Investigationes Linguisticae, vol. XI, Poznań, December 2004

From SGML to XML with TEI: Automated Conversion of a Corpus of Polish from P3 to P4 Format

Maciej Ogrodniczuk

Faculty of Modern Languages and Oriental Studies, Warsaw University
Krakowskie Przedmieście 26/28, 00-927 Warszawa

Maciej.Ogrodniczuk@empolis.com

Abstract

The article presents experiences gathered in the process of migration of an SGML corpus encoded in TEI P3 format to XML-enabled TEI P4.

1. Introduction

The article presents experience gathered in the process of migration of an SGML corpus encoded in TEI[1] P3 format into XML-enabled TEI P4 format. The data used for this task constituted *The corpus of Polish language of the sixties* – the enriched[2] version of the *Corpus of Frequency Dictionary of Contemporary Polish* [9]. Its SGML version was created with the support of several projects, including the State Committee for Scientific Research grant *Test suites for verification and evaluation of analyzers of Polish* supervised by Janusz S. Bień. In particular, autor's Master Thesis [11] aimed at designing and verification of the representation format for Polish linguistic data compatible with SGML (Standard Generalized Markup Language) [7] and TEI [14] standards.

In 2000, when current encoding of the Corpus was prepared, the most recent edition of the TEI guidelines was the SGML P3 version;[3] the preparation of P4 edition, planned as fully XML-compatible, was still under way. Today, after it has been released, the author would like to comment on the differences between both editions and the possibility of automated conversion of TEI P3-compatible resources into the new XML format[4].

[1] Text Encoding Initiative – "an international and interdisciplinary standard that helps libraries, museums, publishers, and individual scholars represent all kinds of literary and linguistic texts for online research and teaching, using an encoding scheme that is maximally expressive and minimally obsolescent"; see http://www.tei-c.org.

[2] See [1] and [12].

[3] Actually, the version released in mid-1999 under misleading – since it contained relatively minor corrections as compared to base P3 dating to May 1994 – code name *P4beta*.

[4] In the meantime, Adam Przepiórkowski included the corpus of Frequency Dictionary into the IPI PAN Corpus – a large (over 300 million segments) corpus of Polish, developed at the Institute of Computer Science, Polish Academy of Sciences (see http://korpus.pl/en/index.php?page=project.html). For this purpose, the corpus has been encoded in XCES – Corpus Encoding Standard for XML (http://www.cs.vassar.edu/XCES/).

2. The P4 version of the *TEI Guidelines*

The work on an XML version of TEI was started by one of the TEI editors, C. M. Sperberg-McQueen, who presented, as early as in 1999, the draft of a method of conversion of TEI DTD from SGML into XML. The concepts presented in this unpublished document has been used, to a large extent directly, in the current version of TEI.

As editors of the brushed-up *TEI Guidelines* [14] write in the introduction, the main purpose for the new edition was "to revise both the text and the DTDs of the scheme in a way compatible with the use of either SGML or XML" (to make the documents created according to previous versions of the TEI encoding scheme consistent with the new version) – and such declaration is a good summary of the complete Guidelines, because XML compatibility have been ensured by using strictly minimalistic approach.

The only part of the document where XML spirit is clearly noticeable is the popular chapter 2 of *the Guidelines* (functioning for a long time as a self-contained tutorial) – introducing the syntax of TEI encoding metalanguage which was, until now, SGML. The chapter has been rewritten for the new version and now concentrates on XML. Further changes were applied to examples of usage of TEI constructs throughout the whole document, now putting attribute values in quotation marks and eliminating minimization of tags – prohibited in XML.

Following XML syntax, case-sensitivity of the characters in TEI element names has been normatively established[1] and *the Guidelines* were extended with the comments concerning case of characters in XML identifiers (values of `ID` fields, referenced by `IDREF` fields).

Apart from instructions describing in principle obvious techniques enforced by the new syntax, many of the comments gathered by *the Guidelines* are only suggesting availability of new XML solutions, without any deeper analysis of their usefulness in the context of TEI. Good example illustrating indecisiveness of TEI authors is the description of mechanism of extended pointers, based on HyTime [8] standard – without the slightest notice of widely-used in XML world XPath standard [17]. In this case *the Guidelines* only state that the reference to new standards will appear in one of the following editions of TEI.

Another controversial recommendation is to use for language representation, even within XML documents and similarly to what previous versions of TEI suggested, global `lang` attribute and not common for XML documents `xml:lang` attribute[2].

