Extreme Markup Languages

Relax NG with Son of ODD

Lou Burnard
Assistant Director
Oxford University Computing Services
Oxford
United Kingdom
lou.burnard@oucs.ox.ac.uk
http://www.oucs.ox.ac.uk/

Sebastian Rahtz
Information Manager
Oxford University Computing Services
Oxford
United Kingdom
sebastian.rahtz@oucs.ox.ac.uk
http://www.oucs.ox.ac.uk/

Keywords: TEI (Text Encoding Initiative); ODD; RelaxNG

Abstract

The Text Encoding Initiative is using literate schema design, as instantiated in the completely redesigned ODD system, for production of the next edition of the TEI Guidelines. Key new aspects of the system include support of multiple schema languages; facilities for interoperability with other ontologies and vocabularies; and facilities for user customization and modularization (including a new web-based tool for schema generation). In this paper, we try to explain the rationale behind the ongoing revision of the TEI Guidelines and how the new tools developed to go with it are taking shape. We describe the mechanics by which the new ODD system delivers its promised goals of customizability and extensibility, while still being a good citizen of a highly inter-operable digital world.

1 Introduction

The Text Encoding Initiative began in 1987 as a research project with a very specific goal: the production of a set of Guidelines for the mark up of (largely) literary and linguistic material. That task was accomplished by a series of publications, from the first release of the Guidelines for the Encoding of Machine Readable Texts known as P1 (**[1]**) in 1990, with subsequent revised editions in 1993 and 1994 (**[2]**). In 2000 both the text of the Guidelines itself and the DTD it documented were revised to use XML, rather than the SGML formalism used hitherto. This edition, known as P4 (**[3]**) is the current stable release of the Guidelines. It describes nearly 500 textual elements grouped into classes and modules or 'tagsets', and is maintained as a single XML document.

The TEI has been, and remains, a major influence in the digital library and in linguistic computing generally. It is probably one of only two or three DTDs which any XML practitioner can name, and new applications for it continue to appear, nearly fifteen years after its first publication. The maintenance of the Guidelines is now in the hand of a non-profit consortium of over fifty members drawn from academic institutions, publishers, research centres and projects world wide. It is planned to open up the process of maintenance still further following the establishment of a Source Forge project ([4]) from which the TEI sources will be available by the end of 2004.

In planning the next release of the TEI Guidelines (P5), our major objectives included a concern to integrate relevant work already achieved by others, as well as to expand coverage into areas not currently addressed. We also wanted to take the opportunity of carrying out a kind of internal audit — sweeping out some dusty corners, removing some dead ends and generally spring cleaning the establishment for another decade of service.

The intellectual territory occupied by the TEI has become a much busier place since 1987. A new edition of the Guidelines therefore must fit into a much larger and more closely integrated digital world. The original TEI predates not only the World Wide Web, but also the whole family of XML and associated W3C standards. It predates the successful establishment of Unicode, as well as the arrival of several other well-supported and well-defined specialist markup vocabularies sich as MathML ([5]), SVG ([6]), or DocBook([7]. It predates the emergence of abstract metadata schemes such as METS ([8]) and EAD ([9]), to say nothing of entirely different conceptual models and ontologies. That does not of course imply that the TEI has nothing to say on the subject of hypertext linking or character encoding, or that it does not suggest markup vocabularies for maths, graphics, document metadata, or conceptual models. But it does mean that what it says may well be in need of revision.

Some parts of TEI P4 were successfully experimental in that the ideas they proposed were subsequently integrated into other standards or simply became part of the intellectual landscape. Examples include the TEI's extended pointer syntax, of which the current Xpointer recommendation is a pure subset; or the TEI's recommendations for encoding language corpora, which, rebranded as the ‘Corpus Encoding Standard’ ([10]), have become a de facto standard in the language resource community. Other notions discussed in the TEI Guidelines have been influential but remain experimental; in particular some problems which might be termed ‘frequently answered questions’ such as those associated with the need to synchronize or aggregate discontinuous segments, using standoff or other techniques. And finally there are recommendations in the TEI which were experimental at the time they were first made, and have either remained so or been subsquently overtaken by events. Examples include the Writing System Declaration, invented in P1 as a means of overcoming the absence of any universal standard for character encoding — a gap which Unicode now supplies; an complex set of recommendations for the representation of linguistic feature structures (which has since been substantially revised and simplified to become a draft ISO standard); and recommendations for the encoding of terminological databases which have since been completely overtaken by other standardization efforts.

