New tricks from an old dog: An overview of TEI P5

Lou Burnard

Abstract

This paper presents an update on the current state of development of the Text Encoding Initiative's Guidelines for Electronic Text Encoding and Interchange. Since the last major edition in 2002, which saw the conversion of the Guidelines into XML, there has been substantial activity on adding new content in areas of particular interest to historical corpus builders. The TEI has also reinvented itself as a membership initiative and set up mechanisms for the continued development and maintenance of the Guidelines. We contrast "old" and "new" TEI, and give a brief overview of some recent technical enhancements to the system intended to facilitate expansion and customization of the scheme.

1 What did the TEI ever do for us?

Monty Python fans will recall the scene in which a spokesperson for People's Front of Judea (or was it the Judean Peoples' Front?) asks rhetorically 'But what did the Romans ever do for **us**?', only to be overwhelmed with responses such as 'Well, the roads... the sanitation... the cooking...'. In rather the same spirit, if asked what the TEI has contributed to the general area of digital scholarship, broadly defined as those who work in a critical and scholarly way with digitized textual materials, we can reasonably point to a not unimpressive list of modest achievements. Each of the four major published drafts of the TEI's Guidelines for Electronic Text Encoding and Interchange (TEI 1990, 1992, 1994, 2000) has consolidated its position of pre-eminence within an increasingly active field. The TEI Guidelines describe and define nearly 500 well-documented textual distinctions, covering the encoding needs of many academic fields in exhaustive detail. The Guidelines are further complemented by a detailed technical implementation, which is based entirely on the use of XML standards and can thus be supported by any of a wide and continually expanding range of tools. The whole is organized within a modular and extensible architecture, with an independent academic governance, and a remarkably wide-ranging and varied community of practice. Approaching its second decade, an eternity in the world of information technology, the TEI might be expected to feel moderately complacent at its success, but also to be preparing to retire gracefully from the scene.

Such a view might also be encouraged by the observation that the TEI had its origins in a specific academic research project, which although well funded initially, had been unable (like many other very successful academic research projects) to agree on either an exit strategy or a viable business model for its continuation. Although the TEI had succeeded beyond expectation in codifying — even perhaps in determining — common encoding practices, it lacked any formal way of coping with their ongoing rapid evolution. Its licencing and development practices were somewhat uncertain. Despite its widespread take-up, it was perceived in some quarters as being unmanageably complex except by an inner circle of devotees, and simultaneously (even sometimes by the same people) as being too simple for real scholarly work. Where were the tools to do new and interesting things with TEI-encoded texts and how might their production best be fostered? Such criticisms reflect above all the lack of an active maintenance and development community for the TEI.

A roadmap addressing these concerns was announced in the year 2000 with the launch of a new governance structure for the TEI, re-organised as a membership consortium. Aside from constitutional issues, this announcement also mapped out a clear path for the future technical development of the Guidelines. The first step would be to convert the whole system to use XML rather than SGML as its means of expression; this was accomplished with the current P4 release in 2002. Backwards-compatibility was an explicit design goal for that release, and therefore no attempt was made to remove or revise much which might be considered obsolescent, nor to address any of the many new areas of digital activity where the TEI approach might be of use or relevance. As the preface to P4 indicates, 'these tasks require the existence of an informed and active TEI Council to direct and validate such

extension and maintenance work in response to the changing needs and priorities of the TEI user community'. Setting up and managing the work of that Technical Council and its associated workgroups has been the major achievement of the TEI Consortium over the last five years. The release of the next definitive edition of the Guidelines, TEI P5, due in October 2007, marks the fruition of that work.

2 The Road to P5

At the second annual TEI meeting, held in Chicago in November 2003, three chief aims for the P5 release of the TEI Guidelines were identified :

Interoperability or, in other words, taking advantage of the work done by others

Internal audit cleaning up the accretions of a decade

Expansion addressing areas as yet untamed

This ordering of priorities reflects a major design goal: where previously the TEI needed only to consider the needs of its own immediate research communities, it now needed to find its place in a world dominated by XML and the World Wide Web, to reinvent itself as a good citizen rather than a frontiersman, without losing touch with its core constituency. It also reflects the confidence of the Technical Council that the architectural advances on previous versions embodied in P5 are capable of sustaining effective customisation, internationalization, and interoperability much more easily than hitherto.

