

Literate Documentation for XML: TEI ODD - METS

Kevin Reiss – Mina Rees Library, City University of New York Graduate Center

Synopsis

An experimental system for XML schema creation and documentation using an extended version of the TEI P5 One Document Does it All (ODD) literate programming module.

This project experiments with constructs that allow the markup language designer to consistently document a XML schema using natural language. This approach can help bridge the gap between current XML documentation practices until tools that can specify machine-readable semantics for XML appear in the future.

Current State of XML Documentation

Current practices for the documentation of XML schema have not progressed much beyond the recommendations made in Eve Maler and Jeanne El Andaloussi's (1996) text *Developing SGML DTD's: From Text to Model to Markup*. That volume recommends an approach to DTD design called document type modeling to SGML DTD designers that, if followed, produces both a reusable and customizable DTD and a well-structured reference manual that documents that DTD.

In fact in many cases the documentation provided with schema is much more inconsistent and haphazard than that recommended by Maler and El Andaloussi. To an instance author working with a text the clear communication of the markup language designer's intentions is critical in making decisions on how to properly apply markup.

Markup Semantics

The problem of interpreting the designer's intentions is compounded by the fact that XML lacks a formalized, machine-readable knowledge representation technology that would allow a designer to explicitly represent in unambiguous fashion the semantics of a XML application (Renear et al., 2002). Researchers have been experimenting with knowledge representation systems that can infer and represent markup semantics via machine-processing.

Common XML Problems/Confusions

These researchers have identified a number of problematic constructs that consistently appear in XML applications whose clear communication is particularly important for effective use of an application (Sperberg-McQueen, et. al, 2002) (Dubin et al., 2002)(Dubin et al., 2003) (Dubin, 2003) . This project experiments with four of these:

- Issues of Propagation - does this property apply to?
- Constraints (Co-occurrence/Dependence/Exclusion)
- Overloaded Parent/Child Relationships among elements
- Clarification of ID/IDREF Relationships

This project does not seek to address the problem of representing these issues in machine-readable form. It seeks only to provide a mechanism that enables a schema author to reliably and concisely represent these issues in the prose documentation accompanying a given schema.

TEI ODD

The TEI ODD is the most fully realized XML literate programming tool currently available. It is well-structured, well-organized, supports customization, and already produces excellent prose documentation for a schema. The version of ODD included in P5 can also be used as a general purpose documentation tool to generate a validating schema and prose documentation for any type of XML application (Burnard and Rahtz, 2004).

Semantic Checklist

This project takes advantage of these characteristics and extends the TEI ODD to include what could be termed a "semantic checklist" for the author to answer the sorts of questions proposed by the set of common XML problems listed earlier.

Checklist Goals

- Improve prose documentation quality
- Ensure consistent use of language in documentation
- Create schema documentation that is usable by authoring tools – better diagnostics, editing suggestions, etc.
- Improve validating schema quality and readability

Extending the ODD

All extensions were implemented using the ROMA tool available on the TEI website. The starting TEI template for the customization was the "TEI for authoring ODD" created by Sebastian Rahtz.

XML Issue	Extension
Propagation	Added the optional attribute "prop" to <attDef> content model. Can take the values "self" "children" "children-until-def".
Constraints	The template provided for authoring ODD by ROMA already allows this by allowing for the inclusion of optional schematron rules within ODD content model declarations.
Parent/Child	Added the optional attribute "relationship" to <elementSpec> and <classSpec> that takes the values "isPartOf" "describes" user defined. This attribute is defined as a "semi" attribute list, this allows authors to add their own classification as appropriate.
ID/IDREF(S)	Added the optional attribute "references" to the <attDef> content to be used with attributes that take an IDREF value. "references" contains the generic identifier of the element that the IDREF value points.

Test Scenario: METS/METS Profile

Serving a similar role that the TEI does for texts, the Metadata Encoding and Transmission Standard (METS) is a loosely-defined format meant to encode metadata for any type of digital object. The METS W3C XML Schema is an open schema divided into seven different sections that are roughly analogous to the TEI's modules.

A METS Profile seeks to restrict a class METS documents in the same fashion that a validating TEI schema produced through ROMA does. However, the profile only provides a prose description of these restrictions. Consider the following example taken from the METS Profile Website:

```
<requirement ID="structMap5">
  <p>An <fptr> element must either 1) directly point to a <file> element via its FILEID attribute; or 2) contain an <area> element that points to a <file> element; or 3) contain a <seq> element comprising multiple <area> elements that point to the relevant <file> elements. METS documents implementing this profile must not use the <par> element. <structMaps> of "physical" and "mixed" TYPEs must not use either the <par> or <seq> elements.</p>
</requirement>
```

Consider a part of this declaration formulated as an extended ODD instance and you can begin to see how the extended ODD could improve both the prose documentation and the validating schema for a XML application..

```
<elementSpec ident="structMap" module="sMap">
  <content>
    <sch:rule context="structMap">
      <sch:assert test="not(ancestor::par)">Par elements N/A</sch:assert>
    </sch:rule>
    <sch:rule context="structMap/@TYPE='physical' | structMap/@TYPE='mixed'">
      <!-- omitted -->
    </sch:rule>
  </content>
</elementSpec>
```

```
<elementSpec ident="fptr" module="sMap"
  relationship="isPartOf"><!-- parent is mets:div -->
  <!-- omitted -->
  <attList>
    <attDef ident="FILEID" usage="opt" references="file">
      <!-- omitted -->
    </attDef>
  </attList>
</elementSpec>
```

Potential Problems

- The checklist seems brittle in this experimental stage
- All extensions are optional
- Checklist may not be suited to general purpose XML use

Future Work

- Modify TEI ODD stylesheets to:
 - Process semantic checklist customizations
 - Partially automate authoring of schematron rules
- Prepare a full definition of METS using the ODD
- Process a METS profile with ODD embedded in the reqs section
- Test checklist with other XML applications

References & Websites

See handout provided