A revision of the TEI scheme also seems overdue if we wish to avoid its fossilization. The tools and techniques available to the present generation of text encoders are already quite different from those available fifteen years ago, and we may expect that process of change to continue. We must therefore be able to ensure that the TEI can continue to evolve in response to those needs, and also in response to collateral changes in available technologies. This involves social and organizational considerations, but also evolution of a technical architecture which is responsive to and facilitates change. We address that architecture in the remainder of this paper.

2 Literate programming ODD-style

We have reported elsewhere on the origins and purpose of the ODD system used to produce the TEI Guidelines, its DTD, and its schema fragments ([11]); only a brief summary is given here. The basic idea has not changed since it was formulated by Burnard and Sperberg-McQueen over lunch in a fish restaurant in Bergen in 1998, though many of the details have been revised. Essentially, the Guidelines are now maintained as a single XML resource containing a lot of descriptive prose and many examples of usage, combined with formal declarations for the following key components of the TEI Abstract Model:

- elements and attributes
- modules
- classes and macros

Although we call this resource an ODD (One Document Does it all), it is currently instantiated as several hundred XML mini-documents, which are managed by the Perforce ([12]) content management system to facilitate access for both

European and North American editors. We have recently carried out some experiments in using the eXist ([13]) XML database system as an alternative way of managing its complexity. This ODD resource is processed by a suite of XSLT scripts written by Sebastian Rahtz, which (inter alia) can generate

- The text of the Guidelines in canonical TEI XML format
- The text of the Guidelines in HTML or PDF
- RelaxNG, DTD, or W3C schema fragments

Crucially, this same library is used by the new customization layer we describe further below. This enables us to generate sets of documentation and schemas which have been tailored to the needs of particular projects, for example by removing all but a few elements, by adding or renaming elements, and even by providing translation into other (human) languages. We discuss these features further in what follows.

The TEI has always been agnostic about the choice of a formal representation language. In its original design documents there is a clear intention to define an encoding model at a higher level of abstraction than that provided by current markup languages: ‘The Text Encoding Initiative will develop a conforming SGML application, if it can meet the needs of researchers by doing so. Where research needs require constructs unavailable with SGML, however, research must take precedence over the standard.’ ([14]). The ease with which the whole TEI scheme was migrated from SGML to XML demonstrated the long term benefit of this approach, as does the ease with which existing TEI SGML applications can be moved to XML[1]. Despite the availability of more sophisticated modelling languages for document grammars, therefore, the TEI has remained cautious, preferring to define its own high-level abstract model which can then be mapped to the schema language of the day.

In the ODD model, a markup scheme is defined by a schema, which may subsequently be expressed using the concrete syntax of an XML DTD, a RelaxNG schema, or a W3C Schema. An ODD schema consists of references to a number of discrete modules, combined more or less as required; the distinction between ‘base’ and ‘additional’ tagsets in earlier versions of the Guidelines has not been carried forward into P5. The modules referenced comprise declarations of particular elements and attributes, which can be combined with further declarations given explicitly in the schema for customization purposes.

The module is really a convenience grouping of elements. Each element declares the module to which it belongs; it follows therefore that elements cannot appear in more than one module. A reference to a module extends the range of elements and attributes available in other declarations by the mechanism of adding new members to existing classes of elements, or by defining new classes.

The TEI class system provides an important way of reducing complexity of the overall system. In the original TEI scheme, classes were implemented as parameter entities, which led to some non-obvious constraints on their usage (for example, the requirement that classes be defined multiple times and in the right order as their membership changes). In the current system, the use of RelaxNG patterns to express classes greatly simplifies the process of class modification.

The TEI class system is used for two distinct purposes. By declaring itself a member of a class, an element

1. acquires any attributes defined for the class
2. declares itself a member of a named ‘club’

As far as possible, the content model for all elements makes reference to ‘clubs’ rather than to specific elements, thus facilitating their customization. In schema terms, each element's content is defined by a pattern, initially empty, with which new elements can be interleaved as their classes are populated. In addition, the model permits the usage of specific special named patterns (which may themselves reference clubs) for common content models such as macro.phraseSeq — a combination of any members of the phrase class with text.

