Towards a TEI-based encoding scheme for the annotation of parallel texts

Peter Boot
Huygens Institute, Royal Netherlands Academy of Arts and Sciences, The Hague, The Netherlands

Abstract

Translation, adaptation, and other forms of appropriation of literary works can result in bodies of parallel texts. For the purpose of studying appropriation strategies, it is important to be able to annotate digital representations of these parallel text structures. This article uses early modern emblem culture (books of engravings or woodcuts, accompanied by mottos and explanatory texts) to investigate the forms this text parallelism may take. It defines requirements for annotation definition and proposes a TEI (Text Encoding Initiative) extension to implement these requirements. In the proposed encoding scheme, TEI feature structures will be used for storing annotation information. This scheme should be useful for annotating parallel text structures as well as for other annotation tasks. The annotation scheme assumes the annotated texts are available in XML. If this is not the case (there is no electronic version of the text at all or perhaps only a facsimile) the article suggests the definition of a TEI proxy document. A TEI proxy document contains enough of the structural aspects of the texts to serve as a basis for attaching annotations to the text. Outside of the annotation context, proxy documents may serve as a basis for adding functionality to image-based editions.

Correspondence:
Peter Boot, Huygens Institute (KNAW), PO Box 90754, 2509 LT, The Hague, The Netherlands.
E-mail:
pboot@xs4all.nl

1 Introduction

The translation and adaptation of emblems was a very frequent phenomenon in early modern emblem book culture. Emblem books were collections of emblems, a combination of a picture, motto, and a subscriptio, follow-up texts that could contain one or more epigrams, explanatory texts, etc. Adaptation or translation could be applied to a single emblem, but also to a series of emblems or to a whole book. A related phenomenon is the presence of epigrams in multiple languages in the emblems of a single book. These processes create bodies of parallel texts that merit study. In the Emblem Project Utrecht (EPU) corpus,[1] we encounter this situation within the confines of a single book (e.g. the vernacular epigrams in multiple languages in Otto van Veen's *Amoris divini emblemata*), between languages and books (e.g. the Dutch translation of Hugo's *Pia desideria*), and between books in a single language (e.g. Jan Suderman's reworking of Otto van Veen's emblems in *De godlievende ziel*).

I conceive of parallel text analysis as a task of annotation creation and visualization. It is clear other perspectives are possible and desirable.

doi:10.1093/llc/fqp023 Advance Access published on 18 May 2009

Automated analysis of vocabulary and stylistics would be very interesting. I see these processes as potential contributors of machine-created annotations. The questions of how to store parallel-text annotations and how to display them in a manner conducive to insight are independent of the way the annotations are created.

There are a number of technical challenges to overcome in annotating parallel texts. (1) Before the parallel texts can be annotated, they have to be aligned. In order for an annotation to be able to comment on two parallel fragments, it must either itself refer to both fragments or refer to some kind of link between the fragments. It is clear that creating a set of links between the parallel texts (an alignment) facilitates annotation and indeed is a condition for serious study of the parallelism. The alignment may be quite complex: in the process of adapting the text, its units may have been reordered, parts of it may have been removed, and units may have been split or merged. (2) An application that facilitates annotating parallel texts will have to provide for an aligned display of these texts (with all of the complexities that the alignment may have), where both the individual texts and the aligned fragments can be subject to annotation. (3) As useful as a body of annotations may be, it will only provide insight if it is possible to visualize the annotations based on the text alignment—only then can we can handle the higher-level differences in style, content and imagery that the parallel texts may reveal.

In this article I will focus on the first issue. The article continues work on the SANE (Scholarly ANnotation Exchange) Markup Language (Boot, 2006). It will formulate a TEI-based annotation format, and will replace the proprietary way of addressing the annotated resources used in the earlier article by the xpointer-based standard references of TEI P5, the most recent version of TEI. The format will use the mechanism of feature structures, recently standardized by ISO. I will begin by discussing some related work (in Section 2), and then discuss the requirements for the annotation format in Section 3. Section 4 proposes a TEI encoding for alignment and annotations. Section 5 discusses a prototype application that uses this format for storing annotations. The article assumes an overall familiarity with the TEI *Guidelines* (Burnard and Bauman, 2007).

2 Related Work

Annotation is a subject that has drawn much interest in recent years. Researchers have worked on annotation frameworks from a number of perspectives. Linguistically oriented frameworks are Bird and Liberman's work on annotation graphs (2001), Carletta *et al.*'s work on the Nite XML toolkit (2005), and the Linguistic Annotation Framework under development by Ide and Romary (2003). Ide and Romary use a model based on feature structures. An annotation model targeted towards humanities collaboratories is developed in (Agosti *et al.*, 2004). Microsoft researchers developed a Common Annotation Framework directed towards collaborative functionality in office applications (Bargeron *et al.*, 2001).