3. TEI P4 in practice

The need to enforce compatibility of previously created TEI documents with the new version of the DTD seems to have been the most important factor that influenced the shape and usage of the new document type definition: the XML features are included in the document instance (in standard TEI way) only after declaration of `TEI.XML` parameter entity:

```
<!DOCTYPE TEI.2 SYSTEM "http://www.tei-c.org/P4X/DTD/tei2.dtd"
[ <!ENTITY % TEI.XML "INCLUDE">
  ... ]>
```

Such construct provides a syntax switch that makes use of parameter section mechanism (in a more interesting way than the TEI itself) to control the DTD creation. Taking advantage of the fact that SGML TEI was using only two occurrence indicators: `- O` (start tag has to be specified, end tag can be omitted) and `- -` (both start and end tag must be specified), the differences between SGML and XML in DTD syntax have been represented in a simple way, by defining two parameter entities `om.RO` (abbr. for *omissibility-required-optional*) and `om.RR` (abbr. for *omissibility-required-required*), used in place of occurrence definition.

[1] Using the technique known as *Camel-case normalization*, which refers to usage of uppercase characters in components of element names.

[2] See e.g. [4] and section 2.12 of [5].

Their default values are corresponding occurring indicator strings, redefined with an empty string if the XML features are turned on:

```
<!ENTITY % TEI.XML 'IGNORE' >

<![%TEI.XML;[
<!ENTITY % om.RO " >
<!ENTITY % om.RR " >]]>

<!ENTITY % om.RO '- O' >
<!ENTITY % om.RR '- -' >
```

Values of `om.RO` and `om.RR` are then used in element definitions in a regular way:

```
<!ELEMENT element_name %om.RO; (content_model)>
```

As shown in the example above, predefined value of the `TEI.XML` parameter is `IGNORE`, which makes the DTD (and consequently, enforces full backward compatibility of TEI-compliant documents) SGML-compatible by default.

4. Data migration

4.1. Is there a need to migrate?

The result of the postulate of compatibility of previously created data with the current version of TEI is that no data migration between SGML and XML is necessary. However, in many cases, the cost of such step seems relatively low as compared to potential benefits, which makes the whole process worth considering.

The major argument should be for sure growing popularity of XML, conducive to better (than for SGML) availability of tools that can assist further processing of XML-encoded resources. This could be exemplified by using XSLT stylesheets for processing XML files to simplify complex tasks of extraction, further conversion or preparation of fragments of resources for presentation.

Obviously, successful migration may require certain efforts – and they seem the bigger, the more enhancements to TEI DTD were used in the project of our own document type definition. The general problem of DTD conversion, which will not be dealt with in this article, seems not trivial: in extreme cases, e.g. when SGML constructs not represented in XML were used for extending DTD (like attributes with `#CURRENT` characteristics[1]), the task can be even impossible without major rebuild of the corpus.

4.2. The format of the source data and its consequences

The corpus of Polish language of the sixties encoded according to TEI P3 notation was available in the form of five SGML files, representing five styles of written Polish (scientific texts, news, essays, prose and drama). Each file contained 2000 samples about 50 words each [9].

The encoding of structural information in the corpus[2] was technically simple: data samples, gathered in `<group>`s, were provided with unique identifiers and bibliographic references:

[1] Differences between SGML and XML with the list of prohibited constructs can be found e.g. in [3].
[2] See [13] for the details of morphological encoding.

```
<teiHeader type=text>
   <!-- ... header information for the style ... -->
</teiHeader>

<text lang=PL>
   <group>
      <text id=pu0001>
         <front>
            <head>
               <bibl>
                  <!-- ... bibliography of the sample ... -->
               </bibl>
            </head>
         </front>
         <body>
            <!-- ... text of the sample ... -->
         </body>
      </text>
   </group>
</text>
```

Additionally, text contents of each sample was split into syntax units using tags for paragraph, sentence and word level. Each element representing a word was recorded with the base form of the word and its morphological information[1]:

```
<body>
   <text>
      <p>
         <s>
            <!-- ... -->
            <w id="wp001151" lemma="zwolennik"
               ana="SSIP---------P">zwolennikiem</w>
            <!-- ... -->
         </s>
         <!-- ... -->
      </p>
   </text>
</body>
```

[1] Symbols used to represent morphological information for Polish are described in detail in [6].

The main TEI header was recorded in separate master file which included five files containing styles in a common SGML/XML way:

```
<teiCorpus.2>
   <teiHeader type=corpus>
      <!-- ... header information for the corpus ... -->
   </teiHeader>

   <tei.2 id=stylA>
      &a-publ;
   </tei.2>

   <!-- ... consecutive styles ... -->

</teiCorpus.2>
```

Similarly, *writing system declaration* (WSD), *feature system declaration* (FSD)[1] as well as feature structures library documenting encoding scheme using in the corpus and SGML declaration allowing to use ISO 8859-2-encoded Polish characters were recorded as separate files.