At some point, however, content models, like other aspects of a TEI schema, have to be mapped to an appropriate construct in the target schema language. For the most part this is done by processing the ODD declarations with an appropriate stylesheet. Irrespective of whether they contain references to classes or to specific elements, content models for elements present a particular problem, in that existing constraint languages already provide a powerful and expressive syntax for this purpose. Should we therefore adopt a specific constraint language, possibly extending it with ODD-specific features, or should we re-invent that particular wheel? Turned inside out, this problem also takes a form known as the Durand Conundrum: if we decide to adopt a modern constraint language (such as RelaxNG) to express content model constraints, should we not also use the same constraint language to express all the other components of the ODD declarations?

In P4 and previous versions, the TEI content models were expressed as raw fragments of XML DTD language. Our experience of the difficulty of parsing and generating this (gained during the XML conversion of P4) made us very reluctant to continue with it. On the other hand, the idea of inventing an entirely new abstract language for later transformation to some schema language seemed like a profligate waste of our limited resources. We therefore chose to revise the ODD system to use a specific constraint language for the purposes of modelling contents and datatypes only. A modern schema language would give us the benefits of being able to express constraints in XML and therefore process them along with the other parts of the ODD documentation; it would also give us access to important new facilities such as name spacing and better datatyping. Of the available schema languages, we chose to use RelaxNG rather than W3C Schema for a variety of usability reasons. At present, for our purposes, the W3C schema language seemed over complex, suffering from both inconsistent inmplementations and rather poor documentation. RelaxNG, on the other hand, has an uncluttered design and good documentation; it is, moreover, an ISO standard, for which there already exist multiple open source 100%-complete implementations.

A full definition of the ODD vocabulary will (for the first time) be included in P5, and a draft of the relevant chapter is now available at [15]. We give below a simple example to demonstrate some of its essentials:

Figure 1

```
<elementSpec module="spoken" ident="pause"
   xmlns:rng="http://www.relaxng.org/ns/1.0"/>
 <classes>
   <memberOf key="tei.comp.spoken"/>
   <memberOf key="tei.timed"/>
   <memberOf key="tei.typed"/>
 </classes>
 <content>
   <rng:empty >
 </content>
  <attList>
    <attDef ident="who" usage="opt">
    <datatype><rng:data type="IDREF"/></datatype>
      <valDesc>A unique identifier</valDesc>
      <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>
    </attDef>
 </attList>
 <desc>a pause either between or within utterances.</desc>
</elementSpec>
```

This is an element specification, which provides all that is needful for an ODD processor to generate declaration and documentation of the TEI element <pause>, which forms a part of the spoken module, and is a member of three classes (tei.comp.spoken, tei.timed, and tei.typed). In addition to the attributes it gains by membership of the latter two classes (which are not specified here, but in the specification of those classes), this element has an additional who attribute, the specification of which is given by an <attDef> element. Parts of this specification use declarations from the RelaxNG namespace to supply a content model for the element, and also to supply a datatype for the attribute. Finally, there is a prose description of the function of the element itself, for use elsewhere in the formal documentation.

It is relatively simple to see how this can be used to generate outputs like the follow:

Figure 2

```
pause = element pause { pause.content }
pause.content =
  empty,
  tei.global.attributes,
  tei.comp.spoken.attributes,
  tei.timed.attributes,
  tei.typed.attributes,
  pause.attributes.who,
  pause.newattributes,
  [ a:defaultValue = "pause" ] attribute TEIform { text }?
pause.newattributes |= empty
tei.comp.spoken |= pause
tei.timed |= pause
pause.attributes.who =
  attribute who { pause.attributes.who.content }?
pause.attributes.who.content = xsd:IDREF
```

which can then be further processed to produce the DTD version:

Figure 3

```
<!ENTITY % pause 'INCLUDE' >
<![ %pause; [
<!ELEMENT %n.pause; %om.RR; EMPTY>
<!ATTLIST %n.pause;
  %tei.global.attributes;
  %tei.timed.attributes;
  %tei.typed.attributes;
  who  IDREF #IMPLIED
  TEIform CDATA 'pause'  >

<!ENTITY % tei.comp.spoken "%x.tei.comp.spoken;
%n.event; | %n.kinesic; | %n.pause; | %n.shift;
| %n.u; | %n.vocal; | %n.writing;">
```

while figure 1 shows how the same document appears when it is rendered as part of the reference documentation

