The James Madison Carpenter Collection in EAD

Robert Young Walser
Presented at the
American Folklore Society Annual Meeting
Rochester, New York, October 2002

Encoded Archival Description is a powerful, internationally recognized standard for the electronic storage and retrieval of data about archival collections. It is used by libraries and archives for creating electronic 'finding aids' which describe physical collections in various media. Its remarkable flexibility and adaptability for use either as a means of making existing ('legacy') paper-based finding aids available electronically or, as we have been doing with the Carpenter collection, creating native digital finding aids have brought it quickly to the forefront of archival work in the internet age.

In the next few minutes, Elaine Bradke and I hope to provide the following: first, I will present a brief overview of EAD and XML: the foundations upon which the Carpenter project team has built its work. Next, Elaine will consider some of the ways in which the project team has adapted our work to fit within EAD's framework, that is, to consider some of the challenges and opportunities presented by this archivist's tool when it is employed in the context of a folklore collection. Finally, I will consider how some of the team's objectives which fall outside the boundaries of EAD have been addressed using the flexibility of EAD and certain aspects of XML. In so doing, we hope to highlight some of the conceptual and technological tools we as folklorists and ethnomusicologists can employ in the high-tech context at least some of the Luddites among us can find forbidding.

EAD and/as XML

Introduction

When Julia Bishop first gathered the Carpenter project team in Sheffield just over a year ago, none of us knew much about EAD, XML, or any of the other acronymically labeled technologies we have come to use in our daily work. Though all team members were

used to word processing, e-mail, and some had database experience, only a few knew much about markup languages, programming and so forth. The decision to use EAD and hence XML had been taken. We simply had to cope.

Curiously, part of what we have learned is that many of these high-tech tools are, thankfully, astonishingly practical and indeed simple in their design. They are built on comprehensible building blocks and generally work in intuitive ways. Let me illustrate by discussing first, XML, the system in which our data is organized and stored, and then EAD, the particular set of XML rules chosen for use with the Carpenter project.

Markup Language (like html)

First, what is XML? The letters XML stand for eXtensible Markup Language. Though the inclusion of an X may make this seem mysterious, in fact it is a simple and elegant system for working with many kinds of data. XML is subset of SGML (Standard Generalized Markup Language) (SGML, ISO 8879:1986) which was developed in the 60s and 70s and adopted as an ISO standard in 1986. XML is a simpler derivative which only became official per W3C in 1998.

Plain text – platform independent (but…)

First of all, XML is a markup language. Think of marking student papers or preparing a manuscript for publication. As the teacher or editor marks each page, some marks tell about how things should appear (capital letters, paragraph marks, spelling etc) and other marks relate to the content: questions or comments. XML can work in just this way: text (or images or other data) is surrounded by tags that either tell what sort of data it is or how it should be displayed. A tag called <title> suggests what it contains, another called <emph> suggests emphasized (usually italic) display. It is the first of these that makes XML a quantum leap beyond the familiar HTML which provides primarily for control of display.

Another advantage of XML is that the data is usually stored as plain text – just like simple e-mail – and thus can be used on many kinds of computers. In our case most of the team uses Windows-based machines, but one team member uses a Mac. We regularly exchange files and those prepared on one machine can be used on different

platforms. I should add that this is an imperfect world and that though what I've said is basically true, the movement of data across platforms is not always without hitches!

Nesting

A basic requirement of XML is that all tags must be properly nested. This means that the document as a whole must be contained in a single set of tags and within these any set of tags must be contained within another. Thus particular configurations of data are allowed and others not as shown.

Allowed:

```
<Document>
        <SomeData>Blah blah blah</SomeData>
        <SomeData>Blah blah blah</SomeData>
        <SomeData>Blah blah blah</SomeData>
</Document>
```

Not Allowed:

```
<Document>
        <SomeData>Blah blah blah</SomeData>
        <SomeData>Blah blah blah</SomeData>
</Document>
        <SomeData>Blah blah blah</SomeData>
```

Not Allowed:

```
<Document>
        <SomeData>Blah Blah
                <MoreData>XOXOXO
        </SomeData>
                </MoreData>
</Document>
```

This hierarchy, basic to XML is useful in the context of archives as it can nicely reflect the organization – either physical or intellectual – of the items it contains. For example, papers are often contained in folders which in turn are held in boxes, a system easily represented in an XML hierarchy.

