A Digital Edition of a Spanish 18th Century Account Book: Formalisation and Encoding

Abstract

In this, part two of a two part paper, we will discuss our approach to the formalisation of our document encoding approach, derived from software engineering, which treats of the three classes of a digital edition; the Logical, the Physical and the Interaction Classes. We specifically address our decision to use XML (Extensible Mark-up Language), not TEI (Text Encoding Initiative), as our encoding language. An argument is provided as to why TEI is unsuitable for function-based documents, this addresses both source integrity and the restrictive nature of TEI. TEI does not support our forward engineering approach, which allows us to simultaneously produce the model, the encoding and the software environment.

[1] Introduction

[2] We will discuss the formalisation and practicalities of the encoding of the Alcalá Account Book. Part one, *User Driven Digitisation*, has discussed the origins of the project and the manuscript. The source manuscript was chosen, encoded and made available in a web based, dual language, searchable and interactive environment. This environment was specifically designed to support the historian in the type of research directly prompted by the source. We will discuss the theoretical framework that lies behind this assertion, including the identification of Primary Use Cases. The remainder of the paper provides a thorough description of our digitisation decisions, the maintenance of source integrity, the restrictive nature of TEI and our conclusions.

[3] Digitisation Decisions

[4] (i) Objective: Primary Use Case Support

[5] Our project objective was to produce a digital edition that supported the user by supporting the primary Use Cases. We define primary and secondary Use Cases for any document. Primary Use Cases are content-driven and are derived from the document's original *raison d'être*, for example criminal, medical, financial or creative. They are also based on the meaning of the document. For instance, if the medical records of an Irish Hospital for one year were available, the primary Use Cases would answer questions such as, »how many patients were admitted during the year?«, »who were they?«, »what procedures did they undergo?«, »what care were they given?«, »how long was their average stay?« and »what was the average cost?«. Secondary Use Cases are driven by the interests of a specific audience and may not necessarily be linked to the document's original *raison d'être*, for instance, a linguist might wish to perform linguistic analysis of the records, but that is not contingent upon the document being a medical record.

[6] Naturally, secondary Use Cases should be supported where there are sufficient resources to do so. However, concentrating on the provision of support for primary Use Cases has two main, linked advantages: user-base size and longevity. Given an audience for any particular document, most of that audience will be interested in the content of the document, rather than any other aspect, such as the paper or language. For those seeking to study other aspects, many will wish to relate that aspect to the content, for instance a linguist studying language patterns will seek to link those to the historical and geographical context provided by the content. Regardless of the size of the user base population, we contend that supporting the greatest proportion of it will improve the edition's longevity. Therefore, in order to provide for the greatest number of users, for the longest time, the edition should support primary Use Cases.

[7] Our project was based on an account book, therefore our sample primary Use Cases include »how much was spent in total in 1798?«, »what was the relation between food expenses and student numbers across the period?«, »what was the relation between food expenses and time of year?« and »how much did Rector X sign off on over a given period, did that amount vary?«. We created a software environment that supports these primary Use Cases, and many others. The digital edition will no doubt prove to be a rich resource for codicologists, palaeographers and linguists, but no more so than any other machine-readable, internet-based text. Lavagnino states:

[8]
> Ten years ago, it seemed sufficient to say that you were going to create such an edition in the form of a hypertext: often with very little elaboration on just what the result would be or why it would be significant, as though the medium itself would automatically make such an edition significant. [1]

[9] The value of these editions lies in their machine readability and general availability, the value of this edition lies its usability.

[10] (ii) XML or XML/TEI [2]

[11] One of the decisions to be made in relation to the digital edition was whether this usability could be best supported by designing a custom schema and associated document-tree [3], like XML, or by exploiting an existing subset that provides a generic schema and document-tree, like XML/TEI. In this document we refer to TEI as XML/TEI because, in reality, even if not historically, it is a specialisation of XML, implemented as a schema (in P5). To provide the functionality required by our users we had to create an encoding and accompanying software guided by the logical and physical models and their methods. Only in this way could our software deliver the functionality of the original account book and manuscript, to be built upon by the functionality of the digital edition. We found that this approach preserved our source's integrity while preserving the usability of the original manuscript and providing additional, digitally driven, uses, for instance instant translation and keyword search. It also facilitated the creation of a high-quality, self-documenting, strongly-typed and validatable encoding.

[12] (iii) XML/TEI Encoding Guide

[13] The TEI was established in 1987, thus it pre-existed XML. It is a specialisation of their common ancestor, SGML. [4] Used by scholarly projects and libraries, [5] it seeks to develop, maintain, and promulgate »high-quality guidelines for the encoding of humanities texts« and then provide a language to embody those guidelines. [6] It is this attempt at standardisation that separates it from XML, which uses a custom approach. The TEI community is large, active and is a valuable source of support to its practitioners. Having a focus and a forum in which to discuss commonly occurring encoding difficulties and successes with researchers from a similar field is a major benefit to the digital humanities community. There is also, as Unsworth points out, »some benefit in expressing the consensus of those communities in things like standards and guidelines, if only to lay our cards on the table, and articulate in public the rules that we will, in any case, apply in private«. [7] However, these benefits are derived from the community involved in encoding. They are not benefits of the encoding language itself. There should be something specific to the language itself to recommend it, on technical grounds, as an encoding choice for a given project. In our case there were insufficient technical grounds to use XML/TEI.

