One Document Does it all

Lou Burnard and Sebastian Rahtz

TEI

October 2005
The TEI Guidelines, its DTD, and its schema fragments, are all produced from a single XML resource containing:

1. Descriptive prose (lots of it)
2. Examples of usage (plenty)
3. Formal declarations for components of the TEI Abstract Model:
   - elements and attributes
   - modules
   - classes and macros
4. We call this resource an ODD (One Document Does it all).
The TEI scheme can only be used by customizing it and customizations are also expressed in the ODD language. For example:

```xml
<schemaSpec ident="myTEIlite">
  <desc>This is TEI Lite with simplified heads</desc>
  <moduleRef key="tei"/>
  <moduleRef key="core"/>
  <moduleRef key="textstructure"/>
  <moduleRef key="header"/>
  <moduleRef key="linking"/>
  <elementSpec ident="head" mode="change">
    <content>
      <rng:text/>
    </content>
  </elementSpec>
</schemaSpec>
```

produces something like TEI Lite, with a slight change
ODD processors

- We maintain a library of XSLT scripts that can generate
  - The TEI Guidelines in canonical TEI XML format
  - The Guidelines in HTML or PDF
  - RelaxNG, DTD, or W3C schema fragments
- The same library is used by the customization layer to generate
  - project-specific documentation
  - project-specific schemas
  - translations into other (human) languages
- We use eXist as a database for extracting material from the P5 sources
The TEI abstract model sees a markup scheme (a schema) as consisting of a number of discrete modules, which can be combined more or less as required.

A schema is made by combining references to modules and optional element over-rides or additions.

Each element declares the module it belongs to: elements cannot appear in more than one module.

Each module extends the range of elements and attributes available by adding new members to existing classes of elements, or by defining new classes.
The TEI class system

- Class membership can do two distinct things for an element:
  1. give it some attributes
  2. allow it to join a ‘club’

- Content models reference ‘clubs’ rather than specific elements (wherever possible)

- Content models are named patterns, distinct from element names

- (There are also special named patterns for common content models such as `macro.phraseSeq`)
Expression of TEI content models

Within the class system, TEI elements have to be defined using some language notation; choices include:

1. using ‘raw’ XML DTD language
2. using W3C Schema language
3. using the Relax NG schema language
4. inventing an entirely new abstract language for later transformation to specific schema language

We chose a combination of 3 and 4 — using our abstract language, but switching to Relax NG for content modelling.
Why that combination?

- Expressing constraints in XML language is too attractive to forego.
- There is a clamour for better datatyping than DTDs have.
- The schema languages are so good, it is silly to reinvent them.
- But we like our class system and literate programming.
DTD vs Relax NG vs W3C Schema

- DTDs are not XML, and need specialist software
- W3C schema is not consistently implemented, is poorly documented, and looks over-complex
- Relax NG on the other hand...
  - uncluttered design
  - good documentation
  - multiple open source 100%-complete implementations
  - ISO standard
  - useful features for multipurpose structural validation
  - Compelling leadership (can James Clark do wrong?)

No contest...
What does an ODD look like?

```xml
<elementSpec module="spoken" ident="pause">
  <classes>
    <memberOf key="model.divPart.spoken"/>
    <memberOf key="att.timed"/>
    <memberOf key="att.typed"/>
  </classes>
  <content>
    <rng:empty/>
  </content>
  <attList>
    <attDef ident="who" usage="opt">
      <gloss>A unique identifier</gloss>
      <desc>supplies the identifier of the person or group pausing. Its value is the identifier of a <gi>person</gi> or <gi>persGrp</gi> element in the TEI header.</desc>
      <datatype>
        <rng:ref name="data.pointer"/>
      </datatype>
    </attDef>
  </attList>
  <desc>a pause either between or within utterances</desc>
</elementSpec>
```
... from which we generate

element pause   pause.content, pause.attributes
pause.content = empty
pause.attributes =
    att.global.attributes,
    att.timed.attributes,
    att.typed.attributes,
    att.ascribed.attributes,
    [ a:defaultValue = "pause" ] attribute TEIform text ?
model.divPart.spoken |= pause
att.timed |= pause
att.typed |= pause
att.ascribed |= pause
.. or

```xml
<!ELEMENT %n.pause; %om.RR; EMPTY>
<!ATTLIST %n.pause;
  %att.global.attributes;
  %att.timed.attributes;
  %att.typed.attributes;
  %att.ascribed.attributes;
  TEIform CDATA 'pause' >
<!ENTITY % model.divPart.spoken
  "%x.model.divPart.spoken; %n.event; | %n.kinesic;
  | %n.pause; | %n.shift; | %n.u;
  | %n.vocal; | %n.writing;">```
... and, indeed, to

<table>
<thead>
<tr>
<th><strong>pause</strong></th>
<th>a pause either between or within utterances.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Class</strong></td>
<td><code>model.divPart.spoken att.timed att.typed att.ascribed</code></td>
</tr>
</tbody>
</table>
| **Declaration** | `element pause
{    
  att.global.attributes,  
  att.timed.attributes,  
  att.typed.attributes,  
  att.ascribed.attributes,  
  empty
}
` |
| **Attributes**  | Global attributes and those inherited from [`att.timed att.typed att.ascribed` ] |
| **Example**     | `<pause dur="PT42S" type="pregnant"/>` |

142. **Formal Definition**

spoken
A more complex example