The focus of this article is more limited. This article does not attempt to design a generic annotation model, but proposes a TEI implementation of such a model. One respect in which this implementation differs from other annotation models is that it assumes the annotated documents are accessible over the web. The annotation document should be similarly accessible. While in general it may be good system design to hide implementation details behind an application programming interface, there are also important reasons for exposing stored data to the world. One of these reasons is digital durability (applications will inevitably stop working, while data may survive), another is the scholarly requirement of accountability, yet another the fundamental unpredictability of scholarly research needs (see also Section 4.1 below).

In a series of papers, John Bradley has been exploring appropriate data structures for annotation in the humanities (Bradley, 2004, 2005; Bradley and Short, 2005). A recurring topic in these papers is the suitability of XML and relational databases for storing annotation-like information. In its exploration of the use of 'data'-like feature structures for annotation, the present article can be thought of as a contribution to that debate.

3 Requirements

This section explores a number of requirements which an annotation format, more specifically a format that deals with text parallelism, will have to fulfil.

3.1 Parallel texts

Text parallelisms are not given in the text, but are created by the researcher. In creating a parallelism between texts one stresses some aspects of the text at the expense of others. In the simplest case, we can construct a 1–1 correspondence between the constituents of two or more parallel text structures, but usually the situation is more complicated. For one thing, we may want to create a parallelism not between two but between many texts. We may, for example, consider the parallelism between Van Veen and Suderman, but we might as well include one or more of the other adaptations of Van Veen's book.

There are many other complications that attempts to create an alignment between parallel texts may have to deal with:

- New units may have been introduced, old units removed, as in Harvey's reworking (*School of the Heart*) of Van Haeften's *Schola Cordis*. See (Bath, 2005);
- There may be different paratexts: Suderman's reworking of Van Veen's book is introduced by three Dutch prayers to the three persons of the Trinity; Van Veen introduced his book by (a.o. things) a Latin-language 'Carmen de amore'. This represents a dramatic change of context which influences our reading of the parallel texts;
- There may be changes in ordering of aligned units, as in the re-ordering of Alciato's emblems into a thematic collection, bringing it in line with the tradition of the common-place book. See (Bath, 1994; pp. 31–35). What is of interest here is not so much changes in the texts themselves as the way they are grouped into meaningful units;
- Part of a unit may have no counterpart in parallel text;
- Units may have been merged or split;
- Units may have been subsumed under a new hierarchy.

The alignment has to make clear what is aligned, but also what cannot be aligned.

The above assumes a situation of agreement with respect to structure as well as content. It assumes a parallelism at the book level that applies for all or most individual emblems, rather than the existence of an apparently unsystematic number of correspondences between two books. The type of relation that exists between the related constituents will be discussed in Section 3.3.

3.2 Annotanda

It follows from the above that annotation of parallel texts should not just be annotation of the parallel text fragments. It should also be possible to annotate those texts that have no equivalent in the other text. It should also be possible to address them as a group. Therefore, things that it should be possible to annotate (annotanda) are:

- Texts alignments as a whole: it should be possible to discuss the merits and demerits of a proposed parallelism. If, for example, we have created a parallelism at the level of individual poems in two books, it should be possible to argue that the parallelism had better be defined at the higher (emblem) level, or vice versa;
- The individual entries in the text alignment, i.e. two or more aligned text units (it should be possible to describe e.g. their relation in terms of similarity, the way they compare in a certain respect, etc.);
- Structural units in the text (poems, lines, pictures, emblems, pages);
- Fragments of text (a phrase, a sentence) or fragments of pictures;
- Groups or classes of text units ('the Spanish poems in *Amoris divini emblemata*', 'the preliminary material in the book', 'the poems without a counterpart in book x'). It should be possible to create these groups either by enumeration or by referring to some property these texts have (e.g. the xml:lang attribute in an XML file).

Finally, it should also be possible to create second-order annotations: to annotate other annotations and annotation types. It should be possible to comment on one's own annotations and on annotations made by others, as well as on e.g. the usefulness of a certain annotation type. As the annotation types embody a researcher's theoretical position, it is probable that serious discussion of another scholar's annotations will need to target the annotation types as well as the individual annotations.

3.3 Annotation types

In a paper on the on the EDITOR annotation tool (Boot, 2005) I have argued that usually researchers will want to define their own annotation types. The phenomena that a researcher studies are defined by his or her special interest and the texts being studied. The researcher will probably be looking for something new: therefore pre-existing annotation types will usually be of limited interest.

However, I want to add that we learned from work on the EDITOR annotation tool that an annotation system may want to make available a small number of default annotation types (such as 'comment', 'question', 'todo') in order to get annotators started. Following the ubiquitous web 2.0 practice, users will also expect a 'tag' annotation type. It should also be possible to re-use annotation types defined elsewhere, either by copying the definitions or by referring to them. [Compare Ide and Romary's proposal (2004) for a Data Category Registry].