Regularity of the text encoding format, already shown in the examples above, was the result of automated generation of the corpus in SGML at the stage of adding morphological information. Its consequence was such easy-to-implement plan of migration of the corpus to XML format:

1. enclosing all attributes in quotation marks[2],
2. normalizing the case of characters in TEI element names, according to TEI Guidelines,
3. adding the XML declaration,
4. including XML features in the internal subset of DTD declaration.

4.3. Simple migration scripts

Implementation of migration scripts that could execute the algorithm described above seemed very easy. Further analysis which consisted in creation of the list of attributes not yet enclosed in quotation marks confirmed this assumption – the list was limited to identifiers of samples (`id` attributes for `<text>` elements), identifiers of words (`id` attributes for `<w>` elements), single occurrences of attributes representing language of each style (`lang` attributes for `<text>` elements at the style level) and types of TEI headers (`type` attribute for `<teiHeader>` elements).

Fortunately, the case of characters in the name elements was already conformant to requirements of *TEI Guidelines*, except for the base element `TEI.2`.

Implementation of migration scripts in Perl[3], particularly suited for this purpose, lasted a few minutes. Here is the script used to convert each style:

```
print "<?xml version=\"1.0\" encoding=\"iso-8859-2\"?>\n";
while (<>)
{ s/^     <text id=(.*)>$/     <text id=\"$1\">\n/;
  s/^(.*<w id=)(.*)( lemma.*)$/$1\"$2\"$3\n/;
  s/(<teiHeader type=)(text)/$1\"$2\"/;
  s/(<text lang=)(PL)/$1\"$2\"/;
```

[1] Using feature structures in TEI documents is described e.g. in [10].

[2] For several attributes, e.g. `lemma`, this operation has already been completed for the SGML version, as a result of possibility of encountering spaces in word forms (which is, in turn, a consequence of joint representation of multiple-word forms, such as *z grubsza*).

[3] See one of the numerous Perl tutorials, such as [16].

and the script for the master file:

```
print "<?xml version=\"1.0\" encoding=\"iso-8859-2\"?>\n";
while (<>)
{ s/<tei(.2 id=)(styl.)/TEI$1\"$2\"/;
  s/(<teiHeader type=)(corpus)/$1\"$2\"/;
  print; }
```

Obviously, what is worth stressing is that such simple migration scheme was the result of simple representation of data in the source corpus, which had in many places already been conformant to the target format. In general case it would for sure take more work to implement migration scripts.

4.4. Generic migration

The general solution to the migration problem, as suggested by TEI, is using the `osx` tool, delivered as a part of `OpenSP` package, available under GNU General Public License. This tool, according to the note on the Web page of TEI working group for migration initiative[1], has been recently
extended to improve its usefulness for conversion of data encoded
in earlier TEI formats.

The article [15] outlines the general migration process:

- SGML to XML conversion using `osx`,
- normalization of the case of TEI element and attribute names using `tei2tei.xsl` stylesheet.

4.4.2. `osx` converter

The converter is run with additional command line parameters to define the conversion result format:

```
osx -xcomment -xempty -xno-expand-external -xno-nl-in-tag
sfpw.sgm > sfpw.xml
```

Here is a short documentation of `osx` parameters:

`-xlower`	–	attribute names (as well as values) will be output in lowercase; more explanation below
`-xcomment`	–	SGML comments will be output.
`-xempty`	–	shortened syntax will be used outputting empty elements
`-xno-expand-external`	–	external entities will not be expanded, but saved in separate files
`-xno-nl-in-tag`	–	all attributes will be output in single line (default: each in separate line)

Using `-lower` parameter is not meaningful in case `tei2tei.xsl` stylesheet is applied later, because one of the stylesheet tasks is to bring about the proper case of TEI element names. However, what is worth noticing here is that this parameter is applied for all output characters, which results in a side effect which is conversion of attribute values. What is more, the default processing mode for `osx` is changing all element and attribute names to uppercase, which makes impossible to retain the original case of attribute values. This is no longer dealt with by

[1] See http://www.tei-c.org/Activities/MI/Tools/

`tei2tei.xsl`, because such change does not cause validation problems, nevertheless it alters the form of the text for no reason.