Figure 4

[Graphic entity *pausePng* — pausedoc.png]

As mentioned above, these various outputs from the ODD documentation are all generated from a common library of XSLT 1.0 stylesheets developed by Sebastian Rahtz over the last year, and freely available from the TEI website ([16]). The process has several stages. In the first stage, a RelaxNG schema fragment corresponding with the pieces of compact syntax shown above is generated. This is then progressively flattened and simplified by a further set of XSLT transforms to produce forms from which equivalent DTD and W3C Schema fragments can be generated using James Clark's trang ([17]). This use of RelaxNG as a kind of ‘transport layer’ makes it simple to include XML vocabularies from other name spaces, such as MathML or SVG, within a TEI conformant schema.

3 Customizing the TEI

The current TEI draft defines over 20 modules, and about 500 elements. For practical purposes, any real life project will need to choose a subset of these modules, probably narrowing down the set of elements provided by each module. They may also wish to add some local datatyping constraints — having been conceived of as a system for interchange, the TEI scheme is extremely reluctant to narrow down for example the range of acceptable values for most attributes. They may wish to add new elements undefined by the TEI, or narrow the scope of what is permissable within an existing element. They may wish to operate using an XML vocabulary in which the TEI names have been localised to some other natural language.

Facilities to provide this range of customization have been built into the TEI model since the first draft, but in part because of their inevitable reliance on some rather obscure aspects of the DTD language, are not for the use of the fainthearted. More recently, a web application known as the Pizza Chef (http://www.tei-c.org/Pizza) has been provided to simplify the process. In P5, for the first time, schema modification and schema generation are handled in exactly the same way. Where before a customization or view of the TEI scheme involved declaring a number of parameter entities, and defining user-specific modification files in a particular syntax and order, operating directly in DTD language, it is now possible to do the whole job in ODD format. Thus, the following declares a schema which combines four TEI modules:

Figure 5

```
<schema>
<moduleRef name="tei"/>
<moduleRef name="header""/>
<moduleRef name="textstructure"/>
<moduleref name="linking"/>
</schema>
```

which can be directly mapped to the following RelaxNG grammar:

Figure 6

```
<grammar ns="http://www.tei-c.org/P5/"
 xmlns="http://relaxng.org/ns/structure/1.0"
 datatypeLibrary=
   "http://www.w3.org/2001/XMLSchema-datatypes">
<include href="Schema/tei.rng"/>
<include href="Schema/header.rng"/>
<include href="Schema/textstructure.rng"/>
<include href="Schema/linking.rng"/>
</grammar>
```

In principle, this simple case could also be mapped to an old-style DTD subset declaration such as

Figure 7

```
<!DOCTYPE TEI SYSTEM "tei.dtd" [
<!ENTITY % TEI.header "INCLUDE">
<!ENTITY % TEI.textstructure "INCLUDE">
<!ENTITY % TEI.linking "INCLUDE">
]>
```

but we have chosen not to provide this facility, because of the difficulty of supporting the full scope of modifications we now wish to support. Instead, we defer production of a compiled DTD to the last moment, after the RelaxNG schema has been assembled.

As an example of the kind of modifications we can now support, consider the following simple case:

Figure 8

```
<schema>
<moduleRef name="tei"/>
<moduleRef name="header""/>
<moduleRef name="textstructure"/>
<moduleref name="linking"/>
<moduleRef name="teiheader"/>
<moduleref name="verse"/>
<!-- add a new element -->
<elementSpec ident="soundClip">
<classes memberOf="tei.data"/>
 <attList>
  <attDef ident="location">
  <datatype><rng:data type="URI"/></datatype>
  <valDesc>A location path</valDesc>
  <desc>supplies the location of the clip</desc>
  </attDef>
 </attList>
 <desc>includes an audio object in a document.</desc>
</elementSpec>
<!-- change an existing element -->
<elementSpec ident="head" mode="change">
<content><rng:text/></content>
```

```
</elementSpec>
</schema>
```

This example includes the same modules as the preceding ones, but it also adds a new element (<soundClip>) and restricts the content model of an existing one (<head>). The new element declares itself a member of the tei.data class, and thus extends that class, in the same way as any other element specification might. The mode attribute on <elementSpec> is used to state the intended behaviour of this specification in the schema under construction: its value defaults to add, but may also be change where it is intended that any existing declaration should be retained except for those of its children which are being over-riden by the new specification (<content> in the above example), or replace where the new specification replaces the whole of an existing one, as in the next example.