2.1 Interoperability

The change to XML in P4 removed immediately many fundamental interoperability problems associated with the secure transmission and interchange of digital data, notably those relating to the management of characters and writing systems. With the advent of Unicode, it is hard to remember how ambitious it once seemed to propose that of the TEI might somehow support in a single format the 'blind interchange' of data in all languages from all times, using all writing systems. In order to take full advantage of the sterling work carried out by such agencies as the Unicode Consortium, however, some encoding habits need to change.

Rather than representing 'non-ASCII' characters by ad hoc (or even community-defined) conventional names, TEI encoders now need to be encouraged to use Unicode characters directly or (where input devices do not permit this) to prefer numeric character references; such usage is a more reliable way of ensuring the portability of their data. For similar reasons, the old TEI global attribute lang, which used to indicate language and writing system by means of an arbitrary user-defined code has been replaced by the functionally-equivalent (but W3C-defined) attribute xml:lang, which takes as value language and writing system identifiers defined by ISO standard. This is one particularly striking aspect of a design decision running throughout TEI P5: wherever ISO or W3C standards exist for particular aspects of encoding (other examples include values for dates and for sex), they have been adopted. The fact that current XML tools (even those entirely ignorant of the TEI) generally permit the use of such predefined value ranges to be validated automatically is one good reason for this; another is that there is simply no good reason to re-invent such standards.

At P5, since we assume a substrate of XML represented in Unicode, the representation of non-standard (i.e. non-Unicode) characters and glyphs can only be achieved by means of markup constructs: the new `<g>` element is introduced for that purpose. This enables the encoder to record a particular character or glyph variant, associating it with both a convenient Unicode representation (which might for example be taken from the Private Use Area) as content and with full documentation of its properties by means of a pointer to the new `<charDesc>` element provided in the TEI header. The `<g>` element is permitted everywhere that text is permitted in a TEI document; its name is short for the Japanese term gaiji (`[U+5916][U+5B57]`), literally meaning "external characters", used for kanji that are not represented in existing Japanese encoding systems, but its application extends to any language.

Two further examples may be given of the way in which TEI P5 has preferred to adopt standards or practices developed outside the TEI to its own ends, rather than re-invent them. The first is the

replacement of the global id attribute of TEI P4 by the semantically equivalent xml:id attribute defined as part of XML itself. The second is the adoption by the TEI of the W3C Namespace Recommendation, and its application throughout the TEI scheme. The use of namespaces within TEI schemas is a key enabling technology, making it possible to embed other XML vocabularies within TEI documents in a controlled and validatable manner, and also to provide a more exact definition of TEI conformance than was previously possible.

As a simple example of the first point, consider the TEI `<figure>` element markup which is used to mark the presence of some graphic component in a TEI document. At P5, the graphic component itself can be represented by means of an embedded `<graphic>` element, which supplies a pointer to the required graphic file:

```
    <figure>
  <graphic url="picture.jpg" width="6in"/>
</figure>
```

But with the advent of XML vocabularies for the direct representation of graphic information, notably the Standard Vector Graphics language SVG, it becomes possible to embed an SVG representation of the graphic directly within the `<figure>` element itself:

```
    <figure>
  <svg xmlns="http://www.w3.org/2000/svg" width="6cm" height="5cm" viewBox="6 3 6 5">
    <ellipse xmlns="http://www.w3.org/2000/svg"
      style="fill:  #ffffff"
      cx="9.75"
      cy="6.35"
      rx="2.75"
      ry="2.35"/>
  </svg>
</figure>
```

The TEI community now has a decade's experience of developing and deploying digital libraries composed of linked page images and transcriptions: recommendations for best practice in this area, using these and other linking techniques, are also included in TEI P5.