Other requirements for the annotation types are largely similar to the requirements used for EDITOR: annotation types should be named and consist of one or more fields. These fields should have their own data type. In EDITOR the allowed data types were boolean, string, hyperlink and symbol (an enumeration of valid values), but other desirable data types include number, date, and rich text. Rich text is needed to do all those things that one usually does in prose: create lists, italicize, create hyperlinks, include figures, and create tables.[2] Two other requirements are the possibility to group fields together and the possibility to have repeating (groups of) fields (see examples in Section 4.2).

In addition to the fields that the researcher defines, an annotation system should create fields to hold information about the date the annotation was created, the date of last modification, and the users involved in these acts. These should be system-maintained fields available with each annotation type.

Using these components, scholars can create annotation types relevant to their research. An investigation into a reworking of an emblem book might define an annotation type that contains a number of rich text fields to describe possible changes in imagery, in the way of addressing an audience, and in vocabulary; a more quantitative approach might compare the rhetoric appeal of texts using a number field to hold the number of second person personal pronouns; another approach might characterize the adaptation process by choosing one or more of a limited set of available characteristics (e.g. 'literal translation', 'free translation', 'inspiration', 'modernization', 're-telling').

3.4 Annotations

An annotation connects an annotandum and an annotation type, and assigns values to the annotation type's fields.

Annotations will be created, stored, and displayed in sets. An annotation set will usually be stored in a single document. Besides the annotations, the document will contain information at the annotation set level (ownership, description, motivation, rights information), information about the documents that have been annotated, and about the annotation types.

4 TEI Representation of Annotations

4.1 Annotation sets

I assume a situation where the texts that will be annotated are available in XML form. The parallel texts may coexist in a single document, they may be stored each in their own document, or the individual text units may be stored in separate documents.[3] For the purpose of annotating the parallel texts, this is irrelevant. If one of the texts is not available as an XML document, we may want to create a TEI proxy document for that text (see Section 4.6).

For a number of reasons, the annotations will not be stored within the XML documents that contain the annotated texts:

- The choice would be either to create a new copy of the original XML file(s) or to modify the original copy. The first option would multiply the number of file copies and potentially create difficulties for annotation exchange. The second option may not be open to me (I will probably not have the right to change the original file) and would open up the files to corruption;
- As the number of annotations increases the size of the files might become unmanageable;
- As multiple people may want to annotate the same file, we would need complex locking schemes.

I will therefore assume the annotations are stored externally in a single TEI document for each annotation set. For performance reasons an actual annotation system may need a database backend, but conceptually I envisage a set of annotations as a document, available at a URL. One reason for this is that, unlike a database, a document is something that one can store and come back to a few years later with a reasonable chance of the document still being readable. Another reason is that from elsewhere in the world one can point to a location in a document, not to a record in a database.

The information at the annotation set level is best stored in the TEI header. A number of applicable elements follow, but this choice of elements needs further work.

Element	Information category
Title	Annotation set name.
Author	Annotation set author.
Availability	Rights information.
AboutDesc	To identify and describe the files that are being annotated, I suggest the creation of a new element, aboutDesc, that contains either a series of paragraphs or a list of bibliographical entries (listBibl). This new element would form part of the file description [The source description (sourceDesc) is unsuitable as these files are not sources].
ProjectDesc	Describes the purpose of the annotations and the thinking behind it.
FsdDecl	Declares available feature structures.
Creation	Annotation set creation date.
RevisionDesc	Revision log for the annotation set.
AppInfo	Records information about an application which has modified the annotation set.

I propose to use a new element dataSection to hold the data about the alignment and annotations. The dataSection is an element at the level of the text element in annotation documents. It would not be impossible to store annotation data in a text element but it is inappropriate. In 'regular' TEI documents, elements like this are usually relegated to the text's back element, but there too, it might be cleaner to introduce a dedicated element.

4.2 Annotation types

I will use the TEI mechanism of feature structures to model and store annotations. Feature structures are a mechanism originally developed for linguistic annotation. The TEI chapter on feature structures (Burnard and Bauman, 2007; ch. 18) has recently been reformulated as an ISO standard (ISO TC 37/SC 4, 2006). There is a separate mechanism (the feature system declaration) that is used to define acceptable feature structures and their allowable values. The feature system declaration is in the process of being standardized by ISO.

The feature system declaration is an element containing feature structure declarations, and each annotation type will correspond to a feature structure declaration. The feature structure declaration (an fsDecl element) can be defined in the annotation document, or it can be defined elsewhere and referred to using an fsdLink element. The fsDecl elements contain fDecl elements for the features (fields), and feature structure (fs) elements for the groups of features. The feature data types ('atomic values’) allowed in the TEI Guidelines are numeric, binary, string and symbol. I will extend this by also allowing date, note and ptr. The availability of the note element makes it possible to create annotations that use rich text. The availability of ptr makes it possible to refer to a web site (e.g. a Wikipedia page) as well as to point to any other resource accessible by URI: perhaps an image file, perhaps an entry in an external taxonomy, perhaps even another annotation. For the details of declaring feature structures I refer to the Guidelines, Section 18.11.