Another irritating feature of `osx` is outputting default attribute values in the result file – even though they were not used in the source file. Again, it produces entirely correct format, but spoils the ideas represented in the original file. If we would like to keep the source format of the file we would have to either temporarily (for the time of migration) remove the default values from the DTD (exactly, turn them
off by replacing the real values with `#IMPLIED` or process the output of `osx` with additional filter (created e.g. basing on the `tei2tei.xsl` stylesheet).

4.4.3. `tei2tei.xsl` stylesheet

The next additional step in the conversion process aims almost exclusively at eliminating the side effects of running `osx`. The action is restricted to running free[1] XSLT processor – `saxon`[2].

```
saxon sfpw.xml tei2tei.xsl > sfpw2.xml
```

Major actions carried out by the stylesheet are normalization of XML element names used by TEI DTD and reformatting text for better viewing.

4.5. Character encoding and feature structure declaration

Although *the Guidelines* mention new character encoding standards such as ISO 10646 and Unicode they do not recommend any of them for encoding of TEI documents.

In our case Polish characters in the corpus were encoded according to ISO 8859-2 standard, so no conversion was necessary.

Still, what should be changed is documentation of applied method of character encoding. Standard (for TEI) mechanism of specifying writing system declaration (WSD) cannot be used as before, by associating WSD as dependent document:

```
<!ENTITY wsd.polish SYSTEM "iso88592.wsd" SUBDOC>
```

because of inavailability of SGML `SUBDOC` construct in XML), but it needs to be replaced with corresponding notation declaration (see section 25.6 of *The Guidelines*):

```
<!NOTATION wsd PUBLIC
   '-//TEI P3-1994//NOTATION Writing System Declaration//EN'>
<!ENTITY wsd.polish SYSTEM "iso88592.wsd" NDATA wsd>
```

Similar change needs to be applied to inclusion of feature system declaration into the master document, as described in section 26.1 of *the Guidelines*:

```
<!NOTATION fsd PUBLIC
   '-//TEI//Feature System Declaration (1994)//EN'>
<!ENTITY fsd.morf SYSTEM "morf.fsd" NDATA fsd>
```

[1] Available under Mozilla Public Licence 1.0.

[2] Lite (without the source code and sample applications – just the executable) but fully-functional version of `saxon` (code name *instant*). is available at the Web page of the project: http://sourceforge.net/projects/saxon. At the time when conversion was being made the most recent version of the processor was 6.5.3.

Readers accustomed to strict markup language requirements could be astonished with such constructs, because such use of notation mechanism attaches both declarations to master document as unparsed entities (i.e. without checking well-formedness of referenced data). Considering the fact that previous versions of TEI assumed that both writing system declaration and feature system declaration are valid fragments of structured documents (i.e. SGML documents, what could be verified automatically while parsing the master document), such relaxation of the document format is not easily explicable. This decision could be possibly explained by concern for the consequences of changing the way declarations are connected to master document – enforcing structural well-formedness of dependant documents could require redesigning of the core parts of TEI DTD.

4.6. Conclusions

Both migration patterns described above help producing valid XML documents – although they are different in format. Preservation of original appearance of the text is certainly of minor concern in most applications, but it should be known that it can sometimes facilitate real usage of corpus information. Unimportant from XML point of view argument can be e.g. the intention to use fragment of corpus with its actual formatting for presentation of query results or allowing users for direct edition of samples. In such case original encoding of corpus data in XML file can be of great importance and we should not resign from keeping it without reason (and the fact that generic conversion process result in such reformatting does not seem a good enough reason).

Consequently, the decision of creating own migration scripts should be taken after considering the importance of preserving original format of converted data (mostly, white space) and whether implementation of such scripts will be simpler than fine-tuning the stylesheet that processes the result of generic conversion to desirable form (which needs answering the question about the degree of XML-ization of the current SGML data – were the attributes originally enclosed in quotation marks or apostrophes, were end tags always or at least very often used etc.) In case of easier conversions – such as ours – even leaving aside the format of the corpus, implementing own migration scripts seems much simpler.

5. Final remarks

As it turns out, TEI still depends much on SGML and *The Guidelines* seem to have taken just a first small step towards XML rather than consciously adopted the new standard. Reading TEI recommendations leaves strong impression that XML features are to a large extent underused in the current edition, taking into consideration the present position of XML in the world of corpora.

Hopes can be built on the next edition, P5, which is scheduled to be released by the end of 2004. It is promised to make use of XML Schema languages and provide support for XML namespaces – both widely used in XML world for 3-4 years. Such change would bring more possibilities of expression than old SGML way (e.g. stronger data-typing), so moving towards XML seems inevitable, even for older data.