As regards the impact of namespaces with respect to TEI conformance, we note here only that with P5 a TEI namespace is for the first time defined, and its proper use in encoded documents becomes a mandatory requirement for TEI conformance. In practice this means that documents which use only elements defined by the TEI encoding scheme can now indicate as much by explicitly invoking the TEI namespace, thus distinguishing themselves clearly from documents which merely claim to be 'TEI-like' or 'TEI-inspired'.

2.2 Internal audit

A system the size and complexity of the TEI necessarily provides much more scope for revision than can feasibly be described by a short article of this kind. We discuss here just a few of the more striking revisions which have been implemented in P5, in particular: revision and systematic tidying up of the range of possible attribute values; revision of the linking mechanisms; improvement and generalisation of the element class system; development of a fully TEI-conformant schema specification language.

2.2.1 Attribute values

In SGML, and consequently in earlier versions of the TEI, the range of datatype checking available for attributes is fairly limited. An attribute value may be given a 'declared value' such as numeric, name, or character data; it may be restricted to one value from a predefined list; it may be a pointer. Earlier versions of the TEI attempted to build more specific restrictions on possible values into the documentation, distinguishing between values which are 'Legal' (i.e. enforced by the DTD) 'Suggested' (i.e. recommended but not enforceable), or 'Sample' (i.e. used simply as illustration). At P5, the availability of schema languages such as RELAXNG or W3C Schema enables us to build in far more sophisticated datatype-checking, where this is appropriate. For example, attributes used to supply a

normalized value for date or time can do so by reference to the ISO or W3C standard representations for such values, in the the reasonable expectation that the values will be checked automatically. Equally, where a restricted range of possible values is defined (whether canonically in the Guidelines, or by means of a user customization), a schema processor can check that only legal values are used in a given document.

Earlier versions of the Guidelines licensed almost any string of characters to be supplied as an attribute value, including material in natural language which might equally well be considered as content. This is a matter of idiom; however, although one may be subsituted for the other, there is an important formal difference in XML between an attribute value and the content of an element: the former may not contain any markup. With the introduction of the <g> element mentioned above, it becomes increasingly probably that markup constructs will be needed wherever text is used; consequently, a design decision was taken to re-express any attributes which might plausibly be consided textual in nature by equivalent element content.

This necessitated the introduction of a new <choice> element to replace the small family of 'mirror tags' originally used in TEI to represent for example both the original form of a word and a standardized spelling of it, or an error and its correction. For example, a text using the old spelling form yeere for the modern year might, in P4, be tagged in either of the following ways:

```
<reg orig="yeere">year</reg>
<orig reg="year">yeere</orig>
```

Aside from the processing overhead of maintaining two different ways of encoding essentially the same phenomenon, this has the serious drawback that it is now impossible to indicate the presence in the original text of (say) an unusual glyph, such as a variant E defined here as 'e42':

```
<reg orig="y <g ref="#e42"/>re">year</reg>
```

is syntactically invalid.

Using the new <choice> element provides an effective solution to both problems:

```
<choice>
  <reg>year</reg>
  <orig>y<g ref="#e42"/>re</orig>
</choice>
```

Similar idioms are feasible in several other cases. For example, many of the elements used to document non-linguistic annotations in the transcription of speech have a desc attribute used in the following uncontrolled manner:

```
<event desc="transcriber dozes off"/>
```

Making this description into a child element not only allows one to include markup in the string, but also permits parallel versions in distinct languages, which the attribute solution would not:

```
<event>
  <desc>transcriber <name>Jim></name> dozes off</desc>
  <desc xml:lang="fr">transcripteur <name>Jim</name> s'endort</desc>
</event>
```

2.2.2 Pointing mechanisms

A rather different kind of change in TEI P5 is the revision of the pointer mechanisms. It is worth remembering that the TEI largely predates the mass take-up of the World Wide Web; indeed that one of the two original TEI editors, Michael Sperberg-McQueen, moved on after his TEI work to become one of the two editors of the original XML specification for the W3C. It is not surprising therefore to find in the original TEI specifications notions which prefigure current web orthodoxy. When the TEI system was developed, in a world which was not yet massively interlinked by computer networks, it seemed natural to distinguish formally between 'intra-document' and 'inter-document' links, and to assume different syntaxes for each. At that time also, while the linking syntax supported by the fledgling Web was certainly worthy of consideration, this was as but one of a number of alternative approaches to the development of the docuverse of hypertext pioneers.

