Kobe University Repository : Kernel

PDF issue: 2026-08-11

XML Encoding for Spoken Learner (and Other) Corpora : A Modest Approach

Hardie, Andrew

(Citation)
Learner Corpus Studies in Asia and the World,2:49-62

(Issue Date)
2014-05-31

(Resource Type)
departmental bulletin paper

(Version)
Version of Record

(JaLCDOI)
https://doi.org/10.24546/81006689

(URL)
https://hdl.handle.net/20.500.14094/81006689

KOBE UNIVERSITY

XML Encoding for Spoken Learner (and Other) Corpora

—A Modest Approach—

Andrew HARDIE
Lancaster University

Abstract

Since the earliest days of corpus linguistics, markup has been used to represent features of corpus texts other than the actual words of the text. The first systems used were somewhat *ad hoc*, often based on using (sequences of) punctuation marks to indicate para-linguistic properties of texts, over time the field has standardized on markup systems based on tags delimited by <angle brackets>: first SGML, and more recently the derived XML system. While the official standards for the encoding of corpora with XML – most notably the guidelines of the Text Encoding Initiative – are extremely heavyweight, and therefore most suitable for the development of large-scale reference datasets, I argue that a more modest level of XML can productively be applied within the context of corpora developed by individual researchers or small teams. To apply "Modest XML", it is necessary only to comply with certain fundamental rules of XML (such as the layout of tags for regions and for points within the text, the use of attribute-values to add extra information, the use of a single document-level tag in each XML file, and the use of entities to represent certain special characters). However, it is also useful to be aware of certain *de facto* standard XML tags and attributes very commonly used in spoken corpora (such as <u>, <voc/>, <pause/>, <unclear> and so on). Finally, it is always possible to extend the XML vocabulary that one uses to support the specific needs of a particular corpus development project.

Keywords

Markup, XML, Spoken corpus, Learner corpus, Corpus construction, Corpus encoding

I Introduction

In a recent paper (Hardie 2014) I have argued for a "modest" approach to applying XML in corpus markup. In this paper, I will explore how my proposed "modest" approach can be applied in the specific context of the construction of a spoken corpus,

particularly but not only a corpus of learner language. The paper is structured as follows. In (II) I outline some relevant background on different kinds of corpus markup that have been used in the past, and the emergence of XML as the most widely used system of markup over the past fifteen years or so. In (III) I summarize the argument of Hardie (2014) for a set of guidelines that will allow non-specialists to apply a modest level of XML to the corpora they create. I then move on to consider in (IV) what kinds of considerations would apply to the application of the proposed "Modest XML for Corpora" recommendations in the context of a spoken corpus, looking particularly at the style of Modest XML currently being implemented in an ongoing project at Lancaster University to develop a large corpus of transcribed interactions between learners of English as a foreign language and examiners in an examination context.

II Some background: corpus markup and XML

Markup has been an important component of the corpus development process for as long as corpora in the modern sense – that is, machine readable collections of language data – have existed. The earliest corpora to be published and distributed widely (for example, the Brown Corpus of American English, or the British English equivalent LOB (Lancaster-Oslo/Bergen) Corpus) contained from the outset, not just the words of their component texts, but also representations within the text files of non-linguistic (or para-linguistic) features such as paragraph breaks, sentence breaks, text positions, and so on. Consider example (1), drawn from the original-format version of text A01 in the Lancaster-Oslo/Bergen (LOB) Corpus.

```
A01  23 *<*6AFRICANS DROP RIVALRY TO FIGHT SIR ROY*>
A01  24 *<*4By DENNIS NEWSON*>
A01  25    |^T*0HE two rival African Nationalist Parties of Northern Rhodesia
A01  26 have agreed to get together to face the challenge from Sir Roy
A01  27 Welensky, the Federal Premier.
A01  28    |^Delegates from ¥0Mr. Kenneth Kaunda's United National
A01  29 Independence Party (280,000 members) and ¥0Mr. Harry Nkumbula's
A01  30 African National Congress (400,000) will meet in London today to
A01  31 discuss a common course of action.
A01  32    |^Sir Roy is violently opposed to Africans getting an elected
A01  33 majority in Northern Rhodesia, but the Colonial Secretary, ¥0Mr. Iain
A01  34 Macleod, is insisting on a policy of change.
A01  35    |^Sir Roy's United Federal Party is boycotting the London talks on
A01  36 the Protectorate's future.
```

Example (1). 10 lines of text A01 in the original format of the LOB Corpus.