An annotation application would most likely, include a number of predefined annotation types (see Section 3.3), i.e. a number of feature structure declarations. Others will be created by the annotator. Such an application should provide a guided interface for the creation of these annotation types. A number of examples follow.

A ‘tag’ annotation type would consist of a single string field/feature, as follows:

```
<fsDecl type=“tag” xml:id=“. . .”>
  <fsDescr>Tag</fsDescr>
  <fDecl name=“tag”>
    <vRange><string/></vRange>
  </fDecl>
</fsDecl>
```

The vRange element defines the permitted content of the feature.

A researcher that wants to characterize a text reworking in terms of use of imagery, rhetoric and vocabulary, could define the following feature structure declaration. It would allow him or her to create a note about each of these subjects for each alignment entry or for the alignment as a whole.

```
<fsDecl type=“reworking” xml:id=“. . .”>
  <fsDescr>Describes reworking in terms of  three characteristics</fsDescr>
  <fDecl name=“charimag”>
    <fDescr>Describes changes in imagery</fDescr>
    <vRange><note/></vRange>
  </fDecl>
  <fDecl name=“charrhet”>
    <fDescr>Describes changes in rhetoric</fDescr>
  <vRange><note/></vRange>
  </fDecl>
  <fDecl name=“charvocab”>
    <fDescr>Describes changes in vocabulary</fDescr>
    <vRange><note/></vRange>
  </fDecl>
</fsDecl>
```

Outside of the context of annotation of parallel text, a researcher interested in the representation of conversations in novels, for example, might define an elaborate annotation type that, apart from a general description, contains information about the participants (name and age), about the approximate duration of the conversation (number of minutes), a reference to a discourse typology, and a series of references to earlier conversations that this conversation refers to. Such a feature structure might look like this:

```
<fsDecl type=“conversation” xml:id=“…”>
  <fsDescr>Describes conversations and their connections </fsDescr>
  <fDecl name=“general_description”>
    <fDescr>Describes general aspect of conversations</fDescr>
    <vRange><note/></vRange>
  </fDecl>
  <fDecl name=“duration”>
    <fDescr>Estimated duration in minutes</fDescr>
    <vRange><numeric value=“0”/></vRange>
  </fDecl>
  <fDecl name=“conversation_type”>
    <fDescr>Reference to xyz’s conversation typology</fDescr>
    <vRange><ptr/></vRange>
  </fDecl>
  <fDecl name=“participants”>
    <fDescr>A list of participants</fDescr>
    <vRange>
      <vColl org=“list”>
        <fs type=“participant”/>
      </vColl>
    </vRange>
  </fDecl>
  <fDecl name=“conversations_referred_to”>
    <fDescr>References to discussed conversations (refers to xml:id of corresponding
      annotations)</fDescr>
    <vRange>
      <vColl org=“list”>
        <ptr/>
      </vColl>
    </vRange>
  </fDecl>
</fsDecl>
<fsDecl type=“participant” xml:id=“…”>
  <fsDescr>Describes participant</fsDescr>
  <fDecl name=“name”>
    <vRange><string/></vRange>
  </fDecl>
  <fDecl name=“age”>
    <vRange><numeric value=“0”/></vRange>
  </fDecl>
</fsDecl>
```

‘Participant’ is a separate feature structure here, used in the larger ‘conversation’ feature structure. Notice the repeating field ‘conversations_referred_to’ and the repeating group ‘participants’ (see Section 3.3). The vColl element indicates the feature’s content is a collection of items.

It is clear feature structures, especially as extended here, offer a very powerful means of attaching many different types of annotation to texts in general and to parallel texts in particular. One question that may arise is how this general purpose mechanism relates to the more specific data structures that the TEI already offers. Suppose that in annotating a text we need to describe a number of persons in terms of date of birth, death, affiliation, etc. Obviously, we would want to use the prosopography elements made available with TEI P5. On the other hand, there is something to be said for having a single uniform mechanism for all annotations. Thus for at least some annotation types there will not exist ready-made TEI elements, using feature structures for all annotations remains an attractive solution—even more so if we should have an application that creates entry templates for feature structures based on arbitrary feature structure declarations.

One possible solution to this dilemma would be to reformulate the abstract TEI model for these elements, e.g. person, now formulated as an XML structure, in terms of feature structures. One could even imagine a feature system declaration that contains feature structure definitions corresponding to all or most data-like TEI elements (person, event, biblStruct, etc.). A user who needs to annotate a text using a bibliographical item could just point to the biblStruct feature structure and the application would present him/her with a template ready to fill out. It shouldn’t be too hard to convert this feature structure biblStruct into a canonical biblStruct when needed.