One of the major simplifications in the new ODD system, for which we are indebted to Laurent Romary, is a conscious effort to provide uniform levels of description and hence processing for each of the components of the ODD model. Thus, modules, elements, attributes, value-lists are treated uniformly, and their specifications are all members of the same TEI class. Each has a unique identifier, an optional gloss, a description, and one or more equivalents. Each can be added, changed, replaced, or deleted within a given context according to a clear set of rules. This can be done at any point in a schema declaration. This gives us a simple method of satisfying a very long-standing TEI requirement: the ability to constrain data values for the same attribute in some cases but not others. For example, as mentioned above, there is tei.typed class, members of which all have a generic type attribute defined, by default, as having datatype rng:text. For some element (say, <list>) we might wish to constrain the legal values for that attribute, obviously without wishing to affect its values on all members of the tei.typed class. The new ODD system allows us to achieve this very simply by providing an additional declaration for <list> like the following:

Figure 9

```
<schema>
<moduleRef ident="core"/>
<!-- ... -->
<elementDecl ident="list" module="core" mode="change">
<attList>
<attDef ident="type" mode="replace">
<valList>
<valItem ident="ordered">Items are ordered</valItem>
<valItem ident="bulleted">Items are bulleted</valItem>
<valItem ident="frabjous">Items are frabjous</valItem>
</valList>
<desc>specifies the type of a list as ordered, bulleted, or frabjous.</desc>
</attDef>
</attList>
</elementDecl>
</schema>
```

This schema references the core module, which provides a default specification for the <list> element, in which the type attribute is not explicitly defined, but has the default characteristics implied by the element's membership in the tei.typed class. The schema above provides a new specification for the same element, to be processed in change mode, so that any child elements not present in the new specification will be retained unchanged. The new specification includes an <attDef> specification which is to be processed in replace mode, and which therefore entirely replaces the equivalent specification for this attribute in the default, supplying an explicit value list.

Modifications of this kind can also be chained together. Suppose, for example, that some other element (say <lexeme>) were specified by some hypothetical TEI module as follows:

Figure 10

```
<elementDecl ident="lexeme" module="hypothetical">
<classes>
<memberOf key="tei.typed"/>
</classes>
<attList>
<attDef ident="gender">
<valList>
<valItem ident="m">Masculine gender</valItem>
<valItem ident="f">Feminine gender</valItem>
```

```
</valList>
<desc>specifies the gender of a lexical item.</desc>
</attDef>
</attList>
</elementDecl>
```

To provide an additional legal value for this attribute in a schema, all that is needed is a specification like the following:

Figure 11

```
<schema>
<moduleRef key="hypothetical"/>
<elementDecl ident="lexeme" module="hypothetical" mode="change">
<attList>
<attDef ident="gender">
<valList>
<valItem ident="n" mode="add">Neuter gender</valItem>
</valList>
</attDef>
</attList>
</elementDecl>
</schema>
```

This works because the new specification for the <lexeme> element, and hence for its child gender attribute specification, is to be processed in change mode. The new <valItem> specification however is processed in add mode, and is thus simply added. If no mode value were specified, the default would again be to process this element in change mode — which would be an error, as there is no existing <valItem> with the identifier n.

We have now started to test Roma (http://tei.oucs.ox.ac.uk/Roma) a replacement for the ‘pizza chef’ application mentioned above. This new web-based application was first reported in a presentation at XML Europe in 2003 ([18]) but was completely recoded by Arno Mittelback and Sebastian Rahtz in July 2004. The Roma web-based service allows users to choose and customize modules, to add new elements or classes, and to modify existing ones, by selecting from a simple series of web pages, built dynamically by queries against an eXist database which has been populated by processing the current TEI sources. Outputs from this interface naturally include a schema or DTD in any of the various schema languages supported, plus reference documentation for the subset of ODD components chosen for inclusion in the target schema. As a further refinement, and building on ideas proposed by Alejandro Bia ([19]), the schema generated can use some language other than English for its identifiers and values: at present Spanish and German are supported.