In example (1), we see a relatively large amount of non-linguistic information is encoded along with the actual words. Every line contains two position labels – one indicating which the text within the corpus, the other a simple sequence number for the lines. (The position labels are not strictly markup in the usual sense, since the text ID is an item of metadata and the line number a matter of corpus storage, rather than being encoded aspects of the original text, which is what the term *markup* usually means; however, this distinction is not important in the present context.) The text itself uses numerous punctuation symbols as markup. Paragraph-initial words are preceded by a vertical bar; sentence-initial words are preceded by a circumflex. Abbreviations are coded by the sequence ¥0.[1] *< and *> indicate beginnings and endings of headings. *6, *4 and *0 indicate typeface shifts (respectively to bold face all-capitals, to bold face, and to non-bold non-italic type) (see Johansson et al. 1978).

This detailed custom markup allowed a very large amount of para-linguistic information to be encoded in LOB – more, in fact, than has proven to be of significant interest for most subsequent research undertaken using LOB; to my knowledge, for instance, almost no research has exploited the typeface codes. Interpreting the markup was the task of customized software running on the mainframe computers of the time. Both software and markup were designed specifically for the needs of the corpus, and thus can be characterized in retrospect as somewhat *ad hoc*. When analytic annotation, in the first instance part-of-speech tagging, was applied to LOB and other early corpora, the manner of its representation in the text files was likewise *ad hoc*. Later, as corpus creation became a more common practice, the field moved towards the creation of standards for markup, annotation, and textual metadata.

At first, the systems established – though not limited to single corpora – were still specific to corpus linguistics. The COCOA reference system for encoding text metadata in a corpus file header (see McEnery and Wilson 2001: 34-35) has, for instance, not seen use outside the field, although it was in widespread use for corpora designed and constructed in the 1980s and may be found in some more recent corpora which retain a conservative design. However, a fully systematised and flexible approach to encoding not only structural markup but also text-level and corpus-level metadata as well as analytic annotations did not arise until the *Standard Generalised Markup Language* (SGML) was adopted for this purpose. SGML is well-known as a markup system based on the use of angle brackets to delimit the tags from the actual corpus text; the earlier COCOA system *also* used angle brackets, but was less systematic and less general than SGML. Critically, SGML itself does not actually define any tags. Rather, it consists of a set of general rules, together with a system (called a *Document Type Definition* or DTD) by which a vocabulary of tags for some particular purpose can be defined. Such a vocabulary is called an *application* of SGML. The single most widely-known application

of SGML for corpus encoding is that of the Text Encoding Initiative (TEI), developed in the late 1980s and early 1990s under the aegis of a number of scholarly associations in the area of digital humanities, but also promoted widely for use in corpus linguistics. The single best-known corpus to use SGML markup according to the TEI standard is the British National Corpus (BNC) (Aston and Burnard 1998).

In the late 1990s, the World Wide Web engineering standards body called the World Wide Web Consortium (W3C) developed a system of markup called the *eXtensible Markup Language* or XML. XML was intended as a successor to SGML, and was motivated by the fact that processing SGML programmatically was rather complex. XML (a) was initially defined to employ only a subset of the markup structures that are legal in SGML but (b) has since been augmented with further technologies allowing XML documents and databases to be manipulated – including XML namespaces, the Xpath addressing system, the XSLT stylesheet language, and the XML Schema language for defining XML applications. Many earlier SGML standards – including the TEI – were redefined to operate as XML applications, and many recent corpora – including the most recent edition of the BNC – have been published using TEI XML.