Fifteen years later, however, such assumptions seem inappropriate. Just as in Unicode we have what is effectively a universal character encoding, so in the URI syntax developed by the W3C, we have what is effectively a universal linking mechanism, which it would be idle for the TEI to ignore — if only because of its strong similarities in concepts and execution to ideas already familiar to the TEI designers. Hence, at TEI P5, all pointing is done in the same way, using a Universal Resource Indicator (URI). An intra-document pointer which in P4 would have been represented using an IDREF such as `<ptr target="foo"/>`, would in P5 be represented as `<ptr target="#foo" />`, thus making explicit that intra-document pointing is actually a special case of the inter-document version `<ptr target="http://www.examples.org/somewhere.xml#foo"/>`.

This simple change has the interesting side-effect that the pointing mechanisms used throughout the TEI for such purposes as linking a coded value with its expansion, or a reference to some entity such as a name with its canonical form, can now expand beyond the confines of a single document. Collaborative projects can share a common authority file to which individual TEI documents can link simply and directly, without the expense and inefficiency of duplication.

At the same time, the URI syntax permits extensions functionally almost identical to those proposed by the original TEI pointing language, though with minor differences in syntax. And, it may be noted, major differences in the availability and extensiveness of their implementations.

2.2.3 The TEI class system

Fundamentally the TEI is, as it always has been, a large XML vocabulary or ontology: a set of concepts for which suggested names are provided. It goes a little beyond that however in also saying something about the ways in which the named elements may meaningfully be combined within a document: for example that the element named `<div>` may contains elements named `<p>` rather than the reverse. And of course it also attaches a substantial amount of semantic information by suggesting what kinds of real-world textual information it is appropriate to represent by means of a `<p>` element and which it is not.

Almost any system which manages several hundred differently named components will find it necessary to find some way of grouping and organizing those components, and the TEI is no exception. In earlier versions, each TEI element was assigned to a different 'tag set', and also, in some cases, to one or more 'class'. The former provided a means of grouping related sets of element declarations so that they might be combined together to form a document type declaration (DTD). A distinction was made between a tagset used to represent the basic structure of a TEI document (a 'base tagset') and the sets of related application-specific elements (the 'additional tag sets') which might optionally be combined with it; there was also the concept of an 'auxiliary' tagset, used to define specific kinds of metadata such as Writing System, which might be employed or referenced by a TEI document. At the same time as this, some but not all elements were organized into model classes and attribute classes, represented in the original TEI scheme by an SGML construct known as a parameter entity. Several key parts of the whole architecture depended on the availability of these parameter entities, notably those parts concerned with supporting extension and modifiability[1] which were an important concern in the original system design.

In developing P5, it was decided to retain many of these design features, while at the same time simplifying and generalizing some of the underlying concepts. The first decision was to remove the distinctions between different kinds of tag set. In TEI P5, every element is defined once in a module and any combination of modules can be used as required to build a schema. This is possible because of the existence of a single TEI infrastructure module which defines the basic TEI infrastructure of classes and macros. Although the old 'core' and 'header' modules remain (defining the large set of elements considered useful in 'almost every kind of text', and the TEI's general purpose metadata framework respectively), in principle they are no different from any other module with which they can be combined ad lib.

[1] For an informal introduction to the so-called Pizza Architecture of the original TEI scheme, see Lou Burnard and C.M. Sperberg-McQueen "The design of the TEI encoding scheme". *Computers and the Humanities* 29.1 17-39. Rpt. in *The Text Encoding Initiative: Background and Contexts*, ed. Nancy Ide and Jean Veronis. Dordrecht, Boston: Kluwer Academic Publishers, 1995.