At any rate, it may prove useful to consider an attempt to convert TEI-encoded data into current XML-ized version. As shown with our simple example, if the corpus is limited to standard TEI concepts, automated conversion seems easy. In a general case, when advanced SGML constructs are met, this task can prove to be fairly complex.

References

[1] Bień, J. S., Woliński, M. 2003. Wzbogacony korpus Słownika frekwencyjnego polszczyzny współczesnej. [In:] J. Linde-Usiekniewicz, R. Huszcza (Eds.) *Prace językoznawcze dedykowane Profesor Jadwidze Sambor*. Warszawa: Wydział Polonistyki Uniwersytetu Warszawskiego, pp. 6 –10.

[2] Bień, J. S., Woliński, M. (Eds.) 2001. Wzbogacony korpus Słownika frekwencyjnego polszczyzny współczesnej. Warszawa. Compressed CD image: http://www.mimuw.edu.pl/polszczyzna/wksf/wksf.iso.bz2.

[3] Clark, J. 1997. Comparison of SGML and XML. World Wide Web Consortium Note. http://www.w3.org/TR/NOTE-sgml-xml.html.

[4] Dürst, M. J. 2004. Language tagging in HTML and XML. World Wide Web Consortium. http://www.w3.org/International/O-HTML-tags.html.

[5] Extensible Markup Language (XML) 1.0 2004. (Third Edition). World Wide Web Consortium. W3C Recommendation. http://www.w3.org/TR/xpath.

[6] Głowińska, K. Taksonomia morfologiczna dla Słownika frekwencyjnego. [In:] [2], Dokumentacja\taksonomia.pdf.

[7] ISO 8879 Information Processing – Text and Office Systems – Standard Generalized Markup Language (SGML). Geneva 1986. ISO (International Organization for Standardization).

[8] ISO/IEC 10744 Information Technology – Hypermedia/Time-based Structuring Language (HyTime). Geneva 1992. ISO (International Organization for Standardization).

[9] Kurcz, I., Lewicki, A., Sambor, J., Woronczak, J. Szafran, K. 1990. Sownik frekwencyjny polszczyzny współczesnej. Kraków 1990. Instytut Języka Polskiego PAN.

[10] Langendoen, D. Terence, Simons, Gary F. A Rationale for the TEI Recommendations for Feature-Structure Markup. [In:] N. Ide and J. Veronis (Eds.) *Text Encoding Initiative – Background and Context*. Kluwer Academic Publishers, pp. 191-209.

[11] Ogrodniczuk, M. 2000. Wykorzystanie SGML i TEI do zapisu polskich danych lingwistycznych. Master thesis, prepared under supervision of Dr. Janusz S. Bień. Warsaw: Faculty of Mathematics, Informatics and Mechanics, Warsaw University.

[12] Ogrodniczuk, M. 2003. Nowa edycja wzbogaconego korpusu słownika frekwencyjnego. [In:] Stanisław Gajda (Ed.) *Językoznawstwo w Polsce. Stan i perspektywy*. Polska Akademia Nauk – Komitet Językoznawstwa, Uniwersytet Opolski – Instytut Filologii Polskiej. Opole, pp. 181-190. http://www.mimuw.edu.pl/~jsbien/MO/JwP03/.

[13] Ogrodniczuk, M. 2003. Rozszerzenie opisów morfologicznych w tekstach korpusu „Słownika frekwencyjnego polszczyzny współczesnej". [In:] Jadwiga Linde-Usiekniewicz, Romuald Huszcza (Ed.) *Prace językoznawcze dedykowane Profesor Jadwidze Sambor.* Wydział Polonistyki Uniwersytetu Warszawskiego, pp. 164--168.

[14] Sperberg-McQueen, C. M., Burnard, L. (Eds.) 2001. *TEI P4. Guidelines for Electronic Text Encoding and Interchange. XML-compatible edition*. Chicago, Oxford: The Association for Computers and the Humanities (ACH), The Association for Computational Linguistics (ACL), The Association for Literary and Lingustic Computing (ALLC). http://www.tei-c.org/P4X/.

[15] TEI SGML to XML Migration Introduction and Workflow Recommendations. Second Draft, 2003. http://www.tei-c.org/Activities/MI/miw03d.html.

[16] Wall, L., Christiansen, T., Schwartz, R. L. 1996. *Programming Perl*, 2nd Edition., O'Reilly and Associates, Inc., ISBN
1-6592-149-6.

[17] XML Path Language (XPath), version 1.0. World Wide Web Consortium, 1999. http://www.w3.org/TR/REC-xml/.