directly select the sub-corpus of all the words which are verbs in a particular tense to compare them to another sub-corpus of all the verbs, or one can efficiently select all the lemmas said by a particular

speaker, provided that a structural unit encloses all the corresponding words, to contrast them to the corpus of all the lemmas of words said by another speaker.

2.3 Data workflow

Because the conceptual model is by nature experimental and because the software components used in the platform are interchangeable through a plug-in mechanism (for example we plan to be able to use the TigerSearch engine in addition to the IMS CWB engine), we put most of the effort in designing a flexible software import environment based on a scripting programming language instead of a specific fixed input data format for corpora. That environment manages four different data format perimeters:

- TXT: any data less than or differently structured as XML (CSV, etc.)
- XML: any data encoding information in XML with Unicode characters
- TEI-TXM: any data encoded in a specific pivot XML-TEI extension format
- IDX: the platform dependant internal representation of data

The XML perimeter includes an original XML-TEI import module. That module is responsible for interpreting corpora encoded in the TEI international standard[3] (TEI Consortium, 2008). This is a key contribution of the TXM platform to the search and analysis of sustainable and interoperable textual resources.

At the moment, the TEI-TXM schema expands the TEI P5 schema only by:

- inserting a `<txm:form>` element inside the `<tei:w>` element for the graphical representation of words
- inserting any number of `<tei:interp>` elements inside the `<tei:w>` element for any annotation on words (for inline encoding)
- inserting a `<txm:application>` element inside the `<tei:appInfo>` with various sub-elements encoding the history of calls and parameters of tools which have been applied on data sources during the import process (like NLP tools)

The import environment also implements a TEI-TXM offline (standoff) schema using the TEI P5 `<linkGroup>` and `<link>` elements in standalone TEI text files. Informations are joined through the `ID` attributes of the `<w>` elements inside TEI texts. A facility is given to transcode informations offline or to inline them back on demand, in the same spirit as the ANCTool. We didn't use the GATE or UIMA strategy to encode annotations, and especially tokenization and word tagging, because of the TEI requirement. In the TEI representation of texts, the surface of the text can be multiple (for example through the parallel `<choice>` element) and can change for philological reasons during a project. For example, when medievists decide to change the way they interpret some signs as characters in manuscripts a whole corpus can change some characters in the surface of several texts. It is thus difficult to always rely on an hypothetic character level to anchor annotations. In that context we then decided to let tools and persons interchange information at the lexical level only through the `<w>` element independently of what elements contains them or are contained by them (especially other `<w>` elements) and of the fact that they could be repeated or be tokenized in different ways.

To deal with corpora with no pre-tokenization encoding, we developed an original XML-TEI aware tokenizer. That tokenizer implements some of the design of the tokenizer developed by Alexei Lavrentiev for the TEI P5 encoded Old Medieval French Database. That component uses Unicode character classes, TEI element classes and some contextual analysis to identify in a text:

- what is above the sentence segmentation level
- where are the sentence boundaries
- what is above the word segmentation level

[3] Specialized for the BFM corpora.

- where are the word boundaries
- what is lower than the word level

and tries to encode its results in a TEI compatible way with the pre-existing TEI encoding. That tokenizer is compatible with all the levels of the different data format perimeters.

As an example of typical TEI-TXM encoding that the TXM platform can manage, we give an excerpt of a medieval French text where the word “Gormund” is written with the “un” letters missing in the original manuscript.
The scientific editor transcribes the text as follows:

> sa hanste brise par asteles.
> E Gorm[un]d ad l'espee traite,
> si l'ad feru sur le heaume :
> la teste en fist voler a destre,
>
> *Gormont et Isembart*, Champion, Paris, 1931, p. 4

Then the text is encoded in XML-TEI as follows for input to TXM :

```
<lb n="52"/>sa hanste brise par asteles.
<lb n="53"/>E Gorm<supplied rend="crochets">un</supplied>d ad l'espee traite,
<lb n="54"/>si l'ad feru sur le heaume :
<lb n="55"/>la teste en fist voler a destre,<pb n="6"/>
```