III The need for a modest approach

In a recent paper (Hardie 2014), I have outlined an argument for a shift in how we approach the use of XML in corpus linguistics. The basic reason for this is that the major standards for using XML, which include the aforementioned TEI as well as a number of systems derived from it (such as the *Corpus Encoding Standard* or CES and TEI-Lite), may fairly be characterized as *heavyweight* in nature. They were designed for what was, in the late 1980s and 1990s, the most typical corpus construction scenario: a major collaborative undertaking aimed at the creation of a multi-user resource. Collecting any sizeable amount of text in electronic form was still at that time fraught with difficulties. This being the case, it was by and large safe to assume that the 'average' corpus would be widely distributed and re-used. Therefore, dense and careful markup of as many potentially useful features as possible was highly desirable – since the requirements of the corpus user could not be precisely predicted. Moreover, given the scale of corpus building projects, it was also safe to assume that any corpus construction team could have access to personnel with a substantial degree of technical knowhow: computer scientists, or at least programmers. Some corpora are still developed within this paradigm, with the aim of constructing large, standardised resources for a wide general audience; such corpora are typically the output of large teams or consortia with substantial technical resources, and in this case the heavyweight standards retain all the many advantages that the TEI presented when the BNC was constructed. However, it would be difficult to maintain that these kinds of

undertakings constitute the 'average' or typical corpus creation undertaking as they did in the early 1990s. A much larger number of corpora are created by individual researchers for their own research interests, and are not expected to be used very widely. In this case, the assumptions that are true of projects like the BNC do not apply. First, there is no especial need for any markup directed at the speculated requirements of end-users, since there is no end-user other than the corpus creator. Second, when a corpus is created by a single linguist, the assumption that a high degree of technical expertise related to computers is available is by no means always met, since expertise in the structure and exploitation of XML document types is a quite distinct skillset to expertise in corpus construction and analysis.

In such a context, the heavyweight nature of standards like the TEI becomes a disadvantage rather than an advantage. By *heavyweight*, I mean (a) that the standards themselves are extremely long and complex documents, which a novice cannot easily acquire an effective working knowledge of; (b) that they dictate an extensive level of markup – even the minimal TEI file header is a very large, complex block of markup, for instance; (c) that there is a substantial technical overhead in the use of these standards – perversely, the use of a full encoding standard with a DTD can make it actually harder to use a corpus, since one slip on the part of the corpus builder can lead to a failure of the corpus files to validate, usually forcing XML-aware software to abort. In light of this, it is unsurprising that in practice single-researcher corpus construction projects have often opted to avoid XML altogether, relying solely on plaintext corpus files with no explicitly encoded internal structure at all below the level of the text. But in many cases this is less than satisfactory; the possibility of marking up structure or analytic annotation into a file is a valuable one. XML is ideal for these purposes, because of its standard nature and also because so much corpus software is (at least partially) XML-aware. It is for this reason that much of the literature on good practice in corpus construction – notably Wynne (2005) – strongly recommends the use of XML.

It is with these dual considerations in mind – that XML is both potentially extremely useful and the standard recommended format for corpora, and that the accepted standards are however heavyweight to the point that in many contexts they do not represent a practicable option for many linguists engaged in corpus construction – that I now generally recommend the use of what I dub a "modest" level of XML markup, the application of which requires a correspondingly "modest" degree of technical knowhow. In the remaining pages of this paper, I will expound in full everything about XML that a researcher would need to know in order to adopt the *Modest XML for Corpora* recommendations, and explain in particular how the creation of spoken learner corpora may be approached from this perspective. The markup I will describe here is being applied within a current project at Lancaster University that aims to construct a large collection of transcribed learner–examiner interactions from recordings of spoken

examinations in English.