4 Living in a joined-up digital world

We stated above that a major objective of the P5 revision was to improve the points of contact between the TEI scheme and the many other standards and initiatives now in use. We have already indicated how the use of modern schema languages allows us very simply to incorporate XML vocabularies from different name spaces, but that is of course a long way short of providing the kind of interoperability envisaged for the semantic web. The new ODD format allows us to make a step in that direction, by providing for the specification of an <equiv> element for each and every object (element, attribute, value) in the TEI object model. This element can be used to supply a URI which denotes a concept equivalent to that which the TEI object denotes. The URI will typically reference some externally-defined ontology, for example an item in such things as the ISO data category registry ([20]), or the CIDOC conceptual reference model ([21]), or even a published subject as used by the Topic Map standard ([22]). We have started some experiments with this notion in the context of linguistic markup, and the approach seems viable.

Aside from the difficulty of mapping between markup and markup semantics, the problem of dealing with multiple namespaces is that categories in different name spaces often overlap. In work reported elsewhere ([23]), we described some experiments in intertwining the TEI and DocBook vocabularies. The objective of that work was to allow a TEI document in which DocBook elements could appear, and in which DocBook elements could themselves contain TEI elements. This is not such a fanciful case: a TEI document might wish to use the DocBook GUI elements, rather than go to the trouble of defining them; within such elements, one might well wish to use the TEI structured inline elements such as <date>. Using elements from the two name spaces is easy:

Figure 12

```
<TEI  xmlns="http://www.tei-c.org/ns/1.0"
       xmlns:dbk="http://docbook.org/docbook-ng">
<text>
 <body>
  <p>The button on our web page has the current date:
    <dbk:guibutton>
     <date  calendar="Julian" value="1732-02-22">
        Feb. 11, 1731/32, O.S.
     </date>
    </dbk:guibutton>
    or at least the date on which we last updated it.
  </p>
 </body>
</text>
</TEI>
```

The problem is: how is such a composite document to be validated? Somehow, we have to be able to tell the TEI that it can use the DocBook gui.inlines class along with its normal phrase-level elements, and at the same time tell DocBook that it can use the TEI class data in its phrase-level class. Using a RelaxNG schema this turns out to be simple because we can directly manipulate the patterns which define the two classes:

Figure 13

```
include "tei.rnc" {
        tei.hqphrase |= gui.inlines
        }
include "docbook.rnc" {
        docbook.text |= tei.data
        start = TEI
}
```

We are not yet sure how to achieve this effect in the ODD language, which currently lacks any way of referencing externally defined classes such as gui.inlines in the above example. Nor is it clear how such a composite schema should be converted to a DTD. Nevertheless, the possibility is an exciting one, on which work continues. Other possible enhancements which we are considering include ways of embedding better data validation using a constraint language such as Schematron ([24]), and also ways of using the Namespace Routing Language ([25]) to integrate the validation process.

5 Conclusions and future directions

The work reported here and over the last few years provides a clear way forward for the TEI, enabling it to achieve the goals set out for its next major development cycle. Some concerns remain: many members of the TEI community are firmly wedded to DTD technologies and may find it difficult to move to the new generation of schema languages. While concerned not to disenfranchise such users, it is hard or impossible to ensure that all the facilities developed in the context of newer languages can be made available to them; partly for this reason, the TEI has agreed to maintain the P4 release, the last version to use an XML DTD, in parallel with P5 as that takes shape.

Another, and perhaps more fundamental concern, is that the new extensibility features take the TEI into uncharted waters. It should be remembered that the Initiative originally came about as a means of facilitating reliable interchange of digital data. The implications of widespread deployment of multi-namespaced, highly configurable, modern schemas for that process are yet to be discovered, though some of them are teased out in Syd Bauman and Julia Flanders' presentation at the present conference ([26]).

In conclusion, we should remember that constraint languages (whatever their syntax) are always going to be a minority interest. For most TEI users, the greatest attraction of the scheme described here is that it offers a single abstraction layer in the shape of the ODD language which can be used by designers, users, and system developers alike. A single vocabulary and a single set of tools for definition and for customization, applicable to the structure and the content of marked up documents is an idea whose time has definitely come.