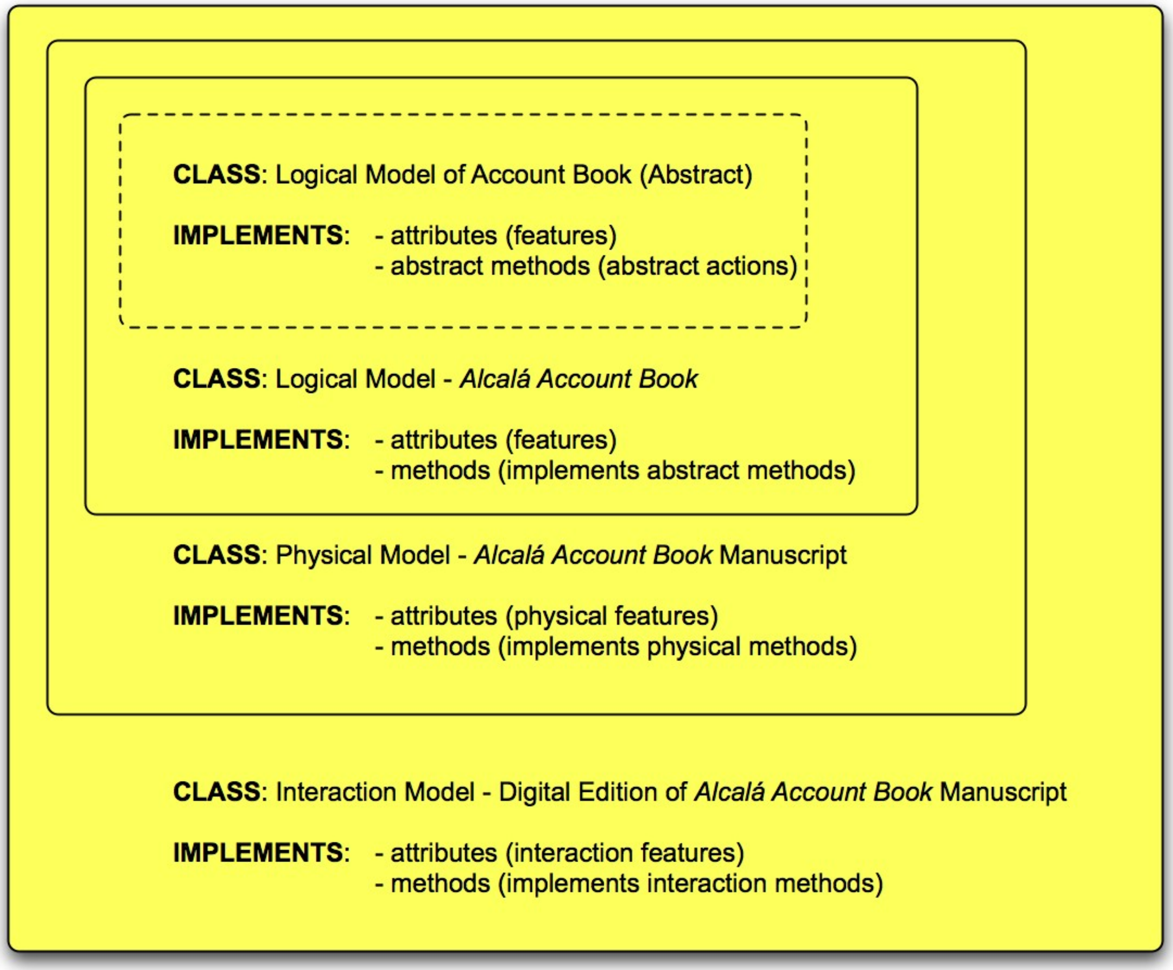
[14] When discussing the ability of XML/TEI to support our approach we have excluded the addition of non-required attributes and the customisation of the XML/TEI schema, as any custom XML Schema can be emulated simply by converting to XML/TEI. Given that XML supports transformations; add a basic header and transform using the rule `<div type= “XMLDocumentElementName”>`. The encoding would validate against the XML/TEI schema but it is clearly not an XML/TEI guided encoding. Our discussion is based on the possibilities provided for, and the approach encouraged by, the existing guidelines. We would argue that unless the document is driven and guided by XML/TEI it is not, in fact, XML/TEI. Indeed, viewing XML/TEI as an approach, not just an encoding language, is de facto within the TEI community; »TEI« references both the guidelines and the language used to express them.

[15] In our discussion we have also excluded from consideration the addition of metadata to the header in XML/TEI encodings. Addition of metadata in paragraphs can be ad-hoc and cannot be validated by automatic means. Apart from that, if the information is recorded in the original source (and not an editorial addition for instance) the encoder should be able to represent the information adequately using an appropriate schema, which would make it manipulable by a computer, not just by humans. This does not preclude the addition of any and all metadata in an encoding, just that metadata in the header that is not properly broken down and tagged. Header metadata should only contain data about the source, not data to make the source more manipulable.

[16] (iv) Classes, Attributes, Methods and Object Instantiation

[17] The digital edition of the *Alcalá Account Book* manuscript is a specialisation of a general account book. The notion of an account book, the heuristic discussed in the previous paper, that was used to inform the iterative segmentation process corresponds to an Abstract Class (in a software development sense). This Abstract Class is specialised through several other Concrete Classes until it can be instantiated as the digital edition of the *Alcalá Account Book* manuscript. Firstly, a logical model that corresponds to the notion of the *Alcalá Account Book* is described, indicating what it is, what it can be used for, and how to use it. Secondly, a physical model is added, corresponding to the *Alcalá Account Book* manuscript, what it can be used for, and how to use it. The last specialisation, an interaction model, provides a description of the features (attributes) of the digital edition of the *Alcalá Account Book* manuscript, of the user-requirements (methods) and how to implement them. An example of the structure of these classes is provided in Figure 1. The innermost class (Logical Model of Account Book) is an abstract class; all other classes are concrete. All of the concrete classes shown implement the enclosed classes' attributes and methods. An example of the attributes and methods belonging to each of the classes can be seen in Figure 2.

[18]


CLASS: Logical Model of Account Book (Abstract)
IMPLEMENTS: - attributes (features)
- abstract methods (abstract actions)
CLASS: Logical Model - *Alcalá Account Book*
IMPLEMENTS: - attributes (features)
- methods (implements abstract methods)
CLASS: Physical Model - *Alcalá Account Book* Manuscript
IMPLEMENTS: - attributes (physical features)
- methods (implements physical methods)
CLASS: Interaction Model - Digital Edition of *Alcalá Account Book* Manuscript
IMPLEMENTS: - attributes (interaction features)
- methods (implements interaction methods)

[19] Figure 1. Class structure for the Digital Edition of *Alcalá Account Book* Manuscript

[20] Once the classes have been defined it is possible to instantiate an object, in this case, the digital edition of the *Alcalá Account Book* manuscript. This object includes all of the attributes of all of the classes, that is, it encapsulates what tasks (methods) it can perform and it knows how to perform them. The methods are as inherent to its existence as the attributes; without them, it is useless.