4.3 Alignment

Alignment will be represented by a linkGrp containing link-elements. The linkGrp element will have type=“alignment”. The targFunc attribute will specify the functions of the link elements’ targets, the first value corresponding with the first target, etc. All link and linkGrp elements carry xml:id-attributes, as they are all potential annotation targets (either within the current annotation set or from outside, perhaps in an annotation set created by a later scholar). Alternative alignments will be represented by a second linkGrp (or a third, etc.). In another extension to the *Guidelines* we use the head element to label the alignment.
An example:

```
<linkGrp
   targFunc=“English_epigram Dutch_epigram”
   type=“alignment” xml:id=“lg-1”>
   <head>Alignment Van Veen Amorum Emblemata dut-eng</head>
   <link n=“emblem 1”
      targets=“http . . ./vae.xml#v012 http . . ./vae.xml#v014”
      xml:id=“lg-1-001”/>
   <link n=“emblem 2”
      targets=“http . . ./vae.xml#v022 http . . ./vae.xml#v024”
      xml:id=“lg-1-002”/>
   <!-- more link-elements -->
</linkGrp>
```

This alignment aligns poems in Dutch and English within a single book. The targFunc attribute describes which is which. We might have chosen to align poems in three languages, or poems from multiple books.

One possible complication in an alignment of more than two text series, is that not all elements of the alignment are necessarily complete. In that case, the correspondence between the series of labels in the targFunc attribute and the series of pointers in the links' targets attribute will break down. The easiest solution would be to place the individual targets, with their own labels, in separate elements. This would result in the following:

```
<link n=“emblem 1” xml:id=“lg-1-001”>
      <target target=“http . . ./vae.xml#v012 ” targFunc=“English_epigram”/>
      <target target=“ http . . ./vae.xml#v014” targFunc=“Dutch_epigram”/>
</link>
```

It may sometimes be desirable to make clear which text units cannot be aligned in a given alignment. I propose to include in the linkGrp that defines the alignment pointer (ptr) elements with type=“unaligned” to point to the unaligned text units.

4.4 Annotanda and annotations

An annotation links something that is being annotated and the annotation's content. In our case, the annotation's content is a feature structure (an fs element). I propose to create annotations by creating a pointer (ptr) element that points at the annotandum using the target attribute, and at the annotation content (the fs element) using the ana attribute.[4] The ptr element will be assigned an xml:id attribute, so that the annotation can be a target for further annotation. The feature structure's type attribute refers to the type attribute of the corresponding feature structure declaration. Its feature (f element) children give the values for the fields defined in the feature structure declaration.

An alternative for using the ptr element to connect annotandum and feature structure would be to use link elements. However, the advantage of using a ptr element to point at the annotandum is that its target attribute can use multiple whitespace separated URI's to refer to a number of locations that are to be considered a single annotation target. This can be done while still unambiguously identifying the annotation content (the feature structure) in the ana attribute. When using the link element, both annotandum and feature structure are pointed to from the link element's targets attribute, causing room for confusion.

Pointers can point, according to the *Guidelines* (Section 16.2), at nodes, node sets, points (a location of length zero) and ranges (a range is a stretch of text between two points). TEI P5 uses the W3C's xpointer framework, for which it has registered a number of extension schemes. This seems sufficient for pointing at fragments of the annotated texts, as they can be identified by an xml:id attribute, by an xpath expression, or even by a range of text within some element. Where necessary it is possible to point at a number of nodes, e.g. by selecting them in an xpath expression that refers to some value for the xml:lang attribute.

In the case of parallel texts, the annotation target will typically be an entry in an alignment, that is, a link element part of the same document that houses the annotation. In that case the pointer will point not at the external document but at the link element that represents the alignment entry in the annotation document. The same holds true for any situation where what we want to annotate is not a resource or group of resources but a relation between resources.

A number of examples of annotations encoded as feature structures follows. First an example of the ‘reworking’ annotation type defined in Section 4.2. The ptr's ana attribute connects ptr and fs.

```
<ptr target=“#lg-1-001” ana=“#fs-001” xml:id=“a-1”/>
<fs type=“reworking” xml:id=“fs-001”>
   <f name=“charimage”>
      <note>about the <emph>imagery</emph></note>
   </f>
```

```
  <f name=“charrhet”>
      <note>about the <emph>rhetorics</emph></note>
  </f>
  <f name=“charvocab”>
      <note>about the <emph>vocabulary</emph></note>
  </f>
</fs>
```

