TEI-XML Exercises 4

Lou Burnard *February 2004*

1 Using TEI-XML: Exercises

In this week's exercises we will be using a different set of sample data files. Download the `samples4.zip` from `http://www.tei-c.org.uk/Talks/OUCS/2004-02/samples4.zip` before you start, and unpack it on your H drive or your desktop. This week, it contains rather a lot of data and also some software so this may take longer than usual. You should finish up with a folder called `samples4` containing the following folders:

Lampeter Contains the Lampeter Corpus: a collection of early Modern English tracts.

Scripts Contains the XSLT and XQuery scripts we will use in parts 1 and 2 of the exercises

Xara-106 Contains the latest release of the software we will use in part 3 of the exercises

The Lampeter Corpus (`http://www.tu-chemnitz.de/phil/english/chairs/linguist/real/independent/lampeter/manual/pages/manual.htm`) is a collection of 120 printed tracts or pamphlets, selected from the holdings of the Founders Library at St Davids College Lampeter, as a project in historical linguistics carried out by researchers from the TU of Chemnitz. It contains relatively complete XML markup of typographic features such as font shifts, and a lot of additional metadata, all encoded in a TEI-conformant way.

The corpus consists of 120 separate files with names like `eca1641.xml` (the two letter prefix indicates the topic of the text, followed by an A or a B, followed by the year of first publication) held in the directory `Lampeter/texts`. Each text is a `<TEI.2>` element, with its own `<teiHeader>` and its own `<text>`. There is also a corpus header file, called `corpus_header.xml` and a file called `driver.xml` which can be used to process the whole corpus together.

1.1 More XSLT

Last week, we did a lot of rendering of TEI XML in HTML for display on the web. You can use the same stylesheets you prepared last week to display the Lampeter corpus as well. (If you want a reminder, take a look at last week's files). This week, we will be doing some more analytic tasks. If you are stumped, completed solutions to these exercises are available in the samples4 folder.

Here's a suggested testing procedure:

1. type the XSLT script you want to run into emacs
2. save it in a file called (e.g.) `test.xsl`*in the samples4 folder*
3. type ESC-! to get a shell buffer
4. type `xsltproc test.xsl Lampeter/driver.xml` into the minibuffer and press RETURN
5. For subsequent invocations of ESC-!, you can get back the same command line and re-edit it using the arrow keys

- Count the number of pages in each text in the corpus.

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">

  <xsl:output method="text"/>

  <xsl:template match="/">
        <xsl:apply-templates select="//TEI.2"/>
  </xsl:template>

<xsl:template match="TEI.2">
Text: <xsl:value-of select="text/@id"/>
Pages: <xsl:value-of select="count(descendant::pb)"/>
</xsl:template>

</xsl:stylesheet>
```

Note the use of the `id` attribute on the `<text>` element to supply the identifier for each corpus text.

- In the header of each text, there are several `<keywords>` elements each of which contains a number of `<term>` elements describing the text in various ways. Make a sorted list of all the `<term>` elements in the corpus.
- Improve the list so that (a) it includes only terms which are within a `<keywords>` element whose `scheme` attribute specifies `lamtop` i.e. topic (b) the identifier of the text is included in square brackets following the term. (Hint: the `<keywords>` element is inside the `<teiHeader>` element, which is a sibling of the `<text>` element, not its parent)
- Make a nice bibliography for the corpus, in the form of an HTML table containing the identifier, the author, the title, and topic keywords for each text. You will find the author information inside the header using the path `teiHeader//person[@role='author']/persName`. Title information is available in the header too, but the one to use here is the `<docTitle>` element in the `<text>` itself: use a path like this `text/front//docTitle`. The titles contains numerous `<lb>` elements to mark line breaks: it is conventional to replace these by | (vertical bar) symbols. Put all the topic terms for each text together in a list separated by semicolons.
- The corpus contains many words in languages other than English. Most of these are tagged using the `<foreign>` element element. Write out a list of all words so tagged, showing for each the language (this is given by the `lang` attribute), the identifier of the text containing it, and its page number within the text (this is given by the `n` attribute of the immediately preceding `<pb>` element). Sort the list alphabetically within each language.
- For extra brownie points, use an `<xsl:keys>` element to translate the coded language value given by the `lang` attribute into the corresponding full name of the language, which is given by a `<language>` element in the corpus header.

1.2 A taste of eXistence

We have set up eXist to run on a local machine at OUCS. Visit `http://spqr.oucs.ox.ac.uk:8080/exist/xquery/xquery.xq` to see the Xquery interface provided by the system. There are many other interfaces and possibilities, but we will try just this one today.

- The simplest kind of Xquery just consists of an Xpath. So, to list the `<bookSeller>` elements in the corpus, just type