```xml
<elementSpec module="corpus" ident="birth">
<gloss>Birth details</gloss>
<desc>contains information about a person’s birth, such as its date and place.</desc>
<class>
  <memberOf key="model.personPart"/>
</class>
<content>
  <rng:ref name="macro.phraseSeq"/>
</content>
<attList>
  <attDef ident="date" usage="opt">
    <desc>specifies the date of birth in an ISO standard form (yyyy-mm-dd).</desc>
    <datatype>
      <rng:ref name="data.temporal"/>
    </datatype>
  </attDef>
</attList>
</elementSpec>
```
Which produces …

The TEI Guidelines

<birth>

<table>
<thead>
<tr>
<th>birth</th>
<th>()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Клас</td>
<td>model.personPart</td>
</tr>
</tbody>
</table>
| Декларация | element birth {
| | att.global.attributes,
| | attribute date { data.temporal }?,
| | macro.phraseSeq |
| Атрибуты | (Освен глобалните атрибути и атрибуитите, наследени от) |
| date | Състояние: Незадължителен |
| | Тип данни: data.temporal |
| Пример | <birth>Before 1920, Midlands region.</birth> |
| Пример | <birth date="1960-12-10">In a small cottage near <name type="place">Aix-la-Chapelle</name>, early in the morning of <date>10 Dec 1960</date></birth> |
| Забележка | Dates and place names, if included in the content of this element, should in general be tagged using the <date> and <name> elements respectively. If the additional tagset for Names and Dates is in use, the more specific elements defined by that tagset may be used as an alternative. |
And some XSD for a change . . .

```xml
<xs:element name="birth">
  <xs:annotation>
    <xs:documentation>(Birth details) contains information about a person’s birth, such as its date and place.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="ns1:birth.content">
        <xs:attributeGroup ref="ns1:birth.attributes"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:complexType name="birth.content">
  <xs:complexContent>
    <xs:extension base="ns1:macro.phraseSeq"/>
  </xs:complexContent>
</xs:complexType>
```
Customizing the TEI

The TEI has over 20 modules. A working project will:

- Choose the modules they need
- Probably narrow the set of elements within a module
- Probably add local datatype constraints
- Possibly add new elements
- Possibly localize the names of elements
We can do this in ODD

A simple selection of modules

```xml
<schema>
  <moduleRef key="tei"/>
  <moduleRef key="core"/>
  <moduleRef key="header"/>
  <moduleRef key="textstructure"/>
  <moduleRef key="linking"/>
</schema>
```
More interestingly..

```xml
<schema>
  <moduleRef key="header"/>
  <moduleRef key="verse"/>
  <elementSpec ident="soundClip">
    <classes>
      <memberOf key="tei.data"/>
    </classes>
    <attList>
      <attDef ident="location">
        <desc>supplies the location of the clip</desc>
        <datatype>
          <rng:ref name="data.pointer"/>
        </datatype>
      </attDef>
    </attList>
    <desc>includes an audio object in a document.</desc>
  </elementSpec>
  <elementSpec ident="head" mode="change">
    <content>
      <rng:text/>
    </content>
  </elementSpec>
</schema>
```
Uniformity of description

- modules, elements, attributes, value-lists are treated uniformly
- each has an identifier, a gloss, a description, and one or more equivalents
- each can be added, changed, replaced, deleted within a given context
- for example, membership in the \texttt{att.type} class gives you a generic \texttt{type} attribute, which can be over-ridden for specific class members
Overriding a value-list