Now an example of a more elaborate feature structure, involving the ‘conversation’ annotation type, also defined earlier:

```
<fs type=“conversation” xml:id=“fs-002”>
    <f name=“general_description”>
      <note>Conversation taking place. . .</note>
    </f>
    <f name=“duration”><numeric value=“15”/></f>
    <f name=“conversation_type”>
      <ptr target=“http://example.org/typology/a34”/>
    </f>
    <f name=“participants”>
      <vColl>
        <fs type=“participant”>
          <f name=“name”><string>John</string></f>
          <f name=“age”><numeric value=“19”/></f>
        </fs>
        <fs type=“participant”>
          <f name=“name”><string>Alice</string></f>
          <f name=“age”><numeric value=“34”/></f>
        </fs>
      </vColl>
    </f>
</fs>
```

Finally a number of examples of valid TEI P5 pointer targets. They point at, respectively, an element by its id attribute (lg-1-1), an element that is the second child of the fourth child of an element with id ‘emb01’, a range of text between elements, a range of text given by offset and length within an element, a point with in a text, and a series of lg elements having ‘fr’ as the value of its xml:lang attribute. It is clear these are very powerful facilities.

```
http://. . ./vae.xml#lg-1-1
vae.xml#element(emb01/4/2)
vae.xml#range(element(emb01/4/2),element(emb01/4/5))
vae.xml#string-range(element(emb01/4/2),5,3)
vae.xml#left(string-range(element(emb01/4/2),5))
vae.xml#xpath1(//lg[@xml:lang=‘fr’])
```

I have not yet looked into addressing sections of images in TEI P5. The *Guidelines* suggest using SVG for that purpose (Section 16.4.3).

4.5 TEI conformance

The proposed encoding scheme extends the TEI in a number of respects:

- It allows a number of existing TEI elements (note, ptr, date) to appear as atomic features, where the *Guidelines* do not permit them to occur;
- It allows head within linkGrp;
- It proposes a new element (dataSection) to store 'data'-like information;
- It proposes a new element (aboutDesc) to store data about the files that this document is about;
- It proposes a new element vDescr to describe the individual symbol values.

These modifications to the *Guidelines* have been documented in an ODD document (*Guidelines* Section 23.4). In terms of the section on TEI conformance (23.3), this makes the proposed encoding scheme a TEI extension. Documents conforming to this scheme can be transformed into TEI conformant documents, replacing the non-canonical note, ptr and date elements by string, replacing dataSection by text, and ignoring the aboutDesc, vDescr and the offending head. As in this transformation information is lost, documents conforming to this scheme are not TEI conformable.

4.6 TEI proxy documents

If one of the texts is not available as an XML document,[5] we may want to create a *TEI proxy document* for that text. A TEI proxy document I define to be a TEI document that contains enough of the structural aspects of the texts to be able to serve as a basis for defining an alignment between the texts involved and for annotations to be attached to. In the case of an emblem book a TEI proxy would contain structural divisions for the emblems, the mottoes, the epigrams, the pictures, the quotations, but probably not for each line in a poem, depending on the level of detail needed for the annotation. The TEI proxy would not contain a transcription of texts. It might contain pointers to a digital facsimile instead, for instance one digitized in Google Books.

A reason for creating a proxy document rather than a full transcription is that it, presumably, saves work. Having a full transcription is preferable, but in a world of limited resources a proxy document may be an acceptable compromise between having all and having nothing. Once we can attach annotations to portions of a proxy document we can see which annotations refer to the same document fragment. We can also create links between document fragments. If we have related the text structure to facsimile images, we can also fetch the page that contains the text that is being annotated.

Apart from the usual TEI element and attributes, a TEI proxy document will have one new element (proxy), and one new attribute (proxyFor). The proxy element is used as a placeholder for those elements not transcribed in the proxy document. The result attribute[6] may be used to give the name of the element which the proxy element replaces. The proxyFor attribute contains a URI and can be used on the sourceDesc element. It can be used to point to a document that this document proxies for—e.g. by pointing to an entry in a library catalogue or to a book in a digital library somewhere.

To create links between text structure and a facsimile elsewhere, we can use the TEI P5 facsimile element. The facsimile element contains surface elements, corresponding to pages or other objects that text is written or printed on. Zone elements can be used to identify regions in surfaces. The graphic element relates surfaces and zones to image files. The images can reside on a local server but might as well form part of a digital library elsewhere. From the text structure, the facs attribute points to the corresponding surfaces and zones in the facsimile.

Example:

Google Books contains a copy of Edmund Arwaker's translation of Herman Hugo's *Pia Desideria*.