IV Modest XML for Corpora, and its application to a spoken learner corpus

4.1 An introduction to the rules of XML

As noted above, like its predecessor SGML, XML is a system of markup where the information that we add to the text is represented by tags surrounded by <angle brackets>. Anything within angle brackets counts as markup; anything outside the angle brackets is part of the actual text. A simple example would be the use of a <u> tag to mark the beginning of an utterance; however, while some de facto standard tags (including <u>) do exist, it is always possible to define whatever tags you need for your own particular purposes. An XML file is a type of *plain text* file. A plain text file is a computer file that consists only of letters, numbers and other characters. There is no formatting information, such as text colour or text font, in a plain text file; however, when we add XML tags to a plain text file we can use those tags to indicate the formatting that was present in the original document from which the plain text file derives. An XML file which follows all the rules of XML is called *well-formed*, which means that computer programs can process its structure correctly. One important rule is that XML files should use proper Unicode encoding (for corpus linguistics, we most often use the form of Unicode called UTF-8). It's traditional to save XML files with the three-letter extension **.xml** (whereas other kinds of plain text file are often saved with the file extension **.txt**).

XML tags consist of a label between two angle brackets – like the <u> tag mentioned above. the labels of tags are case-sensitive. So <u> and <U> would count as two different kinds of tag in XML. Because of this rule, we normally stick to lowercase letters for our tag labels. Tag labels can't be more than one word, and normally shouldn't contain any characters other than the letters of the alphabet. One of the other rules of XML is that all 'whitespace' counts as the same. Whitespace includes line breaks, tabs, and spaces: when a computer reads the XML file, these are all treated as being just the same as a space. Another rule is that putting spaces around the tags themselves is normally optional. So examples (2) and (3) are not significantly different in XML: they are just different ways of indicating the same pair of utterances.

```
<u>Good morning.</u><u>Hi how are you!</u>
```

Example (2). Two utterances in XML, with no whitespace around tags.

```
<u>
        Good morning.
</u>
<u>
        Hi how are you!
</u>
```

Example (3). The content of example (2) with whitespace added for readable layout.

Very often, as in example (2) and (3), we use XML tags to represent *regions* in the text. Each utterance, for instance, is a region that has a beginning point and an end point. When we use XML to mark up regions, the following rules apply:

- the beginning of each region is indicated by a *start-tag* that consists of just the tag label between angle brackets, for instance <u>
- the end of each region is indicated by an *end-tag* that has a forward-slash before the tag label, for instance </u>
- every start-tag *must* have a corresponding end-tag
- regions cannot overlap (that means one region must end before the next region of the same type starts: this rule is sometimes called *perfect nesting*).

We do not always want to mark up regions in our texts. Sometimes, we want to mark up something that occurs *at a particular single point* in the text. In a spoken text, for instance, we might want to indicate that a speaker has coughed or laughed. For phenomena such as this, there is a special type of XML tag, as shown in example (4).

```
<u>so <voc desc="cough"/> three llamas right <voc desc="laugh"/> walk into a bar
<voc desc="laugh"/> and one says </u>
```

Example (4). An utterance containing three vocalizations ("voc") encoded as XML tags.

A final rule is that the entire XML file must be nested within a single pair of start- and end-tags. That is, the very first thing in the file must be a start tag, and the very last thing in the file must be the corresponding end tag. We very often use a <text> tag for this purpose, since most of the time one file corresponds to one text.

Since < and > are used as special symbols to delimit tags, we must represent any actual instances of these symbols in the text by some alternative mechanism. (Obviously, we would expect this to be a concern largely in written rather than spoken texts.) The mechanisms in question are *XML entities*, short codes starting with ampersand and ending with semi-colon that represent the < and > characters. Because ampersand is used for this special purpose, we also need a code to represent ampersand itself. The

codes are:

- Less-than sign (open angle-bracket): < should be represented as *<*
- Greater-than sign (close angle-bracket): > should be represented as *>*
- Ampersand: & should be represented as *&*

Codes also exist for the single and double quotation marks, for use within attribute-values (see below):

- Double quotation mark: " should be represented as *"*
- Single quotation mark (apostrophe): ' should be represented as *'*

