TEI Analytics: converting documents into a TEI format for cross-collection text analysis

Brian L. Pytlik Zillig
Center for Digital Research in the Humanities, University of Nebraska, Lincoln, NE, USA

Abstract

For the purposes of large-scale analysis of XML/SGML files, converting humanities texts into a common form of markup represents a technical challenge. The MONK (Metadata Offer New Knowledge) Project has developed both a common format, TEI Analytics (a TEI subset designed to facilitate interoperability of text archives) and a command-line tool, Abbot, that performs the conversion. Abbot relies upon a new technique, schema harvesting, developed by the author to convert text documents into TEI-A. This article has two aims: first, to describe the TEI-A format itself and, second, to outline the methods used to convert files. More generally, it is hoped that the techniques described will lead to greater interoperability of text documents for text analysis in a wider context.

Correspondence:
Center for Digital Research in the Humanities, University of Nebraska, Lincoln, NE, USA.
E-mail:
bpytlikz@unlnotes.unl.edu; bzillig1@unl.edu

1 Introduction

Digital texts encoded in Text Encoding Initiative (TEI) format can, in theory, be gathered into large text corpora for text analysis. However, the prospect of gathering thousands, or tens of thousands, of texts into a common analytical framework is as fraught as it is intoxicating. With the growing number of digital text collections, it ought to be possible to combine and search across them for patterns that would be simply impossible to discern without the aid of computing. Regrettably, text collections typically employ their own markup scheme, incompatible with others, that reflects editorial intentions and purposes related to the goals of that specific project. While those purposes are no doubt clear and sensible to their respective editors, the resulting structural differences among collections constitute deep and problematic gaps in what might otherwise represent interoperability. Nevertheless, should it be achieved even for the limited purpose of text analysis, interoperability is a goal worth pursuing.

The Monk (Metadata Offer New Knowledge) Project has, with support from the Andrew W. Mellon Foundation, developed a Web-based system for undertaking text analysis and visualization with large, full-text literary archives (Metadata Offer New Knowledge, 2006). The primary motivation behind the development of this system has been the profusion of sites like the Brown Women Writers Project, Perseus Digital Library, Wright American Fiction, Early American Fiction, and Documenting the American South, along with the vast literary text corpora offered by organizations like the Text Creation Partnership and Chadwyck-Healey. Every one of these collections represents fertile ground for text analysis. But if they could be somehow combined, they would constitute something considerably more powerful: an immense and searchable full-text literary corpus.

Literary and Linguistic Computing, Vol. 24, No. 2, 2009.

doi:10.1093/llc/fqp005 Advance Access Published on 15 April 2009

2 Obstacles

Howard Besser asserts that '[i]n moving from dispersed digital collections to interoperable digital libraries, the most important activity developers need to focus on is standards' (Besser, 2004). For MONK, a number of issues related to the lack of standards served to impede the combining of files from different collections. Alas, not all digital humanities texts exist in TEI format, or even in the Extensible Markup Language (XML). The Text Creation Partnership (TCP) EEBO (Early English Books Online) files, for example, constitute neither compliant TEI nor XML. They are encoded in Standard Generalized Markup Language (SGML) format, in a structure that resembles TEI but is apparently unique to the TCP. That said, the markup structure below the <text/> element is more or less recognizable as TEI Lite. EEBO files have other problems, including relying upon external headers that are not stored within the SGML files themselves and deploying the caret (^) to mark superscripted characters. (Each superscripted character is preceded by a caret, such as in M^r^s.) Additionally, the EEBO texts use the vertical bar as a character for a soft hyphen at the end of a line, and they use the + sign to indicate the lack of a hyphen where a word is split between two lines.

Instead of the more contemporary practice of using Unicode, several of the collections used for MONK employ named character entities, as in this example from EEBO:

<!ENTITY obreve SDATA "&obreve;"- -= LATIN SMALL LETTER O WITH BREVE - ->

The obstacles standing in the way of creating a consistent corpus for text analysis are fairly well known. While all of the collections mentioned above are encoded in XML or SGML and most of them are TEI-conformant, local variations (even within the TEI-conformant group) can be so profound as to prohibit anything but the most rudimentary form of cross-collection searching. Even with XML-formatted TEI, a scheme that theoretically should lend itself to interoperability, not all texts are created equal. TEI-A, a subset of TEI in which varying text collections can be expressed, grew out of our desire to make MONK work with extremely large literary text corpora of the sort that would allow computational study of literary and linguistic variance across boundaries of genre and geography and over long periods of time. Simply put, TEI-A exists because of interoperability problems associated with the lack of tight standards for text encoding. The procedure for getting from the various forms of source markup to TEI-A involves a command-line tool, Abbot, that performs the conversion. I created Abbot in collaboration with Stephen Ramsay, University of Nebraska, and with editorial input from Martin Mueller, Northwestern University. The Abbot application relies upon a new technique, schema harvesting, developed by the author to convert text documents into TEI-A.