A fragment of a TEI proxy document for this book might look like:

```
<TEI xmlns=“http://www.tei-c.org/ns/1.0”>
  <teiHeader>
    <fileDesc> . . .
      <sourceDesc
        proxyFor=“http://books.google.com/books?id=uyc3AAAAMAAJ”>
      </sourceDesc>
      . . .
    </fileDesc>
  </teiHeader>
  <facsimile>
    . . .
    <surface xml:id=“s150” ulx=“0” uly=“0” lry=“925” lrx=“575”>
      <graphic
      url=“http://books.google.com/books?id=uyc3AAAAMAAJ&pg=PA150 . . .”/>
    </surface>
    . . .
  </facsimile>
  <text>
    . . .
    <div type=“emblem” n=“15” xml:id=“e15”>
        <pb facs=“#s150” n=“150”/>
        <proxy result=“figure” xml:id=“e15.1”/>
        <div type=“quote” xml:id=“e15.2”>
          <proxy result=“quote”>
            <desc> Bible quotation</desc>
        </proxy>
    </div>
    <pb facs=“#s151” n=“151”/>
    <div type=“motto” xml:id=“e15.3”>
        <head><num>XV</num></head>
        <cit>
          <quote>How shall we sing the Lord’s Song
            in a strange land?</quote>
          <proxy result=“bibl”/>
        </cit>
      </div>
      <div type=“subscriptio” xml:id=“e15.4”>
        <proxy result=“lg”>
          <desc>Poem consisting of multiple stanzas</desc>
        </proxy>
      </div>
    </div>
    . . .
  </text>
</TEI>
```

External annotations can unambiguously identify the structural components of this text, e.g. by using xml:id attributes.

Other functions of such a proxy document might be to serve as a table of contents in digital libraries that do not provide one. In Google Books, for example, the generated table of contents is often nearly useless. A proxy document could also provide the basis for a 'jump to next section'-facility.

5 Conclusion

A prototype application called PAT (Parallel Annotation of Texts) has been developed in order to experiment with the proposed annotation scheme.[7] Using PAT, a researcher can create an alignment between two series of texts, display the aligned texts, create and modify annotation types and annotations, and export the annotations either in the annotation encoding proposed in this article[8] or as a TEI conformant document.

PAT's main window by default shows a text alignment, the available annotation types and the created annotations. The user can change window contents and layout. Using the File menu, the user can create, modify and export annotation sets. Once an annotation set is open, the user can (1) select an alignment or create a new one; or (2) create or modify annotation types; or (3) create or modify annotations. Things that can be annotated are alignment entries, the aligned text units, annotation types and annotations. The application uses the annotation types' definitions to create the fields where the user can enter data. For 'symbol' fields, the user can select one of the valid values.

A number of lessons were learned from building this prototype.

(1) The most important lesson is that even though the feature structure formalism may seem intimidating to an encoder without adequate tool support, with the aid of a suitable application feature structures and feature structure declarations are very expressive and easy to create. The user need not even be aware of the fact feature structures are being used to store his or her annotations. Feature structures provide a solid basis for a generically applicable TEI annotation format.

(2) There is no easy solution to the problem of modifying an annotation type that has already been used, that is, to change a feature structure declaration that some existing feature structures conform to. This is a problem that all systems for structured annotation share, but the problem is aggravated by the fact that the feature structure's type attribute, and the feature's name attribute, are repeated in the declaration and in each occurrence. The same problem would also occur, however, in more conventional TEI-XML encoding.

(3) Even using the best of tools, the creation of text alignments is never going to be intuitive. At present the user can create parameterized templates for the URI's of the text units that should be aligned; the application will then increase the parameters' values a specified number of times in order to create the desired number of alignment entries. An alternative procedure would be to manually relate and label a large number of text units. Creating the alignment seems an activity best left to the technically trained user, and an XML editor might be the most convenient tool for creating it.

(4) It is not self-evident how the aligned text units should be presented to the user. If the units are short poems, it is fairly simple to fetch these and display them next to each other. But it is hard, or impossible, to build a general-purpose display tool that will handle adequately any TEI text fragment (possibly including notes, images, references to other texts, perhaps modern translations, etc.) and create a suitable parallel display for an arbitrary number of these aligned texts.

Taken together these lessons suggest that, even though a general purpose tool for annotating text parallelism may not yet be feasible, the proposed annotation encoding format is sound. Feature structure declarations can provide a basis for the dynamic creation of windows for entering the corresponding feature structure data.

As to the alignment of parallel text units, the approach may be rather abstract,[9] but it does have its advantages. Explicitly attaching the annotation to a link between text units makes possible later filtering of annotations based on annotation target. For texts that do not draw much attention this filtering may not seem very important. For texts that we can expect to draw scholarly attention in the foreseeable future, it will become important to distinguish annotations on the texts themselves from annotations that address the relation between the texts and its adaptations. Future generations of scholars will continue to study many of the texts that we study today. Because we live in an electronic age, the output of their studies will remain accessible to later scholars. If we do not want later scholars to be overwhelmed by the scholarly results of the generations before them, we must assure that those results identify as precisely as possible the objects that they are about. When they say something about the relation between two text units, they should make that explicit. This article is a contribution to an infrastructure where they can do so.