4.2 XML attributes and values

Example (4) above illustrates another feature of XML, namely *attributes*. Attributes allow XML tags to contain more information than is held in just the tag label. We use them because quite often, we do not want to only mark the position of a tag in the text: we also want to say something about that tag. The <voc> tags are a good example of this. It is normally not especially useful merely to mark the point in the text at which a vocalisation occurs. We would usually want to say what kind of vocalisation it is. For this, we need the system of markup called *attribute-value* pairs. An *attribute* is a label that specifies what type of information is being recorded; it is then followed by the *value*, which contains the actual information. Attribute-value pairs are encoded inside the actual XML tags, as shown in example (4). The rules for attribute-value pairs in XML are as follows:

- The attribute-value pair is placed inside the tag, after the tag label, with a space in between
- The label of the attribute comes first; the same rules apply to attribute labels that apply to tag labels – they should just be a single word, should normally contain only letters, and are case-sensitive (in example (4), the attribute label is *desc*, short for "description", as it provides a description of the kind of vocalisation that the tag marks up)
- The attribute label is followed by an equals sign; after the equals sign comes the value of the attribute.
- The value of the attribute is surrounded by quotation marks.
- Other attribute-value pairs can then be added inside the same tag.

Typically, we would use the same attributes on every instance of a given tag in the corpus, but the values would (potentially) be different each time. Two commonly-used

attributes which can be used with many tags are *id* and *n*. *n* is short for 'number' and is customarily used to indicate a sequence number. For example, if a spoken text has 100 utterances in it, then the <u> tags can contain the attribute-values *n="1"* to *n="100"*. The *id* attribute is also used to keep track of particular instances of tags by linking each instance to an *identifier* – an arbitrary label that is a 'name' for that particular tag. Identifiers have to be unique – no two tags *in the entire corpus* should ever have the same identifier, even when they are different kinds of tag; this is different from sequence numbers, which *can* be repeated. Moreover, identifiers should usually only consist of letters and numbers – if they contain other characters, this can sometimes create problems when the corpus is processed by some computer programs.

One use of the *id* attribute is specific to, and highly productive within, spoken corpora. This is the coding of speaker identities so that they may be linked to sociolinguistic or educational metadata on the speakers. To explain this usage pattern, let us first consider a use of attributes on <u> tags which is not uncommon among unpracticed corpus builders, and which is not illegal XML, but which represents a substantially suboptimal approach. It is very tempting to use attributes to add speaker information – (anonymized) name, age, sex, social class and so on – to every <u> tag in the corpus, as in example (5).

```
<u who="Mary"  sex="F" age="18">Hello, how are you?</u>
<u who="Timmy" sex="M" age="42">Quite well, thank you.</u>
<u who="Mary"  sex="F" age="18">I'm very glad to hear it!</u>
```

Example (5). Speaker information attributes on <u> tags.

However, this style results in unnecessary repetition: every time Mary speaks, we repeat the information that she is a female aged 18. This is not ideal (imagine, for instance, having to amend an item of speaker metadata due to a mistake: it would affect potentially hundreds or thousands of points in the corpus). The underlying problem here is that things like sex and age are information about the *speaker*, not information about the *utterance*. So, a better way to arrange things is to use a reference to an identifier as the value of *who*, and to store all the speaker information somewhere else. For example, *SP001* could be used as the value of *who* for Mary and *SP002* for Timmy; the actual speaker metadata can then be held in one single locus with the speaker ID providing a link between utterances and metadata, either in an external database or in a corpus header – in the latter case, of course, XML markup can be used to structure the speaker record. The lines in example (5) would then be replaced by the lines in example (6), and the header information would be marked up as in example (7)

```
<u who="SP001">Hello, how are you?</u>
<u who="SP002">Quite well, thank you.</u>
<u who="SP001">I'm very glad to hear it!</u>
```

Example (6). The three utterances in example (5) using identifiers.

```
<speaker id="SP001" name="Mary" age="18" sex="F"
         birthplace="Newcastle-Upon-Tyne" occupation="chef" />
```

Example (7). Header XML coding metadata for one speaker linked in example (6).