Intelligible tag names

Since tag names are not specified by xml, users are free to invent their own as long as they follow a few rules which we needn't worry about today. The effect of this is that someone looking at an xml document, even if they are not familiar with its content, can make an intelligent guess as to just what it is. For example, consider this item from our collection, shown here in raw XML. Even if you don't know the meaning of every tag, you can still understand a lot about what is being represented here.

```
<c05 otherlevel="init" id="p03249.0">
        <did>
                <unittitle>
                        <title>Blow the Man Down</title>
                        <genreform>song text</genreform>
                        <unitdate normal="1928.00.00">1928</unitdate>
                        <persname role="contributor" normal="O'Connors, Dennis"
                        source="local" authfilenumber="OConnorsD">Dennis O'Connors
                        </persname>
                        <geogname>Sailors Snug Harbor, Staten Island,
                                N.Y.</geogname>
                </unittitle>
                <container type="page">
                        <extref href="ms027/0130">03249</extref>
                </container>
        </did>
        <admininfo audience="internal">
                <note>
                        <p>Item entered by RYW, 2002.01.30</p>
                        <p>Item updated by RYW, 2002.07.19</p>
                </note>
        </admininfo>
        <scopecontent>
                <p><genreform>song - shanty - Child Ballad</genreform></p>
                <p><title role="first line">As I was a walking one morning in
                spring</title></p>
                <p>Nearly identical to p.<extref href="ms027/0131">03250</extref></p>
                <p> 24 lines</p>
        </scopecontent>
        <controlaccess>
                <title normal="Farmer's Curst Wife, The">The Farmer's Curst Wife
                        <num type="Child">286</num>
                </title>
        </controlaccess>
 </c05>
```

eXtensible

The X in XML refers to the fact that this language is ‘eXtensible’, that is, there are no pre-existing limits to the number and names of tags, or their arrangement, contents etc. But the system is not without control: though not required by XML there is provision for control provided by a ‘dtd’ or document-type-definition.

Control Provided by dtd

EAD is implemented through a dtd developed and maintained by the Society of American Archivists, though with broad international involvement. This dtd, like all dtds is simply a set of rules for a set of tags and their use. EAD (version 1.0) specifies 145 tags and which tags can contain text and/or other specified tags. It further specifies which attributes apply to which tags – as seen in the example earlier and as Elaine will explain in greater detail.

Controls Validity of Document not Data

When an xml file is loaded by software that is xml-aware it typically validates the xml file, that is, it checks the file against the dtd to see that it conforms to the rules set out in the dtd. It is important to note here that this process checks the arrangement of tags and attributes in the file but does not check the validity of the data contained within the tags. There is an xml technology that provides for some data validation, but EAD does not use that technology.

Rendering the Results – XSL

It is lovely to have the data arranged in tags with meaningful names, but you and I probably don't want to spend much time looking at raw XML. Fotunately a parallel standard called XML – eXtensible Stylesheet Language facilitates display of a data in various ways through the use of Stylesheets. For example, using different stylesheets the small chunk of XML we looked at earlier can be displayed in various ways using some or all of the tagged information. **Demo: 1) Search on Blow the Man Down 2) find 3249 on display 3) notice compact unittitle 4) Use link 5) page search on 03249 6) demo links.**

Summary of Part I

I have tried to intro EAD and XML now transition to Elaine.

Carpenter 'within' EAD (Elaine Bradtke)

The 'Intellectual item'

Identifying 'intellectual items – the complex unittitle

Carpenter 'without' EAD – using the X in EAD

Now that we've seen how the Carpenter team's efforts have been adapted to the world of EAD. I'd like to consider with you two areas where the team has used some technological tools to enhance our work. I want to consider these under the headings of 'Coloring outside the Lines' and 'Dancing Greek to Geek'.

Dancing Greek to Geek – XmetaL macros and the XML-folklore interface.

It is lovely that xml files are plain text, and that the intelligible tags render these text files understandable to the uninitiated. But can you imagine entering thousands of data records in the plain text form we've seen? The opportunities for operator error boggle the mind!

Fortunately, xml is a hot commodity at present so there are various xml editing software packages available. XMetaL was suggested by Mike Heaney of the Bodlian as a good tool for our use and we took his suggestion. When we first loaded it, here's the screen we saw (DEMO) - Needless to say, when the team members first loaded it we stared at the screen with looks of confusion and despair. Learning new technology and new software, particularly for those whose interests are in the content of the Collection rather than technology, can be daunting.