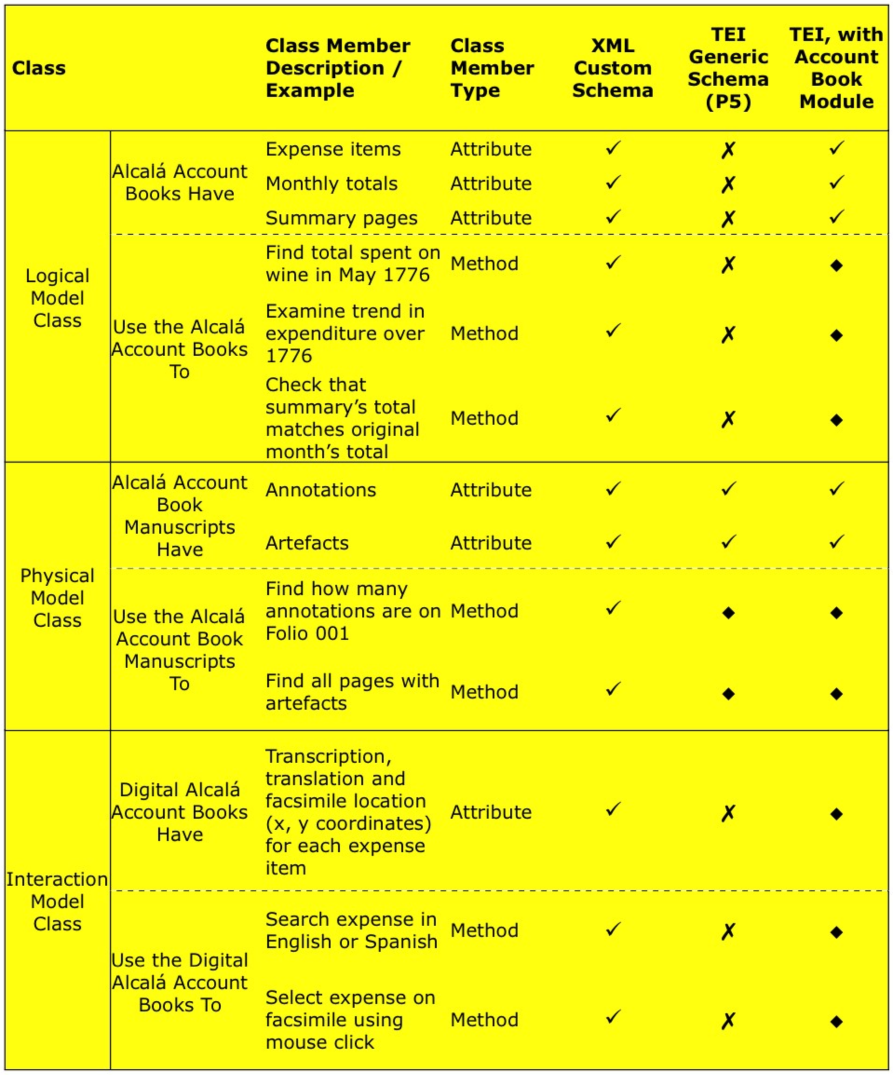
[21] (v) Use Cases and Supporting Class Functionality

[22] To discover the methods in the classes we employed Use Cases. [8] These are typical examples of the goals a user might wish to accomplish using the object, and the methods they would employ to achieve them, for instance, »calculate how much was spent on bread at the college in 1778« requires six methods. If the user speaks only English then they must (1) select the translation as the version to search, (2) enter »bread« as the keyword for the search, (3) examine the returned facsimiles from 1778, (4) select the entries in the journal pertaining to bread using the checkbox, (5) transfer the pertinent entries to the datasheet for calculation, and (6) switch to the datasheet view to read the total. This Use Case requires information from all three classes: the information on expenses and how to manipulate them is drawn from the Logical Class, information on the pages related to the expenses comes from the Physical Class, and the ability to search the translation and select appropriate expenses comes from the Interaction Class.

[23] The example above illustrates that all three classes in our Digital Edition must work together to deliver the functionality. Each class must be encoded so that it supports the class built upon it. Therefore, when the source (Logical and Physical Class) is encoded using mark-up, that mark-up must support the functionality that is to be delivered by the software (Interaction Class). XML/TEI does not provide for the functionality in the logical or Physical Classes to be exploited by the Interaction Class built upon them, nor does it consider the functionality of the Interaction Class. Therefore it was unsuitable for our project.

[24] The following table, developed in the analysis phase of the design approach, illustrates where there is a straightforward provision in XML or XML/TEI mark-up languages for the named classes, their attributes and methods.

[25]



Class		Class Member Description / Example	Class Member Type	XML Custom Schema	TEI Generic Schema (P5)	TEI, with Account Book Module
Logical Model Class	Alcalá Account Books Have	Expense items	Attribute	✓	✗	✓
		Monthly totals	Attribute	✓	✗	✓
		Summary pages	Attribute	✓	✗	✓
	Use the Alcalá Account Books To	Find total spent on wine in May 1776	Method	✓	✗	◆
		Examine trend in expenditure over 1776	Method	✓	✗	◆
		Check that summary's total matches original month's total	Method	✓	✗	◆
Physical Model Class	Alcalá Account Book Manuscripts Have	Annotations	Attribute	✓	✓	✓
		Artefacts	Attribute	✓	✓	✓
	Use the Alcalá Account Book Manuscripts To	Find how many annotations are on Folio 001	Method	✓	◆	◆
		Find all pages with artefacts	Method	✓	◆	◆
Interaction Model Class	Digital Alcalá Account Books Have	Transcription, translation and facsimile location (x, y coordinates) for each expense item	Attribute	✓	✗	◆
	Use the Digital Alcalá Account Books To	Search expense in English or Spanish	Method	✓	✗	◆
		Select expense on facsimile using mouse click	Method	✓	✗	◆

Legend: ✓ : supported; ✗ : not supported; ◆ : not supported without customising TEI specifically for our source and use cases.

[26] Figure 2. Sample attributes and elements of the Logical, Physical and Interaction Classes, and whether XML/TEI can be readily used to support them.

[27] Although these are only a sample of the attributes and methods of each class the table illustrates that XML can support the full suite of all the classes required to create a digital edition of the *Alcalá Account Book* manuscript. In fact, we guarantee that any source that can be adequately modelled as a tree can be encoded, along with its functionality, using XML (though XML is not restricted to this data structure). The diamond marking certain entries signifies that it is possible, though unlikely, for XML/TEI to straightforwardly support the encoding of that attribute or method. For instance, if the special interest group set up to create an Account Book module included people who wished to digitise this particular source for these particular use cases, it probably would support them. However, this is a special case. XML/TEI seeks to create a generic language for generating machine-readable text (TEI Consortium 2007b). Given that it cannot, and does not seek to, forecast user-requirements it does not support the provision of functionality. If this particular source and user group were not envisaged then XML/TEI would probably fail to cater for the methods, and maybe even the attributes of the Logical Class. It would definitely fail to cater for the attributes and methods of the Interaction Class as these are entirely derived from the user-requirement, the primary Use Cases, which inform how the source should be described. Therefore, in the universal case, XML/TEI fails to straightforwardly provide for functionality.