The TEI-TXM result of the TXM tokenizer is as follows:

```
<lb n="53"/>
<s n="17" id="s_17">
    <w id="w_360">
        <txm:form>E</txm:form>
        <interp resp="#txm" type="#ttpos">CONcoo</interp>
        <interp resp="#txm" type="#ttlemma">-</interp>
    </w>
    <w id="w_361">
        <txm:form>Gorm<supplied rend="crochets">un</supplied>d</txm:form>
        <interp resp="#txm" type="#ttpos">NOMpro</interp>
        <interp resp="#txm" type="#ttlemma">Gormund</interp>
    </w>
    <w id="w_362">
        <txm:form>ad</txm:form>
        <interp resp="#txm" type="#ttpos">VERcjg</interp>
        <interp resp="#txm" type="#ttlemma">-</interp>
    </w>
    <w id="w_363">
        <txm:form>l&apos;</txm:form>
        <interp resp="#txm" type="#ttpos">DETdef</interp>
        <interp resp="#txm" type="#ttlemma">-</interp>
    </w>
```

In that example, each word has been tokenized and enclosed in a `<tei:w>` element[4] with an ID number. The original graphical form is enclosed in the `<txm:form>` element. Each word has been annotated by a part of speech and possibly by a lemma, each enclosed in a `<tei:interp>` element.

[4] “tei” has been declared as the default namespace in this example.

3 Software design

3.1 Openness and standards

An important requirement of the project is for the platform to be open-source. This imposes us explicit up to date documentation rules: a Javadoc is generated regularly from the sources, a wiki [5] is dedicated to the auto-documented practices of developers : coding practice, IDE deployment… as are mailing lists, bug trackers and a roadmap site tracking all the activities with the software project milestones and tickets. But open-source gives us the opportunity to use open-source software components that we wouldn't have the resources to develop ourselves in the project.
For sake of sustainability, we tried to only use technologies related to accepted standards. Because the Java Community Process (JCP) is an open and robust standardization context, we choose Java as our base system programming language. The toolbox is designed following the OSGi component design standard rules implemented in the Rich Client Platform (RCP) framework which we found in the Eclipse IDE. The RCP framework gives us the environment to build the RCP GUI based prototype application for all target architectures (Windows, Linux and Mac OS X). For the web based prototype application, using the same Java toolbox as the RCP application, we use the Grails framework based on the Spring standard and producing J2EE compliant web applications and good Ajax integration.

3.2 Architecture

The base component of the platform is the toolbox, which is the core of the prototype applications. The toolbox exposes itself to the applications through an Application Programming Interface (API) offering five groups of services giving access to:

- the textual database: for the management of corpus, subcorpus, partition, focus...
- the statistical environment: for low level statistical models
- high level functionalities of textometry: factorial analysis, specificity analysis, kwic concordances, page edition…
- the import environment
- a scripting language

The toolbox communicates with the textual database and the statistics environment by sockets. This offers the possibility of distributed deployment of the platform on several systems in high end production settings.

The import environment relies heavily on the scripting language component. This gives the possibility to the end user to extend himself that environment. To integrate seamlessly external NLP tools, we designed a command line tool Java Process API wrapper generator based on an XML description of the Command Line Interface (CLI) of the tool to interface to the platform. The description of the CLI of TreeTagger for example is composed of only twenty lines of XML and the user can then immediately call the tool from within the import environment.

4 Technologies used

[5] http://software.textometrie.org

To choose the open-source components for our platform we made a trade off between available functionalities, efficiency and developers community history of available technologies (Heiden et al. 2010).
For the textual database component, we choose the IMS Open Corpus Workbench (Christ, 1994) because it is the most efficient for our specifications. For the statistical environment we choose R (2005) for its richness. For the scripting language we choose Groovy because it is a simple (although sophisticated) syntactic sugar idiom with Python and Ruby flavor strictly compiled in Java giving us access to all available Java components and packages, perfectly integrated above the platform core.

5 The core TXM Toolbox distribution

All the Java sources of the core Toolbox are publicly available under a GPL v3 license in the SVN repository of the project on Sourceforge at the address `https://sourceforge.net/projects/textometrie`. The developers wiki gives all the detailed instructions to download and build it in the Eclipse IDE.

6 Demonstrators

The alpha versions of the demonstrator applications are already available with their source code at the address `https://sourceforge.net/projects/textometrie`.
Those applications are build upon the core toolbox to bring textometric functionalities to the end user through a graphical interface.