3 TEI Analytics

Local text collections vary not because archive maintainers are unaware of the importance of standards or interoperability but because particular local circumstances sometimes demand customization. The nature of the texts themselves may necessitate a custom solution, or something about the storage, delivery, or requirements for display may favor particular tags or particular structures. Local environments also require particular metadata conventions (even within the TEI header). This is in part why the TEI Consortium provides a number of prefabricated customizations, such as TEI with MathML and TEI Lite, as well as modules for drama, speech, and manuscripts. The TEI Consortium's Roma tool allows users to create a myriad of TEI subsets, all of which may be also be customized using the ODD (One Document Does It All) mechanism.

TEI-A is designed with a very particular purpose in mind. If one were setting out to create a new literary text corpus for the purpose of undertaking text analysis work, the most sensible approach might be to begin with one of TEI's pre-fabricated tagsets (TEI Corpus, perhaps). In the case of the MONK project, however, we are beginning with collections that have already been tagged using diverse versions of TEI with local extensions. TEI-A is therefore designed to exploit common denominators in these texts while at the same time adding

new markup for data structures useful in common analytical tasks (e.g. part-of-speech tags, lemmatizations, word tokens, and sentence markers). The goal is to create a P5-compliant format that is designed not for rendering but for analytical operations such as data mining, principal component analysis, word frequency study, and n-gram analysis. In the particular case of MONK, documents marked up in this way have a relatively brief lifespan; once converted, they are read in by a system that stores the information in a relational database. But before that can happen, the texts themselves must be re-expressed in the new format.

4 Using an ODD and a Meta-stylesheet for File Conversion

Refashioning hundreds or thousands of novels and plays into TEI-A requires a programmatic solution that can be performed as a batch operation. Converting thousands of files by hand (even with a good XML editor) is simply not practical. The TEI-A format specifies approximately 135 elements, a fairly small schema compared to the 504 elements defined in the full TEI (TEI-All). TEI Lite is a bit larger than TEI-A, with 145 elements. Our basic approach to the file conversion involves a process I call schema harvesting, which relies upon a TEI-A schema. First, I used Roma to create a base schema (containing the four required modules: core, tei, header, and textstructure) for TEI P5 documents, which I then extended using an ODD file. The TEI-A ODD file contains these main TEI modules:

```
<moduleRef key="core"/>
<moduleRef key="tei"/>
<moduleRef key="header"/>
<moduleRef key="textstructure"/>
<moduleRef key="transcr"/>
<moduleRef key="analysis"/>
<moduleRef key="figures"/>
<moduleRef key="gaiji"/>
<moduleRef key="linking"/>
<moduleRef key="drama"/>
```

Modules added to the base schema include: transcr, analysis, figures, gaiji, linking, and drama. The ODD adds three new elements to TEI-A: <sb> for sentence breaks, <sub> for subscript, and <sup> for superscript. Four other elements were altered. The content model of <w> was altered to allow <hi> as a child, the @scheme attribute on the <keywords> element was changed to optional, the <title> element in the TEI header was changed so that it cannot have child elements, and <w> was given attributes for recording part of speech and lemma information. Any changes to the TEI-A schema begin with editing the ODD file and using Roma to output a new schema. Because the schema contains the document logic for files that validate under TEI-A, and because this logic exists in an XML file, it can be queried and exposed with methods that rely heavily upon the Extensible Stylesheet Language for Transformations (XSLT). Bradley writes that, '[a]s XML becomes the scheme of choice for feeding data to many programs, it is likely that XSLT will be used more and more to transform one type of XML markup into another' (Bradley, 2004). The MONK approach to fashioning a uniform text base has indeed been to rely on XSLT.