References

Agosti, M., Ferro, N., Frommholz, I. *et al.* (2004). Annotations in digital libraries and collaboratories. Facets, models and usage. *Research and Advanced Technology for Digital Libraries: 8th European Conference, ECDL 2004, Bath, UK, September 12–17, 2004. Proceedings* (*Lecture Notes in Computer Science*, 3232), Springer, Heidelberg, pp. 244–55.

Bargeron, D. M., Gupta, A., and Bernheim Brush, A. J. (2001). *A Common Annotation Framework. Technical Report MSR-TR-2001-108*. Redmond: Microsoft Research.

Bath, M. (1994). *Speaking Pictures*. London: Longman.

Bath, M. (2005). Christopher Harvey's school of the heart. In Daly, P. M. (ed.), *Emblem Scholarship. Directions and Developments. A Tribute to Gabriel Hornstein* (Imago Figurata. Studies, 5b). Turnhout: Brepols, pp. 1–23.

Bird, S. and Liberman, M. (2001). A formal framework for linguistic annotation. *Speech Communication*, **33**(1–2): 23–60.

Boot, P. (2005). Advancing digital scholarship using EDITOR. *Humanities, Computers and Cultural Heritage. Proceedings of the XVI International Conference of the Association for History and Computing 14–17 September 2005*, Royal Netherlands Academy of Arts and Sciences, Amsterdam, pp. 43–8.

Boot, P. (2006). A SANE approach to annotation in the digital edition. *Jahrbuch für Computerphilologie*, **8**: 7–28.

Bradley, J. (2004). Tools to Augment scholarly activity: An architecture to support text analysis. In Buzzetti, D., Pancaldi, G., and Short, H. (eds), *Augmenting Comprehension: Digital Tools and the History of Ideas* (Officer for Humanities Communication Publication, 17) London: OHC, pp. 19–47.

Bradley, J. (2005). Documents and data: modelling materials for humanities research in xml and relational databases. *Literary and Linguistic Computing*, **20**(1): 133–51.

Bradley, J. and Short, H. (2005). Texts into databases: the evolving field of new-style prosopography. *Literary and Linguistic Computing*, **20**(Suppl. 1): 3–24.

Burnard, L. and Bauman, S. (2007). *P5: Guidelines for Electronic Text Encoding and Interchange*. http://www.tei-c.org/release/doc/tei-p5-doc/en/html/index.html (accessed 13 March 2008).

Carletta, J., McKelvie, D., Isard, A. *et al.* (2005). A generic approach to software support for linguistic annotation using XML. In Sampson, G. and McCarthy, D. (eds), *Corpus Linguistics: Readings in a Widening Discipline*. London: Continuum International Publishing Group.

Ide, N. and Romary, L. (2003). Encoding syntactic annotation. In Anne Abeillé (ed.), *Treebanks: Building and Using Parsed Corpora*. Dordrecht [etc.]: Kluwer Academic Publishers.

Ide, N. and Romary, L. (2004). A registry of standard data categories for linguistic annotation, *Proceedings of 4th International Conference on Language Resources and Evaluation* (Lisbon), pp. 135–38.

ISO TC 37/SC 4 (2006). *Language Resource Management—Feature Structures—Part 1: Feature Structure Representation*. Geneva: ISO.

Notes

[1] The Emblem Project Utrecht (http://emblems.let.uu.nl/) has digitized about 25 books of Dutch love emblems. The Emblem Project, together with Flemish researchers, was awarded a grant by the Flemish-Dutch Committee

for Dutch Language and Culture to study the reception of the emblem book *Pia Desideria* in the Northern and Southern Netherlands. The present article is part of the Huygens Institute's contribution to that research effort.

[2] Another data type that might be thought desirable is a reference to an external taxonomy, such as Iconclass. I'm assuming, however, that most of these taxonomies will shortly be reformulated as in terms of e.g. the W3C's upcoming SKOS recommendation. It will then be possible to refer to an entry in such a taxonomy using a URI. Thus a hyperlink data type should be sufficient.

[3] In the Glasgow Emblem Project, each emblem is a separate XML file. Studying the parallelism between several Alciato versions there would entail creating an alignment between as many files as there are emblems.

[4] The ana attribute 'indicates one or more elements containing interpretations of the element on which the ana attribute appears' (Burnard and Bauman, 2007; 17.2)

[5] One reason for this might be that there *is* an XML source for a digital edition, but it is inaccessible because of intellectual property concerns.

[6] Borrowed from the join element.

[7] The application was programmed in OpenLaszlo (client component) and Cocoon (server component).

[8] Actually it produces documents conforming to an earlier version of this schema. That version does not allow notes in feature structures, and it does not use the dataSection element. It stores annotations as link elements that link annotandum and feature structure. It implements only a subset of the desirable functionality as sketched in Section 4. It does not, for example, facilitate the annotation of text or image fragments. There is as yet no possibility to use externally defined annotation types. The application does not handle nested or repeating feature structures.

[9] As was remarked during the presentation of the encoding at the TEI@20 conference.