6.1 The Eclipse RCP Application demonstrator

An application demonstrator has been developed with the Eclipse RCP framework for all platforms[6]. This framework gives all the necessary windows and widgets management services to build an efficient and same coherent graphical user interface for all platforms and includes very useful components to integrate other technologies easily. For example, a SVG plug-in is integrated to display the vector graphics generated by R libraries in the interface. Finally, for every platform, a setup is bundled to ease the installation of the application by the end user. Figure 1 shows a sample interface of the Windows version.
The setups for the different platforms are always available on Sourceforge at the address `https://sourceforge.net/projects/textometrie`.

[6] The Mac OS X platform version is planned.

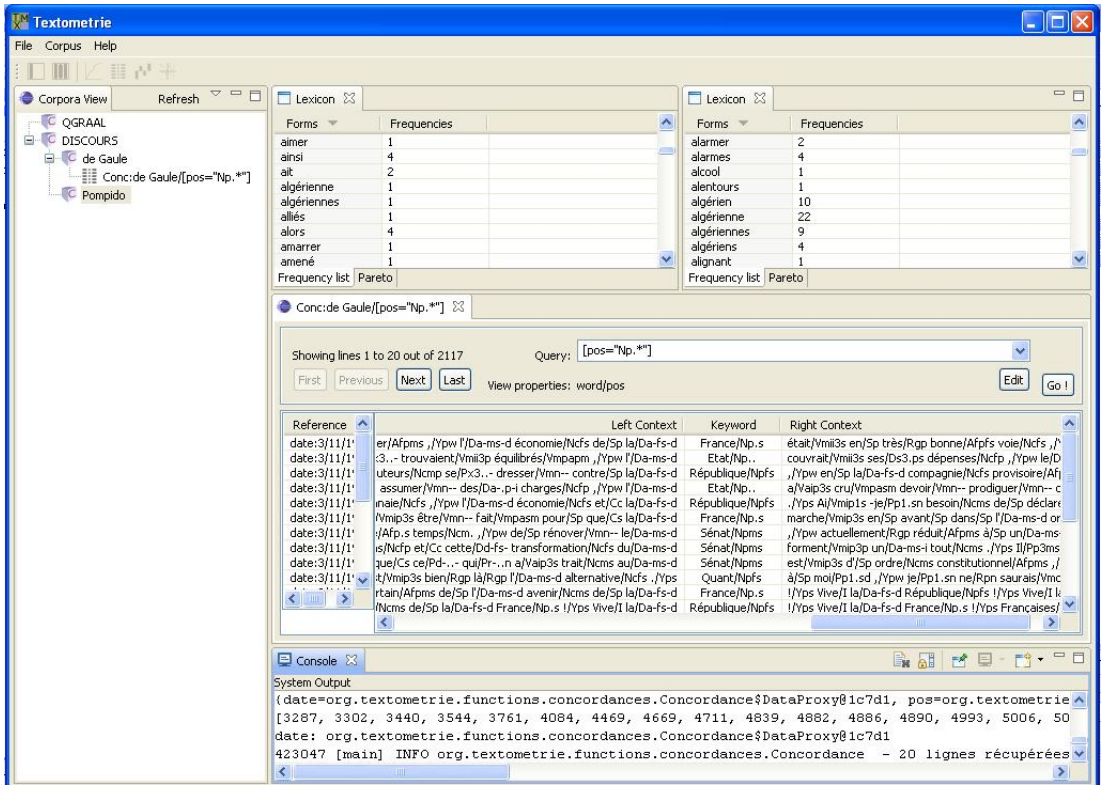

Figure 1 : sample graphical interface of the local RCP prototype application of the TXM platform on Windows. The left panel is a tree view allowing to browse all the different corpora and objects created during the working session (corpus, sub-corpus, partitions, vocabulary lists, concordances, graphics…). The upper panel has been split with the window manager to be able to compare side by side two different vocabularies from two sub-corpora and the lower panel is a kwic concordance of all the proper nouns of the corpus[7]. That concordance displays for each word its form and part of speech (any word property can be displayed in the order selected by the user). On the left side of the concordance, the reference columns displays the date of the production of the text for each concordance line. Any property of structural nodes can be part of the reference display.

6.2 The Web Application Grails demonstrator

A web application demonstrator as been developed with the Grails framework. That framework integrates the Prototype and YUI Ajax libraries used to build the graphical interface and protocol management between the client and the server. Figure 2 shows a sample interface in Firefox.