To harvest the document logic in the TEI-A schema, I authored an XSLT 'meta-stylesheet' that reads the schema to determine the form into which the source files may be converted. This first stylesheet subsequently generates a second XSLT stylesheet used for the actual conversion into TEI-A. The second, or conversion, stylesheet contains the conversion templates (or, more accurately, <xsl:template> elements) which hold the specific instructions to get from the source markup to the TEI-A custom P5 implementation. The term schema harvesting was intended to convey the important detail that the TEI-A schema was itself being harvested for information. Examples of data that are harvested by default are quite simple and low-level: (1) the names of allowable TEI-A elements and (2) the names of allowable TEI-A attributes. For each element in the TEI-A schema, an XSLT template is automatically created in the second stylesheet. Of course, all of this assumes that the text markup in the source files bears some similarities to the desired TEI-A output format. It helps, for example, if there is a <text>, <body>, or <p> element in the source file, though this is not required.

Allen Rennear describes text markup as serving 'to identify logical or physical features or to control later processing' (Rennear, 2004). The MONK text conversion processing described here honors the principal features in source documents, but is selective in order to keep the TEI-A texts and aspects of their conversion fairly uncomplicated. To that end, as I detail later in this article, some elements in the TEI header are discarded. Elements in the source text that are permitted in TEI-A are passed through to the output. Those that are not have their tags stripped and their text nodes are passed through. Attributes of each element that passes through the transformation are reviewed against the TEI-A schema. Attributes that are not permitted are removed, and those that are required are added. A key point is that, apart from the header information, no text content is removed by default.

Elements that are not needed for analysis are removed or renamed according to the requirements of MONK. For example, numbered s are replaced with un-numbered ones. The conversion is mainly concerned with what Goldfarb calls 'unformatted abstractions' (Goldfarb, 2003). Therefore, source markup that is exclusively concerned with formatting is discarded. In addition, simpler structures are used, where possible, to replace any that are unnecessarily verbose or complex. For example, this structure:

```
<p>
  <q>
    <text>
      <body>
        <div type="letter"/>
      </body>
    </text>
  </q>
</p>
```

is replaced with this somewhat simpler version:

```
<floatingText type="letter">
   <body/>
</floatingText>.
```

Bibliographical information is often useful for text analysis, and both copyright and responsibility information must be maintained, but much of the information contained in the average <teiHeader> (e.g. revision histories and records of workflow) is not relevant to the task of text analysis. For this reason, TEI-A uses a simplified form of the TEI header where <profileDesc>, <refsdecl>, <seriesstmt>, and the like, are eliminated. Outside of the header most conversions are simple and straightforward, and can be performed by a fairly simple template in the meta-stylesheet, excerpted here:

```
<xsl:element name="{$elementName}">
    <wxsl:for-each select="./@*">
      <xsl:for-each select="$associatedAttributeList/list">
        <wxsl:choose>
          <xsl:for-each select="child::item[string-length(.) &gt 0]">
            <wxsl:when>
              <xsl:attribute name="test">
                <xsl:value-of select="concat('name()=',$apostrophe,.,$apostrophe)"/>
              </xsl:attribute>
              <wxsl:attribute name="{$attributeNameParam}">
                <wxsl:value-of select="."/>
              </wxsl:attribute>
            </wxsl:when>
          </xsl:for-each>
          <wxsl:otherwise/>
        </wxsl:choose>
      </xsl:for-each>
    </wxsl:for-each>
    <wxsl:apply-templates/>
</xsl:element>
```

This example template constructs the main, or default, template in the conversion stylesheet by matching on elements in the TEI-A schema, and outputting approximately 140 XSLT templates in the second stylesheet. The <xsl:namespace-alias/> element is used to replace a namespace in the meta-stylesheet (wxsl) with a different namespace in the output template (xsl). Many cases exist wherein the main template is not sufficient to express the desired conversion. Examples include situations in which <lb> tags are suppressed if they occur within an <orig> element, and @anchored attributes are switched from 'yes' to 'true'. In addition, complex or conditional element conversions or any custom additions or subtractions that are desired are added to the meta-stylesheet, and take this general form:

```
<xsl:comment>begin substitution of quote tag for q tag and suppression of q tags that surround quoted text </xsl:comment>
<wxsl:template match="q | Q" priority="1">
  <wxsl:choose>
    <wxsl:when test="child::text">
      <wxsl:apply-templates/>
    </wxsl:when>
    <wxsl:otherwise>
      <quote>
        <wxsl:apply-templates/>
      </quote>
    </wxsl:otherwise>
  </wxsl:choose>
</wxsl:template>
<xsl:comment>end substitution of quote tag for q tag and suppression of q tags that surround quoted text </xsl:comment>
```