```xml
<elementSpec ident="list" module="core">
  <classes>
    <memberOf key="att.typed"/>
  </classes>
  <attDef ident="type" mode="replace">
    <valList type="closed">
      <valItem ident="ordered">
        <gloss>Items are ordered</gloss>
      </valItem>
      <valItem ident="bulleted">
        <gloss>Items are bulleted</gloss>
      </valItem>
      <valItem ident="frabjous">
        <gloss>Items are frabjous</gloss>
      </valItem>
    </valList>
  </attDef>
</elementSpec>
```
Ontological mapping

The `<equiv>` element supplies a URI which identifies an equivalent concept (not a name) in some externally-defined ontology, e.g.

- ISO data category registry
- CIDOC conceptual reference model
- Wordnet
Roma: generating validators for the TEI

**Modules**

<table>
<thead>
<tr>
<th>Module name</th>
<th>A short description</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>add analysis</td>
<td>Simple analytic mechanisms</td>
<td></td>
</tr>
<tr>
<td>add certainty</td>
<td>Certainty and uncertainty</td>
<td></td>
</tr>
<tr>
<td>add core</td>
<td>Elements common to all forms of the TEI</td>
<td></td>
</tr>
<tr>
<td>add corpus</td>
<td>Header extensions for corpus texts</td>
<td></td>
</tr>
<tr>
<td>add declares</td>
<td>Feature system declarations</td>
<td></td>
</tr>
<tr>
<td>add dictionaries</td>
<td>Printed dictionaries</td>
<td></td>
</tr>
<tr>
<td>add drama</td>
<td>Performance texts</td>
<td></td>
</tr>
<tr>
<td>add figures</td>
<td>Tables, formulae, and figures</td>
<td></td>
</tr>
<tr>
<td>add gaji</td>
<td>Character and glyph documentation</td>
<td></td>
</tr>
<tr>
<td>add header</td>
<td>The TEI Header</td>
<td></td>
</tr>
<tr>
<td>add iso-fs</td>
<td>Feature structures</td>
<td></td>
</tr>
<tr>
<td>add linking</td>
<td>Linking, segmentation and alignment</td>
<td></td>
</tr>
<tr>
<td>add mdescription</td>
<td>Manuscript Description</td>
<td></td>
</tr>
<tr>
<td>add namesdates</td>
<td>Names and dates</td>
<td></td>
</tr>
<tr>
<td>add nets</td>
<td>Graphs, networks and trees</td>
<td></td>
</tr>
<tr>
<td>add spoken</td>
<td>Transcribed Speech</td>
<td></td>
</tr>
<tr>
<td>add tagdocs</td>
<td>Documentation of TEI modules</td>
<td></td>
</tr>
<tr>
<td>add tei</td>
<td>Structural declarations for the TEI</td>
<td></td>
</tr>
</tbody>
</table>
## Roma: generating validators for the TEI

### Change attribute classes

<table>
<thead>
<tr>
<th>Class name</th>
<th>Description</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>att.TEIform</td>
<td>defines an attribute (TEIform) common to all tags in the TEI scheme, and recommended for all user-defined extensions.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.analytic</td>
<td>defines a set of attributes for associating specific analyses or interpretations with appropriate portions of a text, which are enabled for all elements when the additional tag set for simple analysis is selected.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.ascribed</td>
<td>elements representing speech ascribed to a speaker.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.datable</td>
<td>defines the set of attributes common to all elements that contain datable events.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.datePart</td>
<td>attributes for component elements of temporal expressions involving dates and time.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.declarable</td>
<td>groups elements which may be independently selected (using the special purpose decls attribute) from a candidate list of declarations within a TEI header.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.declaring</td>
<td>groups elements which may be independently associated with a particular declarable element within the header, thus overriding the inherited default for that element.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.divLike</td>
<td>defines a set of attributes common to all elements which behave in the same way as divisions.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.editLike</td>
<td>elements which carry attributes describing editorial interventions.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.enjamb</td>
<td>groups elements bearing the enjamb attribute.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.entryLike</td>
<td>groups the different styles of dictionary entries.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>att.global</td>
<td>defines a set of attributes common to all elements in the TEI encoding scheme.</td>
<td>changeAttributes</td>
</tr>
<tr>
<td>Name</td>
<td>Model classes</td>
<td>Attribute classes</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Omit model.addrPart, model.dateLike, model.editorialDeclPart, model.frontPart.drama</td>
<td>att.TEIform, att.datePart, att.editLike, att.global.linked, att.measured, att.pointing</td>
</tr>
<tr>
<td></td>
<td>model.bibLike, model.datePart, model.encodingPart</td>
<td>att.analytic, att.declarable, att.enjamb, att.indented, att.measured, att.pointing.group</td>
</tr>
<tr>
<td></td>
<td>model.bibLike, model.datePart, model.entryLike</td>
<td>att.ascribed, att.declaring, att.entryLike, att.interpLike, att.naming, att.ptrLike.form</td>
</tr>
<tr>
<td></td>
<td>model.blockLike, model.divPart</td>
<td>att.dateable, att.divLike, att.entryLike, att.lexicographic, att.personal, att.rdgPart</td>
</tr>
<tr>
<td></td>
<td>model.catDescPart, model.divPart.spoken</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.choicePart, model.divPart.stage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.choicePart, model.divPart.verse</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.complexVal, model.divWrapper</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.complexVal, groups elements which can occur at the start of any division class element.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.complexVal, model.entryParts.top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.entryParts.top, model.entryParts.top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.entryParts.top, model.entryParts.top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.entryParts.top, model.entryParts.top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.entryParts.top, model.entryParts.top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.entryParts.top, model.entryParts.top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.entryParts.top, model.entryParts.top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.entryParts.top, model.entryParts.top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.entryParts.top, model.entryParts.top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>model.entryParts.top, model.entryParts.top</td>
<td></td>
</tr>
</tbody>
</table>

```
<content
xmlns:rng="http://relaxng.org/ns/structure/1.0">
</content>
```