[28] (vi) XML/TEI Cannot Support the Logical Class

[29] We consider the *Alcalá Account Book* manuscript to be an account book, which happens to be captured on a manuscript; it is not a »manuscript with specialisations«. By encoding the structure and meaning of the content of the document, the accounts in the account book (Logical Class), we are accurately modelling our source and also supporting the primary Use Cases, and in turn the user. The Logical Class is the basis for the digital edition, it captures the meaning of the content. It should be the primary model (rather than the physical) if the primary Use Cases are to be fulfilled. In order to represent the Logical Class in our encoding we required firstly that the structure of our schema could be freely created to match the tree we had modelled on the logical structure. This is not possible using an XML/TEI P5 generic schema as its structure is pre-defined. Secondly, we required that the element names that described the Logical Class could be drawn from our heuristically informed understanding of the *Alcalá Account Book*. Again, this was impossible using pre-defined element names, especially as the source did not have an associated tagset in XML/TEI. Lastly, it was imperative that the encoding language supported the methods associated with the Logical Class, but XML/TEI currently does not.

[30] The software support that was derived from the encoding of the model of the Logical Class has been previously described in some detail. Suffice to note here that the custom structure and element names contributed to supporting the methods, as did some elements that were explicitly included to provide aid to the software in processing and presenting the data, for instance `<pageID>`. The software, in turn, provides support to the user. In this way we can trace the fulfillment of the objective, that is supporting the user, back through the software and the encoding to the Logical Class, expressable only using a custom language, XML. XML allows us to create the solution to solve the problem, rather than fitting the problem to the solution provided by the tool, XML/TEI.

[31] Source Integrity

[32] (i) Digital Editions

[33] From our perspective, informed by computer science methodologies, it is important that the encoding closely resembles the *Alcalá Account Book* manuscript (Logical and Physical Classes) so that we ensure that as many future requirements as possible can be supported without having to substantially rewrite the encoding. This involves creating an encoding that preserves the attributes and methods of the original source, and the methods allow us to meet these future needs. Source integrity is best maintained and addressed by using a metalanguage closely related to the source, rather than a generic, prescribed encoding language. In the case of facsimile editing, from digital images to diplomatic transcriptions, »the scholar's object is to provide as accurate a simulation of some particular document as the means of reproduction allow«. [9] Therefore, editors should seek to exploit the inherent slide in natural language rather than introducing distance between them and the text by increasing the machine-like quality of the encoding language.

[34] Lavagnino, among others, states,

[35]
> It is true that there are no chapters in this book or in the Guidelines on encoding cookbooks, newspapers, guidebooks, [...]; there just hasn't been space or occasion to discuss them specifically. For the most part these particular genres can be readily handled using TEI's provisions for encoding prose texts, with the addition of some other elements for their distinctive features. [10]

[36] We refute the notion that cookbooks, newspapers, et cetera, are merely prose texts with »some distinctive features« – this particular account book is clearly highly structured with its own elicitable rules that guide internal consistency. From a humanities researcher’s perspective, surely it is more desirable to closely model those actual structures and their meanings, rather than to try to fit them to structureless or inappropriately structured prose tags?

[37] Using a single witness we created a facsimile and diplomatic edition. We understand that our source is not self-describing [11] and that there are non-trivial issues that relate to the transformation involved in the first encoding of the work, that is, the instantiation of the *Alcalá Account Book* that occurred when it was first written down. [12] Essentially, this means that we recognised that the edition was interpretive. This is not in opposition to the close modelling of the document: recognising editions as interpretive does not render them pointless or less scholarly, and so rigorous modelling and strong encoding remained desirable.

[38] If every edition (diplomatic or critical) and every encoding is interpretive then the use of an encoding language that allows for self-documentation (contemporaneous with its creation and use), and for the explicit recording of the thought processes behind the encoding, is imperative. XML/TEI encourages the use of the `<header>` element to document the evolution of the encoding. However, this approach fails to document the thought-processes involved in any particular encoding as it lacks the custom document that is the result of the XML encoding process.

[39] Self-documentation and thought-process analysis are facilitated by examining the schema and resultant encoding, which both act as an unambiguous record of the interpretation of the document. The schema tells us both how the segments of the document were interpreted (as expenses, as signatures, as introductions, as dates, et cetera) and

the hierarchical structure and internal rules the encoder considers the document to embody. In this way the encoding itself acts as a framework for scholarly discussion, embracing the interpretive and subjective nature of mark-up. XML/TEI does not provide for this as the structures are prescribed, the internal rules of the document are not considered, and the elements cannot be custom named. Burnard calls for the creation of »an uncritical edition[,] one which does not attempt to settle controversy, but to ignite it«. [13] The use of XML/TEI as a standardising language promotes the false notion of authoritative objectivity, it fails, as Burnard puts it, »to problematize the textuality that a traditional critical edition tends to gloss over« [14] and thus reduces the document to pre-defined, pre-imagined, prescribed elements and structures.

[40] There are additional issues associated with encoding the facsimile and diplomatic editions for computer hosting, manipulation and use as a digital edition. Firstly, given that the digital edition is dependent on a tree, which instantiates only one possible model that could be used to describe the Logical Class of the source, the choice of tree structure impacts on the accuracy of the representation, and thus, the edition. [15] Secondly, the tag-names used to contextualise the data are themselves open to interpretation, thus introducing more ambiguity. Lastly, the chosen encoding language, in this case XML, impacts on the presentation of the source as it enforces certain rules and constraints. As indicated by Vanhoutte,

[41]
> […] the practice of creating an edition with the use of text-encoding calls for explicit ontologies and theories of the text which do generate new sets of theoretical issues. Maybe they are not different sets of editorial issues, but they are certainly new sets of textual issues such as problems of document architecture, encoding time, etc. [16]

[42] Given all of these encoding decisions, each of which has ramifications for the creation of a facsimile or diplomatic edition, it would seem prudent to attempt to minimise the impact of the encoding language, element names, and structuring decisions by modelling the source as closely as possible. It is only by using a custom structure and custom element names that we were able to achieve this, thus XML was used to minimise these additional editorial issues.

[43] (ii) Exploiting Custom Naming in XML

[44] Definition-slide is inherent to all mark-up, that is, there is a difference between what is recorded on the page and how we mark what is recorded on the page. This is because of the heteroglot nature of language [17] and its attendant interpretation, and it is this inherent slide that allows a computer to process human language at all. [18] XML mark-up allows us to provide entity (element) naming capabilities similar to noun or verb sense in natural language. We can then use these capabilities to create element names (tags) to contextualise data, which is itself often represented as text on a page. Common-sense would dictate that an item of expense, recorded in an expense account, should be marked as `<expenseItem>`. This mark-up allows a human user of the encoding to perform any interpretation that is necessary to make sense of the data contained in the tag:

```
[45]        <expenseItem>
[46]                <description>bread</description>
[47]                <amount>€1.50</amount>
[48]        </expenseItem>
```

[49] By supporting the creation of custom tags XML helped to overcome the additional editorial difficulties associated with the digital edition of our source. We found that XML/TEI offered no such help as it fails to exploit the benefit associated with custom naming.