A prototype has been deployed online on a Tomcat J2EE servlet container at the address `http://txm.risc.cnrs.fr/txm`.

[7] The `[pos="Np.*"]` algebraic expression obeys the syntax of the search engine component. The "`pos`" word property name is for "part of speech". The "Np…" tag prefix for proper nouns obeys the syntax of the tagger used for the corpus.

Figure 2 : sample interface of the web application prototype of the TXM platform rendered by Firefox. The upper panel displays two different views of the edition of the manuscript « la Queste del Saint Graal » (Marchello-Nizia, to be published) : on the left side the image of one column of the manuscript, on the right side one of the three available critical editions encoded in XML-TEI P5 and rendered in HTML. The lower panel displays a kwic concordance of the "Lancelot" word. A double-click on a line of the concordance displayed in the upper panel the column numbered 160d (right column of verso of folio 160) of the edition in which the keyword appears highlighted in red. The concordance was built by the same toolbox which generated the one illustrated on figure 1.

7 Conclusion

This paper has exposed the rationale and the state of development of the open-source TXM textometry platform. This platform is made by the textometry community for the community. Its open-source nature permits everyone to evaluate the code, to make it better, to import new data sources and to implement new textometry functionalities with the rich statistical environment R. It already provides innovative analytical corpus tools and an original triple integration of 1) standard XML-TEI encoded corpora with 2) the CWB efficient search engine for linguistically annotated corpora and 3) R packages for statistical analysis of textual data.

Acknowledgments

The work described here has benefited from all the members of the project. Errors and omissions are of the sole responsibility of the author.

References

Benzécri, J.-P. et al. 1973a. *L'analyse des données. 1, La taxinomie*. Paris : Dunod.

Benzécri, J.-P. et al. 1973b. *L'analyse des données. 2, L'analyse des correspondances*. Paris : Dunod.

Christ, O. 1994. A Modular and Flexible Architecture for an Integrated Corpus Query System. *Proceedings of COMPLEX'94 (3rd Conf. on Computational Lexicography and Text Research)*, pp. 23-32.

Heiden, S., Pincemin, B., Magué, J.-P. 2010. TXM : Une plateforme logicielle open-source pour la textométrie – conception et développement. *Proceedings of JADT'2010 (10th Journées internationales d'Analyse Statistique des données Textuelles)*.

Lebart, L., Salem, , A., Berry, L. 1997. *Exploring Textual Data*. Boston : Kluwer academic publishers.

Marchello-Nizia C. (to be published). *Queste del saint Graal : Édition numérique interactive*, Lyon : ENS de Lyon.

Old Medieval French Database – BFM [online]. 2010. Lyon, <http://bfm.ens-lyon.fr>.

Pincemin B., Heiden S., Lay M.-H., Leblanc J.-M., Viprey J.-M. 2010. Fonctionnalités textométriques : proposition de typologie selon un point de vue utilisateur. *Proceedings of JADT'2010 (10th Journées internationales d'Analyse Statistique des données Textuelles)*.

R Development Core Team. 2005. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

TEI Consortium. 2008. *TEI P5: Guidelines for Electronic Text Encoding and Interchange*. Version 1.1.0., 4th July. Lou Burnard & Syd Bauman eds., TEI Consortium. http://www.tei-c.org/Guidelines/P5.

Unicode Consortium. 2006. *The Unicode Standard*, Version 5.0, 3rd November, Addison-Wesley Professional; 5th ed., http://unicode.org

Software References

ANC Tool: http://www.americannationalcorpus.org/tools/index.html#anc-tool

Eclipse RCP: http://www.eclipse.org/home/categories/rcp.php

Grails: http://grails.org

Groovy: http://groovy.codehaus.org

Hyperbase: http://ancilla.unice.fr/~brunet/pub/hyperbase.html

IMS Open CWB: http://cwb.sourceforge.net

JCP: http://jcp.org

Lexico 3: http://www.cavi.univ-paris3.fr/ilpga/ilpga/tal/lexicoWWW

OSGi: http://www.osgi.org

SATO: http://www.ling.uqam.ca/ato/sato

Spring: http://www.springsource.org

TXM : https://sourceforge.net/projects/textometrie

Weblex: http://weblex.ens-lsh.fr/wlx

Xaira: http://www.xaira.org