The *who* attribute in example (6) is not itself an identifier (the values are not unique – different utterances can have the same value of *who*). But its value *refers to* an identifier, which *is* unique across speakers.

4.3 Tags and attributes for a spoken learner corpus

As noted above, XML as a system does not define any specific tags or attributes: instead, any given *application* of XML defines its own vocabulary of tags and attributes and rules for how they may be combined together. That said, certain XML tags and attributes are used so widely in corpus linguistics that it is almost always the best idea to use those same tags rather than to invent new tags for those purposes. Such tags may be regarded as *de facto* standards. The most prominent of these that is used for spoken corpora is the <u> tag for utterance boundaries, which has been illustrated several times above. It would be perfectly possible – and legal in XML – to use some other tag for utterance boundaries, for example <utt> ... </utt>. However, this would be counterproductive in terms of usability as it would run against established expectations based on how tags are normally used in spoken corpora. In this section I will outline some other *de facto* standard tags for spoken corpora that are in use in Lancaster's current corpus development, as well as our extensions to these *de facto* standards.

As well as the <u> tag and the <voc/> tag, which we have discussed already, widely-used tags in spoken data include <gap/>, <pause/>, <event/>, <unclear>, and <anon/>.

The <gap/> tag is used to make a note of something that has been omitted when the text was transcribed into a corpus file – in a spoken corpus, this might include omitted sections of the original audio recording. <gap/> usually has a *desc* attribute containing a short description of what has been left out.

The <pause/> tag indicates the point in an utterance where a pause occurs. It often has the *dur* attribute to show how long it is (most often measured in seconds), as illustrated in example (8). The <pause/> tag is equivalent to the notation like (2.0) sometimes used for transcribing pauses in traditional transcription of spoken data.

```
<u>I wanted to <pause dur="2"/> er wanted to go home</u>
```

Example (8). The <pause/> tag.

An <event/> tag is similar to a <voc/>, except that rather than a non-linguistic vocalisation it indicates some kind of sound or break on a recording that is not part of an utterance; see example (9).

```
<event desc="prerecorded radio jingle" />
<event desc="noise of passing traffic" />
<event desc="end of side one of tape" />
```

Example (9). The <event/> tag.

The <unclear> tag is very useful for both speech and writing. In a spoken text, it marks a region (a word or words) whose transcription is uncertain – perhaps because it was spoken unclearly, or perhaps because the recording quality was poor. There are two common ways to use <unclear>. If the transcriber has made a "best guess" about what the words are, it is possible to surround the uncertain word or words with <unclear> tags. If no "best guess" was possible, an <unclear/> tag can be used to indicate the unclear words as a point rather than region, optionally with a *dur* attribute to state how long the unclear chunk is. Both styles are illustrated in example (10)

```
<u><unclear dur="3 seconds"/> so I went to um to my <unclear>home</unclear> cos
I wanted to um you know <unclear dur="10 words"/></u>
```

Example (10). The <unclear> tag.

An <anon/> tag can be used as a replacement for any word or phrase that needs to be omitted for reasons of anonymisation; the *type* attribute can be used to indicate why the information has been left out, as in example (11).

```
<u>I want you to assassinate Mr <anon type="name"/> for me uh he lives at <anon
type="address"/> <pause dur="12"/> please hurry this is urgent</u>
```

Example (11). The <anon/> tag.

As well as the tags mentioned above, our current corpus development project makes use of a number of especially defined tags. These tags – including and <timestamp/> – exemplify how the *de facto* standard practice can be extended to cover the needs of a specific project. The project in question is the construction of a spoken learner corpus derived from audio recordings of spoken examinations. Thus, each text represents a single interaction between a student (the person being tested) and an

examiner.