[50] (iii) Creative and Functional Documents

[51] Far from supporting all humanities texts, XML/TEI is, itself, a custom schema created for those creative documents and formats it explicitly seeks to describe (novels, poems, codices, manuscripts), for an intended audience it implicitly foregrounds (librarians, editors, literary scholars, palaeographers). In fact, XML/TEI has created a tagset that is capable of describing most creative texts; there is almost always some tag that can be used to adequately describe the devices used in a novel or poem. Though that tagset is still not capable of describing the meaning held within creative works. This paper, however, is concerned with functional documents. XML/TEI is of little use when seeking to describe a functional document, which, by definition, encapsulates functionality and has a number of predetermined uses that are linked to their primary Use Cases.

[52] (iv) Machine Readable Text

[53]

Given that machine-readable texts preserve (in some sense) their sources and make them available to a wider audience there is some merit in arguing that even allowing for them to be rendered does provide more usage for them than if they were held in archives and subject to conservation. However, hope that this standardised language might meet all user requirements, current and future, is unfounded, and cannot be supported in the generic encoding. XML/TEI manages to provide a tag-set for creating machine-readable texts that help to at least make the sources available, and often additionally provide automated search functionality. Perhaps this is all that can reasonably be expected of a global drive to create a generic encoding approach. The richness of XML/TEI in relation to this type of source and its own, particular, audience is clear; as is its lack of an adequate tagset for those sources and users outside these two spheres. This is evidenced by the fact that the features of our source it can capture are those that instantiated account books have in common with novels, poems, codices and manuscripts, that is, those attributes, and perhaps (but not definitely) methods, derived from the Physical Class.

[54] XML/TEI covers only a subset of documents, formats and user-requirements that are of interest to humanities scholars. Without extensive customisation it cannot even provide a machine-readable version of all humanities documents:

[55]

> Because the TEI Guidelines must cover such a broad domain and user community, it is essential that they be customizable: both to permit the creation of manageable subsets that serve particular purposes, and also to permit usage in areas that the TEI has not yet envisioned. Customization is a central aspect of TEI usage and the Guidelines are designed with customization in mind. [19]

[56] While XML/TEI openly relies on customisation and extension to cover areas, »that the TEI has not yet envisioned«, it is worth noting that XML/TEI can only ever cover a small subset of areas. It cannot create a tagset for every record, functional document, novel, poem, manual, etc. in existence, let alone combine this with every user-requirement of all research audiences. Even exhaustive, on-going work will never create a tagset that describes the majority of humanities texts. This is widely acknowledged but the general response is »any useful markup scheme must therefore be extensible«. [20] We disagree; any useful mark-up scheme must therefore be custom designed. Relying on customisation and extension to fill in the significant gaps is, in effect, relying on people to use XML (translated into XML/TEI, but not guided by XML/TEI). Our source (an account book) had no associated module in XML/TEI and thus we would have had to rely on significant customisation to preserve our user driven approach. We therefore considered it unsuitable as the encoding language for this digital edition.

[57] Restrictive Nature of XML/TEI

[58] Once an encoder becomes comfortable and adept at eliciting a document’s structure it is much easier to freely encode the document using XML. Furthermore, at this stage, the user-requirements should be used to inform the encoding. The XML/TEI Guidelines become restrictive – the encoder ends up wrestling with permitted nesting structures and tag-names that do not quite match the encoder’s tree model of the source, if infact they have created one, or how this model will be used. Time is wasted learning and/or finding the appropriate tags rather than capturing the document correctly. As the edition could be supported in XML, without placing extra demands on the resources, it was chosen as the encoding language.

[59] (i) Problems Associated with Generic Element Names

[60] XML vocabularies cannot define the intended use for element names to the extent that those definitions can be enforced (unless an enumeration or pattern is used, which is only applicable to a tiny subset of elements).

[61]

> It is up to the creators of XML vocabularies (such as these Guidelines) to choose intelligible element names and to define their intended use in text markup. That is the chief purpose of documents such as the TEI Guidelines. From the need to choose element names indicative of function comes the technical term for the name of an element type, which is generic identifier, or GI. [21]

[62] This is problematic because although one might assume that everyone understands what »title« means and even give it a specific definition (the XML/TEI guidelines define `<title>` as, »contains the full title of a work of any kind.« [22] different encoders will, quite legitimately, apply that element tag to different elements in the same poem or novel. The same is true of defining the nesting elements of a given element. According to the Guidelines `<title>` can contain 171 other elements, including other `<title>` elements. [23] XML/TEI practitioners might argue that this flexibility allows an appropriate structure to be made explicit by the encoder. In reality, however, this leaves the description of any given document open to many variations – even from one seemingly easily definable element. If a more refined element name, such as `<expense>` could be used when contextually appropriate this problem would be minimised, though never eliminated. XML provides for the custom naming of elements so that their function as mark-up is clearer, it also provides strongly typed validation on the custom schema to ensure, as far as possible, that the encoding is standardised, regardless of who performs the mark-up. XML/TEI did not provide this, therefore XML was chosen to support the digital edition.

[63] **(ii) Interoperability**

[64] XML is designed for interoperability, which is derived from using a standard mode to store information, not from using the same set of words to describe various sources. By studying a schema that describes how the data were stored, a skilled software engineer can use XSLT, or other translation mechanism, to modify the attendant XML document to work with, and in, their own software engineering environment. The schema acts as a blueprint for the encoding mechanism used for the data, therefore, it must be specific and easily understandable. By using custom naming in the XML tags the schema is made specific and easily understandable to the software engineer.

[65] As demonstrated in our paper »Part 1: Visualisation and Information Architecture«, XML allows for the dynamic generation of the interface and interaction mechanisms of the software application. This XML could be harnessed by another software application to dynamically generate other contextually appropriate interfaces and interaction mechanisms. Our XML can only provide this interoperability because the custom schema allowed us to create this design; a generic schema could not.