The second decision was to extend and generalize the TEI class system. Every TEI element can be a member of one or more attribute classes and of one or more model classes. From its membership in an attribute class, an element inherits definitions of its attributes: this makes it possible to ensure that attributes are all declared uniformly. From its membership in a model class, an element inherits a position in the content model of other elements. As far as possible, all content models are defined with reference to classes of element, rather than to individual elements. In simple terms, the content model for is expressed, not as being 'one or more <head> elements followed by one or more <p> elements', but rather as a sequence of members of the model.headLike class, followed by a sequence of members of the model.pLike class. Classes may themselves be classified: that is, a class may be defined as a combination of other classes. The subclasses of a given model class are generally distinguished semantically: for example, the large model class model.phraseLike includes many other subclasses such as model.dataLike, model.hiLike etc. Each subclass combines elements which have some semantic property in common, but all of them are structurally identical.

It is easier, of course, to understand and remember a few dozen class names than a few hundred element names, but this is not the only reason for considering the generalization of the class system as a major enhancement to the usability of the TEI scheme. As we noted above, a key objective of the scheme is to facilitate extension and modification in a controlled way. The use of the class system means that adding a new element becomes a matter of deciding which of the existing elements it is 'like', or what it is a 'part' of. Specializing the TEI's often very generic structure in response to more specific needs becomes a simple matter of declaring the required additional elements and slotting them into the existing TEI infrastructure. Moreover, when a customization has been expressed in terms of class modification using the TEI mechanism defined for this purpose, documents conforming to that customization are inherently interoperable with other documents.

Space precludes a detailed description of the mechanisms supporting this claim (which is however provided in the TEI Guidelines themselves). A key aspect is the use of a single TEI-defined markup vocabulary, the ODD language, which is used to define both the TEI scheme itself, and also customizations of it. In this language, definitions are provided for classes, modules, elements, attributes, and value-lists, treating all of them uniformly. Every element is defined by an XML element called an <elementSpec> which provides, for example, a declaration of its intended semantics, examples of its usage, and alternative names for it as well as specifying its class memberships, content model, and local attributes, along with the module to which it is assigned (if any). A <schemaSpec> element defines a schema by combining such element specifications, either directly, or more usually by reference to a <moduleSpec>. A document containing these specification elements is called an ODD[2]; the TEI Guidelines are themselves an instance of an ODD. Crucially, these specifications may be chained together into a processing pipeline in which a processor (typically an XSLT transform) is carried out on one ODD to generate another, thus, for example, simplifying or modifying content models in light of the class declarations of available element specifications. From an ODD specification we can also of course generate the documentation and formal schema specifications which any markup scheme needs.

Sample use cases for this mechanism are not hard to find. For example, many elements in the TEI scheme are members of the att.typed class, which means that they are provided with a generic type attribute, the possible values for which are not specified in the TEI scheme. By supplying an additional <valList> declaration in an ODD, the user can constrain the values accepted for this attribute to a closed list of their making. Documents produced for this schema will remain interoperable with other TEI documents, of course, but will be more tightly constrained. As another example, many elements in the TEI scheme are members of the class model.hiLike, used to represent any kind of notable typographic highlighting. If distinguishing between kinds of highlighting is a matter of significance, an ODD can be defined which declares many additional elements such as <fraktur>, <italic>, etc. each of which declares itself a member of this same class. A TEI-aware piece of software can now either provide either a specific behaviour for these new elements or use its knowledge of their underlying class

[2]For One Document Does it all; see Lou Burnard and and Sebastian Rahtz*RelaxNG with Son of ODD*www.mulberrytech.com/Extreme/Proceedings/html/2004/Burnard01/EML2004Burnard01.pdf

membership to provide a fall back behaviour. A person wishing to produce such a piece of software can similarly determine both the purpose for which such elements were introduced into the schema, and their relation to existing TEI scheme.

