All roads lead to .. TEI

Sebastian Rahtz and Lou Burnard

1 Before you start

In this exercise, you will use **Roma**, a web tool available from the TEI web site and also included on your TEI Knoppix CD, to make an XML schema. If you are not using the TEI Knoppix CD, you will need access to the net and a web browser such as IE, Mozilla, or Opera. Once you have your schema you will also need an XML-aware editor to use it (in our case, Gnu Emacs).

Using the Knoppix CD, you may want to choose a different keyboard. This version of Knoppix starts up with a UK keyboard. If you want a different one, click on the national flag in the bottom right corner. If the flag you want doesn't appear for selection, choose the **Configure** option to find it, and then select it.

Our goal is to make a schema which we can use to mark up a document which describes a gravestone. We don't need all of of TEI Lite, much less the full TEI, but we do need bits of various modules. We'll also have to add a couple of new elements, to cope with ideas which the TEI has not covered yet, namely `<death>` (to parallel `<birth>`) and `<nationality>`.

2 Making your own schema

1. Open the Roma application. If you are online, you can point your favourite web browser at `http://tei.oucs.ox.ac.uk/Roma/`, or, if you are working from the CD, at `http://localhost/Roma/` (on the TEI Knoppix CD the recommended web browser is Mozilla Firefox, available on the toolbar; but you can also use Konqueror or even Lynx if you prefer).

2. The Roma start screen allows you to create a new customization, or to upload an existing customization for further work. We will start from scratch, which is the default option. Press the `Submit` button at bottom right of the screen to continue.

The next and subsequent screens show you a row of tabs for acting on your customization (`Save`, `Customize`, `New`, and `Help`), and a row of tabs for each of the major stages or tasks making up a customization (`Modules`, `Add elements`, `Change classes`, `Language`, `Schema`, and `Documentation`. We won't explore all of these in this exercise. By default the Customize your Customization screen is displayed. This allows you to specify a file name and other details for the schema, and also to change the interface language if you wish. For now, accept the defaults. Go to the `Modules` tab to proceed.

The modules screen shows two lists: on the left are all available TEI modules; on the right are the modules currently selected for your schema. You can add modules from the list on the left, and remove modules from the list on the right, by clicking the appropriate word next to the module you wish to operate on.

1. For this exercise, we will need the following modules:

   - `corpus`
   - `figures`
   - `linking`

- namesdates
- textstructure
- msdescription

Click the word add next to each of the modules.

2. The modules chosen contain many more elements than we need, so we now need to remove some of them. Click the name of a module in the *List of selected modules* (the right-most column) to see a list of the elements this module defines.

Each element listed has a name, a radio button indicating whether it is to be included or excluded, a tag name, a description, and a link to a further screen where its attributes are specified. You can toggle inclusion or exclusion of all elements in the list by clicking the appropriate column heading. You can click on Exclude to remove all elements from the module.

Now work down the list clicking the radio button to restore or add the elements needed for this exercise. Remember to press the Submit button when you have finished with each module. Press the Modules or back links to go back to the list of modules.

from the core module delete <add>, <analytic>, <cb>, <cit>, <corr>, <dateRange>, <del>, <distinct>, <expan>, <gap>, <gloss>, <headItem>, <headLabel>, <index>, <l>, <lb>, <lg>, <meeting>, <mentioned>, <milestone>, <orig>, <pb>, <postBox>, <postCode>, <reg>, <series>, <sic>, <sp>, <speaker>, <stage>, <street>, <time>, <timeRange>, and <unclear>

from the msdescription module delete everything except <additional>, <condition>, <depth>, <dimensions>, <height><material>, <msDescription>, <msIdentifier>, <objectDesc>, <physDesc>, <supportDesc>, <surrogates>, and <width>

from the textstructure module delete <argument>, <byline>, <closer>, <dateline>, <div0>, <div1>, <div2>, <div3>, <div4>, <div5>, <div6>, <div7>, <epigraph>, <imprimatur>, <opener>, <salute>, <signed>, and <trailer>

from the figures module delete <formula>

Because we are talking about gravestones, not manuscripts, it would be nice to name the elements accordingly. So now go back to the msdescription module and replace the name msDescription with stoneDescription and msIdentifier with stoneIdentifier.

We now need to define two new elements. Go to the Add Elements tab at the top of the screen. The form for defining a new element is displayed. which allows you to enter a name for your element, to specify the classes of which it is a member, its content model, and its description. Unless you feel confident about working out what the possibilities are for yourself, we suggest that you proceed as follows:

1. make an element ‘death’

2. make it a member of the classes ‘model.personPart’ and ‘model.datePart’
3. give it the content model ‘macro.phraseSeq’
4. supply a description such as ‘describes a death’

Remember to press `Submit` when you have finished. Now we have to add an attribute to contain a normalized date:

1. click on `Change attributes`
2. add an attribute called `date`
3. and set the "Contents" to `data.temporal`
4. give a suitable description
5. Press Submit as usual

When that’s done, repeat the steps above to make `<nationality>`, as a member of ‘model.personPart’, with the same content model of ‘macro.phraseSeq’, and an attribute `target` with datatype `data.pointer`.

