Working with ODD

Sebastian RahtzDiXiT

Objectives

The aim of this exercise is to get you to understand a TEI ODD customzation created by Roma, and create one from scratch using the newer features of ODD. You should be able to

1 add documentation to a TEI ODD customization
2 make modifications not possible in Roma
3 express modifications using pure ODD, rather than RELAX NG
4 create a new element

You should start by making a customization in Roma, saving it, and loading the ODD file into oXygen. You'll find it useful to pretty-print this by pressing Meta-Shift-P (the *Format and Indent* button).

Adding documentation to a TEI ODD customization

As we saw in the previous exercise, you can add as much prose as you want before or after the `<schemaSpec>` element, to act as the documentation for your project.

Test this, by writing some useful prose, and processing the file to get documentation using oXygen using Roma.

Modifications not possible in Roma

One of the things you cannot do it Roma is provide documentation for value lists.

1. In your ODD file, change the *@place* attribute of `<add>` to give it a closed `<valList>` containing just the values you want to support (eg just ‘above’ and `<below>`). A bit like this:

```
<elementSpec mode="change" ident="add">
 <attList>
  <attDef ident="place" mode="change">
   <valList type="closed" mode="replace">
    <valItem ident="above">
     <desc>ABOVE</desc>
    </valItem>
    <valItem ident="below">
     <desc>BELOW</desc>
    </valItem>
   </valList>
  </attDef>
 </attList>
</elementSpec>
```

2. Look carefully here at the use of *@mode* and make sure you understand why the `<valList>` is in ‘replace’ mode.
3. Generate documentation and check you can see your changes.
4. Generate a RELAX NG schema, and use it in oXygen to edit a file. Do you see the documentation values you specified?
5. do the same thing, providing a `<valList>` on another element. You will have to decide whether to use ‘replace’ or ‘add’ mode.

Suppose in your prose section you want to discuss a particular group of elements? add this to your ODD:

```
a
<specList>
 <specDesc key="p"/>
 <specDesc key="p" atts="+"/>
 <specDesc key="p" atts="xml:id"/>
 <specDesc key="valList"/>
 <specDesc key="valList" atts="type"/>
</specList>
```

make documentation and see the effect of each of these `<specDesc>` elements.

Expressing modifications using pure ODD

So far we have not needed to tinker much with content models. But now lets radically change the model of `<div>`, with

```
<elementSpec mode="change" ident="div">
 <content mode="replace">
  <alternate>
   <alternate maxOccurs="unbounded"
    minOccurs="1">
    <classRef key="model.pLike"/>
    <classRef key="model.graphicLike"/>
   </alternate>
   <sequence maxOccurs="unbounded"
    minOccurs="0">
    <classRef key="model.divLike"/>
   </sequence>
  </alternate>
 </content>
</elementSpec>
```

Do you understand what is happening here? Generate a schema, and create a document using it. What are you prompted for within a `<div>`?

Now you try. Alter <p> so that it only contains transcriptional elements (model.pPart.transcriptional), or lists, but not both. Look at the TEI Guidelines to check the model classes.

Make a schema and a document. How is it different from <p> in normal use? What is missing?

Create a new element

Sometimes one just cannot resist adding a new element to the TEI. Your task is to build one from scratch, using all the components of the ODD language.

- It should be called `<alien>`
- It should be allowed to appear in the same places as transcriptional elements
- It should have a *@type* attribute and a *@place* attribute
- It should be able to contain plain text, and the global elements like page and line breaks
- It must not occur inside an `<div>` with a *@type* of 'verse' (you'll need a constraint)
- It needs an example and a description

(I leave it to you to work out what its semantics are)

Check this works in a schema and documentation.

But that wasn't quite right - you are not supposed to add to the TEI. Change your element to be in the namespace `http://www.dixit.eu/ns/` (add `ns="http://www.dixit.eu/ns/"` to the `<elementSpec>`). NOTE: you must add code like

```
<constraint>
 <s:ns prefix="a"
  uri="http://www.dixit.eu/ns/"/>
</constraint>
```

declaring the namespace to Schematron and providing a prefix, inside your `<constraintSpec>`, or the schema generation will fail. See how it looks now in the schema.