Bibliography

[1] C.M. Sperberg McQueen and Lou Burnard (eds) Guidelines For the Encoding and Interchange of Machine-Readable Texts1990 Available from
http://www.tei-c.org/Vault/GL/teip1.tar.gz

[2] TEI Guidelines for Electronic Text Encoding and Interchange (TEI P3) Sperberg-McQueen, C.M. and Lou Burnard, eds. Chicago, Oxford: ACH-ALLC-ACL Text Encoding Initiative, 1994.

[3] TEI Guidelines for Electronic Text Encoding and Interchange (TEI P4). Sperberg-McQueen, C.M., Lou Burnard, Steve DeRose, and Syd Bauman, eds. Bergen, Charlottesville, Providence, Oxford: Text Encoding Initiative Consortium, 2002.

[4] TEI Sourceforge project at
http://tei.sourceforge.net

[5] W3C Math working group What is MathML? Available from
http://www.w3.org/Math/

[6] Scaleable Vector Graphics (SVG) XML Graphics for the web. Available from
http://www.w3.org/Graphics/SVG/

[7] N. Walsh and L. Muellner, DocBook The Definitive Guide, O'Reilly, Sebastopol, CA, USA, 1999.

[8] Metadata Encoding and Transmission Standard: official web site. Available from
http://www.loc.gov/standards/mets

[9] Encoded Archival Description (EAD) : Official EAD Version 2002 Web Site Available from
http://www.loc.gov/ead/ead.html

[10] XCES: XML Corpus Encoding Standard Available from
http://www.xml-ces.org

[11] Sebastian Rahtz, Converting to schema: the TEI and RelaxNG. Available from
http://www.idealliance.org/papers/xmle02/dx_xmle02/papers/03-03-08/03-03-08.html. Paper presented at XML Europe 2002, Barcelona, May 2002.

[12] Information on the Perforce Software Configuration Management System is available from
http://www.perforce.com

[13] Information on the eXist Open Source XML Database system is available from
http://exist.sourceforge.net

[14] C.M. Sperberg-McQueen and Lou Burnard, TEI ED P1: Design Principles for Text Encoding Guidelines 1988 Available from
http://www.tei-c.org/Vault/ED/edp01.htm

[15] TEI P5, Documentation Elements (Draft fascicule available from
http://www.tei_c.org/Activities/META/FASC-td.pdf)

[16]

[17] Information on Trang, a multiformat schema converter based on Relax NG, is available from http://www.thaiopensource.com/relaxng/trang.html

[18] Sebastian Rahtz, Building TEI DTDs and Schemas on demand Available from http://www.idealliance.org/papers/dx_xmle03/papers/03-01-04/03-01-04.html. Paper presented at XML Europe 2003, London, March 2003.

[19] Alejandro Bia, Manuel Sanchez-Quero and Regis Deau, Multilingual markup of digital library texts using XML, TEI and XSLT. Available from http://www.idealliance.org/papers/dx_xmle03/papers/03-01-03/03-01-03.html. Paper presented at XML Europe 2003, London, 2003.

[20] International Organization for Standardization ISO 12620: A data category registry. Drafts available from http://www.tc37sc4.org/document.htm

[21] CIDOC Conceptual Reference Model (CRM); available from http://cidoc.ics.forth.gr/

[22] XML Topic Maps (XTM) 1.0 specification available from http://www.topicmaps.org/xtm/

[23] Sebastian Rahtz, Norman Walsh, and Lou Burnard A unified model for text markup: TEI, Docbook, and beyond, Available from http://www.idealliance.org/papers/dx_xmle04/papers/03-08-01/03-08-01.html. Paper presented at XML Europe, Amsterdam, April 2004.

[24] Schematron — A language for making assertions about patterns found in XML documents available from http://www.schematron.com

[25] James Clark, Namespace Routing Language available from http://www.thaiopensource.com/relaxng/nrl.html

[26] Syd Bauman and Julia Flanders Odd Customizations Presentation at Extreme Markup Languages, Montr�al 2004.

Notes

1. The TEI commissioned an ‘SGML migration taskforce’ to investigate this; its reports are available at http://www.tei-c.org/Activities/MI