In this example, the @priority attribute overrides the normal q | Q template constructed by the default template in the meta-stylesheet. All processes are initiated by the Abbot program in the following sequence:

1. Use the MonkMetaStylesheet.xsl stylesheet to read the TEI-A schema
2. Generate the XMLtoMonkXML.xsl stylesheet, as a result of the prior task
3. Convert the input collection to TEI-A
4. Parse the converted files against the MONK schema and log any errors
5. Move invalid files to a quarantine folder

These steps are expressed in a sequence of Unix shell scripts, and all source files are retained in the processing sequence so that the process can be tuned, adjusted, and re-run as needed without data loss. Depending on the hardware and the size of the input files, the main conversion process takes approximately 60 minutes for roughly 1,000 novels. Unix stream editor (sed) scripts are used to perform any global edits that may be required, such as removing errant empty namespaces in the output elements and adding a schema declaration to the top of each file.

In the validation phase, Abbot parses each output file using the Sun Multi-Schema Validator (MSV). Any file that fails to parse according to the TEI-A RelaxNG schema is automatically moved to a quarantine folder for invalid files. Manual inspection of quarantined files permits the user to identify errors in the input files, the output, or the stylesheets. For quarantined files, follow-up activities typically involve improving the XSLT conversions to bring the output files into conformance with the schema or, in rare cases, ignoring some problematic outliers and hand-editing the files.

Witten, referring to data-mining, proclaims '[a]nything discovered will be inexact. There will be exceptions to every rule and cases not covered by any rule' (Witten and Frank, 2005). Such has been my experience with the Abbot text conversions. For example, there have been a very small number of occasions in which the source files were themselves faulty and required a degree of pre-processing before conversion into TEI-A. It would be foolish to state that the Abbot conversion routines are sufficiently robust and sensitive as to suit all marked-up texts, but in general the results have been extremely positive and Abbot is now a key component in the MONK toolchain. Use of schema harvesting for MONK has resulted in the successful conversion of EEBO files into TEI-A, as well as conversion of files from the Wright American Fiction Archive, and Eighteenth Century Fiction and Nineteenth Century Fiction collections. The size of the converted collections is approximately

100 million words. MONK plans to make the TEI-A schema and the Abbot conversion tool freely available in Spring 2009.

5 Conclusion

John Unsworth, in writing about the rhetorical model in the humanities, asserts: 'we believe that by paying attention to an object of interest, we can explore it, find new dimensions within it, notice things about it that have never been noticed before, and increase its value' (Unsworth, 2006). TEI-A performs an important function within the larger ecology of digital humanities by making the patterns already present across large numbers of textual objects more noticeable by situating them within a single text analysis framework. Even without the need for ingestion into a noticing space, TEI-A facilitates text analysis of disparate source files simply by creating a consistent and unified XML representation. Moreover, for the problem of file conversion the schema-harvesting approach (in which a small XSLT stylesheet reads the schema and then generates the conversion stylesheet) may be useful in other applications where interoperability is a requirement.

References

Metadata Offer New Knowledge. *Humanities Text-mining in the Digital Library*, https://apps.lis.uiuc.edu/wiki/display/MONK/Short+Version+of+Mellon+Proposal (accessed 25 September 2006).

Besser, H. (2004). The past, present, and future of digital libraries. In Schreibman, S., Siemens, R. and Unsworth, J. (eds), *A Companion to Digital Humanities*. Oxford: Blackwell.

Bradley, J. (2004). Text tools. In Schreibman, S., Siemens, R. and Unsworth, J. (eds), *A Companion to Digital Humanities*. Oxford: Blackwell.

Rennear, A. (2004). Text encoding. In Schreibman, S., Siemens, R. and Unsworth, J. (eds), *A Companion to Digital Humanities*. Oxford: Blackwell.

Goldfarb. (2003). *XML Handbook*, 5th edn. p. 59.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann Series in Data Management Systems*. Amsterdam: Morgan Kaufman.

Unsworth, J. (2006). *Digital Humanities Beyond Representation*. University of Central Florida, Orlando, FL, November 13, 2006. http://www3.isrl.uiuc.edu/~unsworth/UCF/.