```
//bookSeller
```

  To see the results, press the **`Submit Query`** at the bottom of the page. (If you get no results, check that you are looking in the right context: select `db/Lampeter` from the drop down list above the query window). Try any other Xpath you like.

- To list the booksellers in alphabetical order we need to use slightly more complex syntax

```
for $s in //bookseller
order by $s
return $s
```

- Count the number of pages in each text in the corpus.

```
for $t in //TEI.2
let $n := $t//pb
return  <count>
{$t/text/@id}
{count($n)}
</count>
```

  How would you change this to get the list out in descending order of page count?

- List the text number and the latin phrases for every text which has more than two latin phrases in it.

```
for $t in //text
let $lats := $t//foreign[@lang='lat']
where count($lats) > 2
order by count($lats)
return
<latin>
{$lats}
<txt>{$t/@id}</txt>
</latin>
```

- Using eXist's string searching facilities, find all <foreign> elements containing a word beginning ann. Find all <title> elements containing the both the word true and the word account.

1.3 Exercising Xaira

To start Xaira up, proceed as follows:

1. Click on the Xaira icon. On the splash screen, if the **OK** button is available, click on it. If it is not available, click the **Menu** button to open the Server List dialog.
2. In the Server List dialog, click **Add** to add a new corpus, or click on one of the corpus names listed to open it.
3. In the Server Properties dialog for a new corpus, enter the name of your corpus (lampeter) and check the Local check box. Click the **Browse** button to navigate to the location where you installed the corpus files (this should be samples4/Lampeter/) inside which there should be a file called corpus_parameters.xml. Click **Open** to select this, and then **OK** to close the dialog.
4. On returning to the Sever List dialog, you will see your newly created corpus listed. Select its name, and click on the **Set default** button to avoid having to go through this tedious procedure all over again next time. Press the **OK** button to open the corpus.

On opening the corpus, Xaira displays a list of the texts making it up. For more information about a text, click on its name (e.g. eca1697), press the right mouse button, and select Source or Browse from the menu. The former extracts elements from the TEI Header; the latter shows you the whole of the text.

Feel free to experiment with the Xaira interface. We present here a few of the things you can do:

- Type true account into the query box on the tool bar. Try typing other phrases that interest you. Select one of the lines in the display and increase its scope. Press the green button to toggle between single and KWIC views. Select from the Display Format dropdown list to see hits displayed Plain, with some styling, or in XML.
- Open the Word Query dialog (File, New Query, Word). Enter a stem (say bug) into the query box and press Lookup to see a list of words matching the stem together with their frequencies. Select one or more of them, and press the Query button to search for concordance lines for all of them.
- Using the XML query dialog to look for occurrences of the <bookSeller> element. Note that they are displayed in XML form, since this was an XML query. Switch the display format back to Plain, by choosing from the drop down list of available formats on the toolbar. Then Sort by one word to the right.
- Do a search for any high frequency item (for example king, or nature). Open the Analysis window (Query -> Analysis) and see whether this word is evenly distributed across different partitions of the corpus. Several partitions are pre-defined: compare Domain, and LampDec (decade): select the one you want from the list at top right of the Analysis dialog.
- If you'd like to improve on the stylesheet, you can do so using the Query-Stylesheets-Edit command. Try to make Latin text appear in a different colour.
- The Query Builder interface allows you to define complex queries as a series of nodes. See if you can work out how to search for the word True followed by either account or relation within the same <title> element.