We are now ready to generate a schema. Click the *Schema* tab, and then press Submit. Your browser will ask whether you want to save or open the generated file: you should save it into your home directory (the default). Look at the result, if you feel strong, or experiment with other options of the web application.

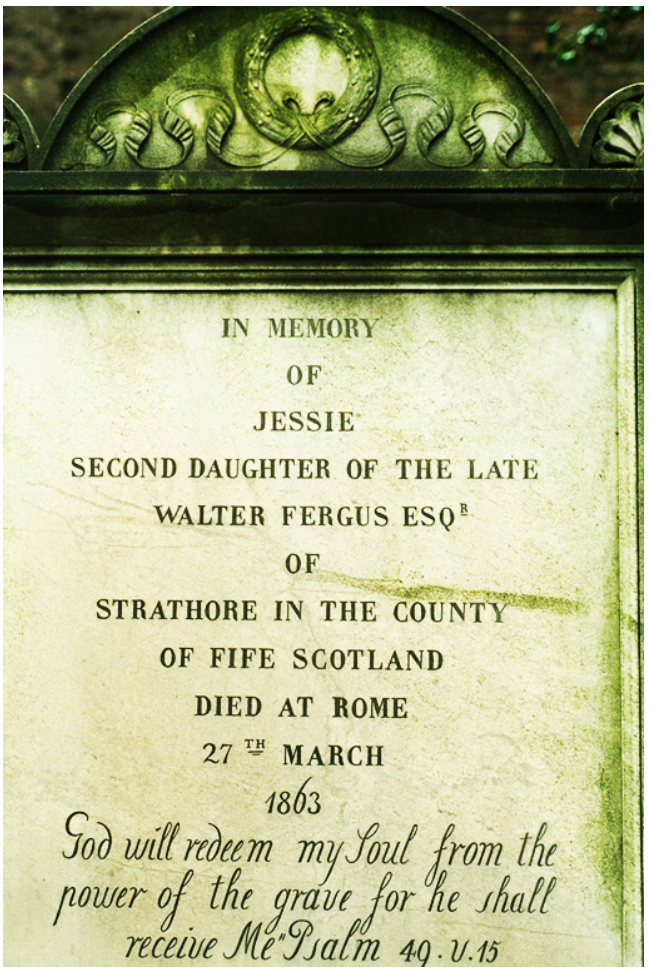
We’ve prepared a little test file which you can use to check that you’ve made your schema correctly. When you open it, however, you will have to tell emacs to use your newly made schema. Proceed as follows, remembering to do all your work in the `samples` directory.

- Open the file `cemtest.xml` with emacs. As this is an XML file emacs will open it in NXML mode, by default.
- Choose Set Schema from the XML menu, and select `File...`, the last option on the submenu this opens.
- Supply the name and location of your schema file (`myTei.rnc` in the directory above `samples`, probably) in the minibuffer at the bottom of the screen. Press Return.
- Emacs will ask whether it should save this information to the `schemas.xml` file. Type yes, and press return.

Now check the mode line—does it say Valid? If it does, try introducing an error in the document. If it does not, click on the word Invalid, and try to see what went wrong.

3 Making a real file

Your next challenge is create a file of your own. Go into Emacs and start a new file called `mycem.xml` and create a document which transcribes the gravestone show below.



You will need to create a `<TEI>` with a `<teiHeader>` and then make a `<text>` containing a `<body>`; within that, make a `<div>` for the inscription, and create `<ab>` elements within that for each line of text. Give the `<TEI>` an xml:id attribute to identify it.

Once your document is valid, you can transform it into an HTML web page by using an XSLT stylesheet. We have prepared a suitable stylesheet for this purpose in the file `cem.xsl`. We have also prepared a simple CSS stylesheet (`cemdisplay.css`) which can be used to display your file without first converting it.

To see what CSS can do, add a stylesheet reference like the following:

```
<?xml-stylesheet type="text/css"
href="cemdisplay.css"?>
```

at the start of your file (*after* the XML declaration if there is one), and then try opening the XML file with Firefox.

To use XSLT, which is probably the most reliable method, you can run a free-standing XSLT processor such as xsltproc and generate a static HTML page from your document. Try typing

```
xsltproc -o
mycem.html cem.xsl mycem.xml
```

at the command prompt (Tools/Shell Command in Emacs). This will generate an HTML file called `mycem.html` which any web browser should be able to display. Alternatively, add a stylesheet reference like the following:

```
<?xml-stylesheet type="text/xsl" href="cem.xsl"?>
```

at the start of your file (replacing the CSS one, if you tried that), and then try opening the XML file with Firefox.

The display using XSL will **not** be the same as when using CSS. The CSS is much simpler.

If have time, enhance the metadata by adding a `<profileDesc>` inside the `<teiHeader>`, something like this:

```
<profileDesc>
      <particDesc>
        <person sex="1">
          <persName>
            <forename>Jessie</forename>
            <surname>Fergus</surname>
          </persName>
          <birth/>
          <death date="1863"/>
        </person>
      </particDesc>
    </profileDesc>
```