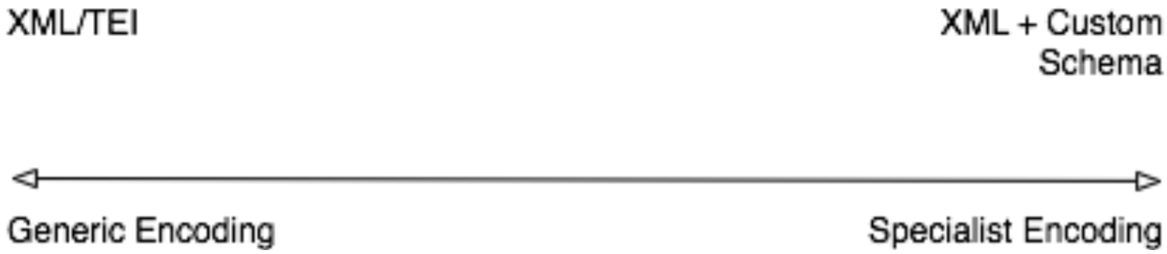
[66] It is very complicated to create a translation mechanism for an XML/TEI document encoding because the generic schema provides no way to extract the rules of the encoding (all rules for all encodings are contained in it). This does not remove the need for translation once the encoding document is transmitted to another processing software application. Instead it introduces ambiguity, for instance, as demonstrated above, the tag `<title>` can be used as the title of a book, or of a poem. The receiving software does not know which way it has been used. Apart from translation mechanisms, namespaces are employed by XML to surmount this obstacle, they qualify which domain the `<title>` element belongs to, for instance one that uses `<title>` for books, or one that uses `<title>` for poems. XML/TEI provides no such method, making translations more difficult. There is no additional interoperability (current or future) that can be, or has been, gained by using XML/TEI instead of XML. This is despite the additional time resources it requires to learn XML/TEI rather than implement XML.

[67] **(iii) Sacrifice Specialist Encoding for Pan-Comparison of Documents**

[68] We have already documented our reasons for closely modelling the source, and believe that this adds to the scholarly value of the digital edition. However, there is another way to value an encoding. After discussing the myriad complexities inherent in various texts and how mark-up can make them explicit, Burnard states,

[69]
> it now should be apparent why the availability of a single encoding scheme, a unified semiotic system, is of such importance to the emerging discipline of digital transcription. By using a single formalism we reduce the complexity inherent in representing the interconnectedness of all aspects of our hermeneutic analysis, and thus facilitate a polyvalent analysis. [24]

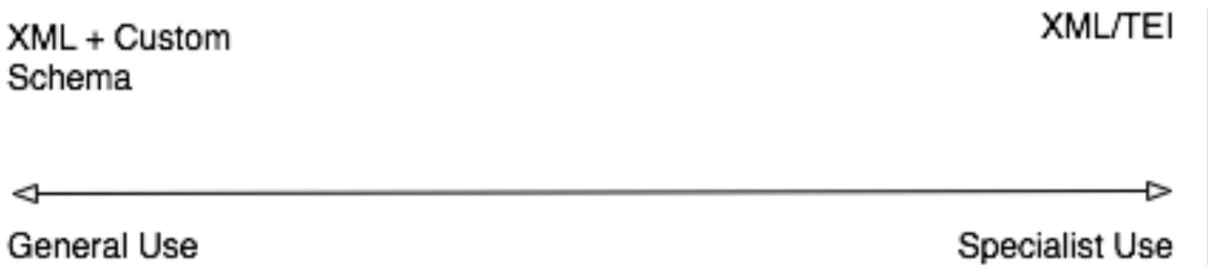
[70] An encoding that supports polyvalent analysis opens up a document to more than one kind of analysis, in other words, it supports scholars from a variety of backgrounds. Burnard’s assertion should mean that XML/TEI supports the study of many disparate, technologically unconnected, documents by one scholar, for one research aim. A popular contention is that XML/TEI helps scholars by allowing for the pan-comparison of various XML/TEI-encoded documents; even if certain subtleties are lost in the standardised encoding there is sufficient value in being able to conduct a pan-comparison to offset this, as in figure 3.



[71] Figure 3. View of XML/TEI supporting pan-comparison

[72] In reality, we have discovered no XML/TEI-dependent tools that allow for semantic analysis across multiple XML/TEI encoded texts. If pan-comparison of documents is to be supported by examining the actual encoding then, as previously argued, the fully customised schema and encoding is by far the most appropriate documenting system to examine. The reality of the situation is demonstrated in figure 4.

[73]



[74] Figure 4. XML/TEI does not allow for general use

[75] The notion of pan-comparison of documents, supporting polyvalent analysis across documents, is one of the main implied benefits of XML/TEI; allowing for the exchange of document encodings across projects and repositories. If this were true, we should now be seeing the fruits of decades of XML/TEI encoding as scholars should be able to compare disparate documents under a variety of lenses – not just for metadata but for primary Use Case material. It is reasonable to expect that software tools developed to interrogate semantically one XML/TEI encoded collection or document could be applied to another, imported, document encoding. From a review of published literature we have been, thus far, unable to locate any example of this having occurred without prior joint planning by the project/repository designers. In fact, any example of either of these expected outcomes (pan-comparison of documents or cross-project document-encoding exchange) would be welcomed by all in the digital humanities community.

[76] (iv) Overcoming XML/TEI Shortcomings

[77] It is possible to manipulate XML/TEI in order to overcome its drawbacks. As discussed above, one can change custom elements to `<div type = “XMLDocumentElementName”>` and perform the manipulation and extraction of meaning on the attribute names. There are three main problems with customising: firstly, it is not really XML/TEI. Secondly, it is difficult to do. Lastly, elements in the original XML that required attributes cannot be given those attributes because they are now attributes themselves. They cannot be broken down into their own sub-elements. In other words, you cannot create `complexTypes` (`complexTypes` are elements defined in an XML schema to contain other defined sub-elements), which are required to accurately model the interface and interaction between the Logical, Physical and Interaction classes of our digital edition.

[78] It is important to note that there is a distinction between what is possible, and what is desirable and appropriate. Of course it is possible to use XML/TEI to express some of the various aspects of our document but it is neither appropriate nor desirable, for the reasons we have described. It is also possible to use XML/TEI to describe the manuscript and then add an account book module using the Roma schema generator, but we do not believe that there are any benefits to be gained by using a standardised encoding – it certainly does not facilitate pan-comparison of documents or cross-project document exchange.