The tag is deployed in our corpus because of the nature of the transcriptions. Each transcription represents a single examination. The examination consists of a predetermined sequence of stages, each of which assesses different skills and each of which constitutes a separate unit of marking for the examination. Since the language used by the learners may differ depending on the contextual factors that vary across the different examination tasks, it is useful if the corpus markup allows us to automatically distinguish language data occurring within different parts of the examination. The tag is used to group together sequences of utterances (<u> tags) and to label each sequence with a reference to the task that is addressed within that sequence. Example (12) shows how this works.

```
<segment name="greet">
        [sequence of utterances in the "greeting" stage of the examination]
</segment>
<segment name="t_disc">
        [sequence of utterances in the "discussion task" stage of the examination]
</segment>
[and so on]
```

Example (12). The use of tags to group sets of utterances.

Finally, and as illustrated in example (13), the <timestamp/> tag creates a link between a point in the transcription and a time-point in the original audio recording from which the transcription was created. This tag's *time* attribute represents the time elapsed since the start of the recording in terms of hours, minutes, seconds and tenths of seconds. The use of these tags – which can occur initially, medially, or finally in an utterance – is a device to enable back-reference from the text to the audio when the corpus data is analyzed; in advanced corpus analysis tools, it is possible to embed links to an audio file in the user interface using the links created by <timestamp/> tags.

```
<u who="TESTID_S">
        it's expensive but now I collect it <timestamp time="0:05:00.4"/>
</u>
```

Example (13). The <timestamp/> tag.

One issue which I have not considered in detail here is the use of XML for corpus text headers. At our current stage of development, the headers we have implemented are very simple, containing only a speaker list; further metadata will be induced and added at a later stage. The header XML is contained within a <header> tag, which sits

alongside a <body> tag that contains the actual text; the document-level tag is <text>, and each <text> consists of a single <header> followed by a single <body>. The overall structure of each text is as illustrated in example (14).

```
<text id="TEXTLABEL">
        <header>
                <speakerList>
                        <speaker id="TEXTLABEL_S" role="student" />
                        <speaker id="TEXTLABEL_E" role="examiner" />
                </speakerList>
        </header>
        <body>
                [actual main text of transcript goes here]
        </body>
</text>
```

Example (14). The overall structure of texts in our corpus of spoken examinations.

V Conclusion

In this paper, I have aimed to accomplish three main goals: first, to explain the key points about the nature of XML (and the historical background to how XML has come to be so widely used for corpus markup; second, to outline a set of generally recommended practices for those XML tags and attributes that are sufficiently widely used that they have attained the status of *de facto* standards; third, to illustrate how additional tags may be deployed for specific purposes within the context of a particular corpus construction project. For this third purpose I used as my example an ongoing corpus-building effort in hand at my own institution, Lancaster University. My hope is that, by outlining this information, I have effectively complemented the more generic argument that I presented in earlier work (Hardie 2014) presenting this "Modest XML" approach to corpus markup; and that other researchers will find this approach a suitable framework around which to build their own practices for approaching, in particular, spoken learner corpora.

Notes

[1] The yen sign that appears here was originally rendered as a backslash in the printer/monitor displays of the European computers on which LOB was compiled.

References

Aston, G., & Burnard, L. (1998). *The BNC handbook: exploring the British National Corpus with SARA*. Edinburgh: Edinburgh University Press.

Hardie, A. (2014). Modest XML for Corpora: Not a standard, but a suggestion. *ICAME Journal*, 38, 73-103.

Johansson, S., Leech, G. & Goodluck, H. (1978). *Manual of Information to Accompany the Lancaster–Oslo/Bergen Corpus of British English, for Use with Digital Computers*. Oslo: Department of English, University of Oslo.
Available online at: http://khnt.hit.uib.no/icame/manuals/lob/index.htm .

McEnery, T. & Wilson, A. (2001). *Corpus Linguistics* (second edition). Edinburgh: Edinburgh University Press.

Wynne, M. (ed.). (2005). *Developing Linguistic Corpora: a Guide to Good Practice*. Oxford: Oxbow Books.
Available online at http://www.ahds.ac.uk/creating/guides/linguistic-corpora/ .