The ODD processor developed for the TEI at Oxford by Sebastian Rahtz should be regarded as the first of a new generation of TEI specific tools able to take full advantage of these mechanisms. Currently available as both an online web-hosted application and a command line tool, roma (`http://www.oucs.ox.ac.uk/roma`) is simply a suite of XSLT stylesheets which carry out the processes discussed above. With its aids, users of the TEI can generate their own customized schemas in DTD, W3C or RELAXNG schema languages as well as their own localized documentation.

2.2.4 Expansion

P5 includes significant amounts of completely new material, not previously addressed by the TEI scheme at all. Examples include new modules for the detailed description of manuscripts, and also (in a striking departure from earlier TEI practice) for the structured description of real world entities such as persons and places independently of textual references to them.

P5's proposals for manuscript description provide a very rich vocabulary for the organization and markup of extremely detailed descriptions (or brief characterizations) of manuscripts or similar text-bearing objects, such as incunabula or monuments. Intended use cases comprise both the inclusion of such descriptions within the header of a digital edition or facsimile, as a highly specialized form of `<sourceDesc>`, and also their inclusion within the body of a document constituting a catalogue of such materials. These two uses also reflect two frequently opposed requirements: on the one hand, the desire to encode without loss of detail the traditional descriptive *catalogue raisonnée* as a text in itself; on the other hand the desire to replace this discursive text with a structured set of data to support true 'digital codicology'. This is a tension to be found in many parts of the TEI — it also pervades the chapter on encoding of dictionaries, for example.

The work on manuscript description also highlighted a need previously encountered within the TEI community, for representing within a TEI document not simply references to real world entities such as persons or places, but also structured information about such entities, analogous to a database record or other canonical definition. In previous editions of the Guidelines, the provision of such elements had been left to one side, since the work group concerned (primarily concerned with the markup of historical sources) had considered it out of scope, altough some work had been done by another group concerned with the markup of transcribed language interactions for whom some way of recording such salient features of the speakers of a transcribed dialogue as age, gender, location etc. was essential.

At P5, the opportunity was taken to revisit this territory, also taking into account the amount of work done elsewhere in defining ontological data about historical events and people, and the experience of some major projects in the encoding of the epigraphy and other records of classical scholarship. The new material resulting from this work enables researchers to maintain detailed gazetteers or biographical histories for all the persons and places referenced in their sources, which can then be used to generate indexes to the source, or to support intelligent searching of it.

With this new work, the scope of the TEI encoding scheme expands beyond the simple representation of textual structures to include the representation of the knowledge inferred from or implicit in those textual structures. A bridge between the TEI scheme and such ontologies as ISO's data catagery registry or CIDOC's CRM is now discernible, which may perhaps prove to be as significant in the development of Web 2.0 as the TEI has already been in the development of Web 1.0.

3 Conclusion

The 'new' Text Encoding Initiative is able to take advantage of a new technical architecture, still recognisably similar to the original model, but completely rethought to facilitate expansion and integration with other systems in the light of new XML technologies. Its products are released under an explicit open source licence, with visible development activity on Sourceforge. Each of the interim candidate releases for P5 since the 0.2 release in 2003 has been readily accessible to the TEI community in source form, together with an increasing number of supporting examples and test files. Because all

of the TEI documentation is now written in TEI, it is also automatically self-validatable. Tools for processing the Guidelines, and hence for processing other TEI documents are distributed through the same standard mechanisms. Complenting these activities by the TEI Council, Editors, and workgroups, we have seen the continued emergence of an active developer community, sharing expertise by means of discussion lists, wikis, training opportunities, and regular meetings of all kinds. Such activities have enormously enriched and enlivened the process of delivering the new edition of the Guidelines, and demonstrated the value of the TEI's new community-based governance structure. In a sense then, to parody another famous remark, the arrival of TEI P5 demonstrates, not what the TEI has done for us, but what we have done for the TEI.