[79] (v) Schema for Validation

[80] XML/TEI’s generic schema approach, and even the work-around mentioned above, precludes proper validation of the encoding. Proper validation is provided only by a custom schema. An XML schema is used to define the elements and attributes of an encoding document. It further defines the order and occurrence of the elements and attributes, and the type of the elements, including enumerations or patterns for the data contained in them. [25] Validating against an XML schema provides assurance that the encoding document is accurate. If a generic schema is used this check cannot be performed, only very basic validation is possible. The designer loses control of the document and, if the source is of any significant size, there is no way to perform rigorous checking of the encoding document. Validation of our encodings was considered to be of paramount importance to the functionality and quality of the digital edition, consequently the custom schema of XML was chosen over the generic schema of XML/TEI.

[81] (vi) Forward Engineering into XML/TEI

[82] While we consider TEI/XML to have a number of flaws, as detailed above, the fundamental issue is that TEI/XML does not support our user driven approach. We use a software engineering model where we follow a forwarding engineering process that provides both the software environment and the encoding model. Forward engineering from the Unified Modeling Language (UML) into a full software solution and encoding is a good approach that provides good functionality for the user.

[83]

One alternative is to use a pre-existing text-encoding framework, such as XML/TEI. The problem with this is that it only provides circumscribed encoding guidelines, and not tools to create an environment, so this becomes a significant design constraint.

[84] It is possible to generate a model based on XML/TEI that is flexible, if you are just concerned with encoding. However, this is also constraining in that you are modelling structure and meaning before you engage in a Use Case Analysis process, which results in fixed use case support.

[85] As part of the editorial process for this paper, we engaged in discussion with XML/TEI experts and we have encoded a validating XML/TEI version of some of the pages of the account book. In order to »make it work« we had to re-purpose some of the encoding tagset. This, in and of itself, is a violation of the TEI guidelines. However, there is one last approach open within the XML/TEI paradigm: customisation.

[86] We have been encouraged to create an Account Book module using the Roma customised XML/TEI schema generator and are engaged in the process. However, there are additional drawbacks to using this approach, while there appear to be no compelling benefits. Firstly, this requires more work than forward engineering straight into the encoding and software environment: it is an additional, and unnecessary, step. Secondly, the next time we need to use the Guidelines for an account book from a different time or place, we will have to again forward engineer into our encoding and environment, and then re-customise the Account Book module.

[87] In fact, the range of possible permutations offered by the combination of project researcher, source document/data and research community is massive. While XML can support the forward engineering process for all mark-up suitable permutations, XML/TEI cannot.

[88] Conclusions

[89] The differences between the language registers associated with computer science and humanities sometimes manifest themselves in misunderstandings and mis-communication: some disambiguation would help to clarify matters. General references to »standards« within the discourse of Digital Humanities and Humanities Computing, without doubt, should be qualified. »Standard« in computer science usually refers to a technology with a strong, independently verified, technical specification – in this register XML is a standard, defined by the World Wide Web Consortium, but XML/TEI is not. »Standard« can also refer to an acceptable level of encoding, as in the phrase, »this encoding is up to standard«. Depending on the needs of the user, both XML/TEI and XML can deliver acceptable levels of encoding and can both thus be considered to support a »standard« of encoding. Lastly, »standard« can be used to describe something that is usual, regular or normal. XML/TEI may be considered by the TEI community to be the usual text encoding language of the humanities, but it is not exclusively the conduit for acceptable levels of encoding. In fact, it is our opinion that it acts as a barrier to acceptable encoding, and is not a technical standard.

[90] We have demonstrated that the appropriate encoding for this class of document (that is functional documents such as medical records, bank statements and police reports) and this group of users is a user-driven language and custom schema, expressed in XML. We have further demonstrated that XML/TEI does not straightforwardly support user-requirements derived from our functional document and thus is not suitable for this approach, encoding or digital edition.

[91] Notwithstanding the popularity of XML/TEI, ready to use with its standardised schema, guidelines, regular training sessions, and active community, we remain unconvinced that XML/TEI offers compelling benefits that are not already available to the community through the use of good, rigorous, XML. We believe that if this community is to conduct humanities computing, rather than TEI research, then alternative, and potentially disruptive, encoding approaches should at least be considered. This assertion, while likely to be contentious, stems from regular public declarations that XML/TEI is the *de facto* encoding standard, which we believe inhibits other possible avenues of digital humanities development and humanities computing research into document and data encoding.

[92] We believe that the XML/TEI approach should at least be broadened to support the entire project life-cycle. Currently the guidelines foreground the use of XML/TEI as a solution, in isolation from accompanying software, and before the problems of the project are even enunciated. Our approach allows us to forward engineer an entire solution: we analyse our project objectives and then design our encoding in tandem with our software, both of which are derived from a single, unified model of the document, process and usability. We found that XML/TEI-guided encoding was not compatible with our engineering methodology without extensive customisation; it significantly curtailed our ability to meet immediate and future user requirements for the development of this digital edition. We recommend that the TEI Guidelines provide support for implementing XML/TEI within a whole-project context, and that appropriate revisions to the TEI Guidelines are undertaken. We believe that this will encourage digital humanities projects to engage in formal design, encoding and development processes. In the meantime, our modelling process, custom encoding and supporting software, allow us to meet the needs of our team (the developers, encoders and humanities scholars) and end-users. It also enables us, the humanities computing community, to exploit this wonderful source for new research.

[93] Acknowledgement

This project was jointly funded by the Higher Education Authority’s PRTLI Cycle 4 and the National University of Ireland, Maynooth’s President’s Fund. We would like to thank the staff of the Russell Library for their continued support. We would also like to thank the editors for insightful comments on the draft paper.

John G Keating/Aja Teehan/Damien Gallagher/Thomas O’Connor
An Foras Feasa
National University of Ireland, Maynooth
Maynooth, Co. Kildare, Ireland
@

(17.06.2009)

Bibliography

Bakhtin, Mikhail
1981 The Dialogic Imagination: Four Essays. Michael Holquist (ed): Caryl Emerson and Michael Holquist (translation). Austin and London: University of Texas Press.

Burnard, Lou
2006 Is Humanities Computing an Academic Discipline? or, Why Humanities Computing Matters [1].

Card, Stuart K./Moran, Thomas P./Newell, Allen (ed)
1983 The Psychology of Human-Computer Interaction. Hillsdale, NJ: Lawrence Erlbaum Associates.

Lavagnino, John
2006 When Not to Use TEI. In: Lou Burnard/Katherine O’Brien O’Keeffe/John Unsworth (eds): Electronic Textual Editing. New York: Modern Language Association of America, p. 334–338 [2].

Lavagnino, John quoted in Vanhoutte, Edward
2006 Prose Fiction and Modern Manuscripts. Limitations and Possibilities of Text Encoding for Electronic Editions. In: Lou Burnard/Katherine O’Brien O’Keeffe/John Unsworth (eds): Electronic Textual Editing. New York: Modern Language Association of America, p. 161 [3].

McGann, Jerome/Buzzetti, Dino
2006 Critical Editing in a Digital Horizon. In: Lou Burnard/Katherine O’Brien O’Keeffe/John Unsworth (eds): Electronic Textual Editing. New York: Modern Language Association of America, p. 53–73 [4].

Redding, Paul
2008 Georg Wilhelm Friedrich Hegel. In: Edward N. Zalta (ed): The Stanford Encyclopedia of Philosophy (Fall 2008 Edition) [5].

Renear, Allen/Mylonas, Elli/Durand, David
1996 Refining our Notion of What Text Really Is: The Problem of Overlapping Hierarchies. In: Nancy Ide and Susan Hockey (eds): Research in Humanities Computing 4. Oxford University Press, p. 263 - 280.

Sperberg-McQueen, Michael
1991 Text in the Electronic Age: Texual Study and Textual Study and Text Encoding, with Examples from Medieval Texts. In: Literary and Linguistic Computing 6 (1). ALLC, p. 34–46 [6].

TEI Consortium
2007 TEI: Goals and Mission [7].

TEI Consortium
2007 TEI: Text Encoding Initiative [8].

TEI Consortium
2008 TEI: Customization [9].

TEI Consortium
2008 TEI: Projects Using the TEI [10].

TEI Consortium
2009a TEI: A Gentle Introduction to XML [11].

TEI Consortium
2009b TEI: Guidelines for Electronic Text Encoding and Interchange, P5 [12].

Unsworth, John
2002 Electronic Textual Editing and the TEI. Annual Convention of the Modern Language Association [13].

Vanhoutte, Edward
2006 Prose Fiction and Modern Manuscripts. Limitations and Possibilities of Text Encoding for Electronic Editions. In: Lou Burnard/Katherine O’Brien O’Keeffe/John Unsworth (eds):): Electronic Textual Editing. New York: Modern Language Association of America, p. 161–180 [14].

Websites

[1] <http://users.ox.ac.uk/~lou/wip/hc.html> (05.06.2009).

[2] <http://www.tei-c.org/About/Archive_new/ETE/Preview/lavagnino.xml> (05.06.2009).

[3] <http://www.tei-c.org/About/Archive_new/ETE/Preview/vanhoutte.xml> (05.06.2009).

[4] <http://www.tei-c.org/About/Archive_new/ETE/Preview/mcgann.xml#body.1_div.5> (05.06.2009).

[5] <http://plato.stanford.edu/archives/fall2008/entries/hegel/> (05.06.2009).

[6] <http://llc.oxfordjournals.org/cgi/content/abstract/6/1/34> (05.06.2009)

[7] <http://www.tei-c.org/About/mission.xml> (05.06.2009).

[8] <http://www.tei-c.org/index.xml> (05.06.2009).

[9] <http://www.tei-c.org/Guidelines/Customization/index.xml> (05.06.2009).

[10] <http://www.tei-c.org/Activities/Projects/> (05.06.2009).

[11] <http://www.tei-c.org/release/doc/tei-p5-doc/en/html/SG.html> (05.06.2009).

[12] <http://www.tei-c.org/release/doc/tei-p5-doc/en/html/ref-title.html> (05.06.2009).

[13] <http://www3.isrl.uiuc.edu/~unsworth/mla-cse.2002.html> (05.06.2009).

[14] <http://www.tei-c.org/About/Archive_new/ETE/Preview/vanhoutte.xml> (05.06.2009).

[15] <http://www.w3.org/DOM/#specs> (05.06.2009).

[16] <http://www.w3.org/MarkUp/SGML/> (05.06.2009).

[17] <http://www.w3.org/XML/> (05.06.2009).

[18] <http://www.w3schools.com/schema/schema_intro.asp> (05.06.2009).

---

[1] Lavagnino in Vanhoutte (2006: 161).

[2] TEI Consortium: Text Encoding Initiative (TEI) [8].

[3] World Wide Web Consortium: Document Object Model (DOM) [15]

[4] World Wide Web Consortium: Standard Generalized Markup Language (SGML) [16].

[5] TEI Consortium: 2008.

[6] TEI Consortium: 2007.

[7] Unsworth 2002.

[8] Card, Moral, Newell: 1983. The formalisation of the processes involved in usability analysis sprung from work undertaken by Card, Moran and Newell in their book, The Psychology of Human Computer Interaction (1983). They used the acronym, GOMS (Goals, Operators, Methods and Selection rules) to classify the interaction between human and computer.

[9] McGann/Buzzetti (2006: 53).

[10] Lavagnino (2006: 334).

[11] In relation to »self-identical«: Hegel’s view (see Redding’s entry on, Georg Wilhelm Friedrich Hegel, The Stanford Encyclopedia of Philosophy [Fall 2008 Edition]) is that nothing can be equal to itself, or self-describing, as everything is self-contradictory, mutable and connected to other things, continuously changing over time. Only abstractions can be motionless and unchanging; concrete realities exist in a time and place and thus can only be understood in relation to other things. Therefore, a document can only be understood within a material context, and an encoding, designed using that understanding, will also be informed by that context. While it may seem obvious that the encoding is not equal to the document, the document is not, in fact, equal to itself either.

[12] McGann/Buzzetti 2006.

[13] Burnard 1999.

[14] Ibd.

[15] Renear provides an in-depth discussion on the problems associated with viewing texts as ordered hierarchies of content objects (OHCO) that can be represented in tree models. For further on-line reading see Renear et al (1996).

[16] Vanhoutte (2006: 161).

[17] In his essay, *Discourse in the Novel*, Bakhtin posited the notion of language as heteroglot, that is that it is written, and eventually read, in a time and place, and that that time and place inform both the creation and understanding of the language (Bakhtin 1981). It is neither immutable nor directly interpretable as it is fundamentally fluid and changing. Therefore, words that are used to describe something will ›slide‹ away from their authorially given meaning as they are being read this slide gets greater with time and distance, and with the use of metalanguage (languages used to describe languages, including metalanguages themselves).

[18] McCann/Buzzetti 2006. They point out that the approximating metalanguage allows the encoder to describe something non-quantifiable or definitive in a way that a computer can manipulate quantifiably and definitively.

[19] TEI Consortium 2008b.

[20] Sperberg-McQueen: 1991.

[21] TEI Consortium 2009.

[22] TEI Consortium 2009b.

[23] Ibd.

[24] Burnard 1999.

[25] Apart from the tutorials and specifications available from the World Wide Web Consortium at [17] there is an excellent introductory tutorial on schemas at W3Schools, Introduction to XML Schema [18].