To address this, the team created a series of macros which enabled fill-in-the blank creation of the data records. This enabled the team members with XMetaL to get on with the job of cataloguing and not have to worry so much about studying xml and the software documentation (which is usually not intelligible to ordinary humans anyway!).

(Demo entry creation)

The point here is not the detail of how we did it, though we are happy to share the details. The key is rather that we needed to move quickly from our specialist knowledge

of mummers plays, ballads and the like (what I'm calling Greek) to the technical context of xml (or indeed another technology – what I'm calling Geek).

Added-value documents

As we have seen, EAD – with a little massaging – is a powerful tool for the Carpenter team's work. However, it does not provide all the functions that we need for comprehending Carpenter's collection. This is where we have begun to explore the X in XML, moving beyond EAD but consistent both with it and the rules of XML. Our question was this: How can we harvest and control data not supported by the EAD tagset? In particular, we wanted to control information about people and places. In the case of the former we had several goals: Gathering the scattered information present in the collection into a compact and accessible form, secondly providing control for Carpenters sometimes erratic spelling to facilitate finding all the items contributed by a single individual even though the name may be spelt in various ways in various places. A further aim is to provide a place for collecting data about individual's descendants for eventual use in clearing intellectual property issues related to 'publishing' the collection in electronic or print form. Thus we wanted to create in part a name authority file – though likely one with little overlap with existing name authorities such as the Library of Congress', and in part a data store for information both for public and internal use. EAD's persname tag alone does not provide these functions.

The issues with placenames are different but related. Again, Carpenter's inconsistent spelling creates challenges in gathering items collected in a particular place. Further, Carpenter's papers present an enormous variety of levels of detail associated with particular items. Sometimes there is a full listing of address, town, county and country, but more often only some of these levels are included. A further level of complication has to do with the shifting boundaries and names of English counties particularly since 1972. Thus the team is here concerned to again provide a geographic name authority and to provide a means for connecting related items and further, through the use of Ordnance Survey references to enable users to locate particular places even though names and boundaries may have changed since Carpenter's sojourn in England.

Having learnt the basics of xml it seemed natural to us to consider using the same technology to address the team's needs beyond the scope of EAD. To do this we considered first what sorts of data we wished to control, next how that data should be tagged and finally how it could be related to the EAD document. Among the team's concerns is that our additional or 'added-value' efforts must in no way compromise the integrity of our EAD document. You will recall that EAD is an inter-institutional standard used by libraries and archives all over the world. In this interconnected context we would like our data be comprehensible by other EAD users. The potential then exists for our work to be accessed through archival meta searches such as the UK archives-hub.

To respond to these challenges we have chosen to create additional xml documents external to the EAD document, but designed for linking among this set of related documents. Initially we considered another inter-institutional standard, EAC or Encoded Archival Context, but after examining the draft specification discovered first that the standard is not yet ready for use, second that it is far more complex than our simple needs require and, third that it would only address our needs for information about people and not places.

Carpenter Project Geographic Names Document

```
<places>
        <country>
                <state>
                <county>
                        <p>
                        <locality>
                                <p>
                                <locality>….
                                        <p>
                                        <address>
                                                <p>
```

We thus decided to create our own dtds to control our own, project-specific external documents. This is not, in fact, the daunting task it seems. Dtds are themselves xml documents and can be created and edited in any plain-text editor. Like other aspects of XML, the underlying concepts are relatively simple and few and we were able to create our dtds in quite short order. I will discuss them briefly in greater detail in a moment, but first want to address the problem of linking from the EAD document to and from these external documents.

The Society of American Archivists, EAD's sponsor, is, of course, well aware of the need for and use of authority files. Hence both the <persname> and <geogname> tag include an 'authfilenumber' attribute. This can contain any plain text – again, the dtd controls the validity of the structure, not the content itself. According to the EAD tag library, the AUTHFILENUMBER is 'a number that identifies the authority file record for an access term drawn from that authority file. If this attribute is used, the SOURCE attribute should also be used to identify the authority file.'

Carpenter Project Personal Names Document

```
<people>
        <person authfilenumber="OConnorsD">
                <nameproper>The most common form of the name
                <DOB>
                <DOD>
                <address>
                <descendant audience="internal">
                <p> narrative data of any sort</p>
        </person>
</people>
```

Linking is through the <person> tag's authfilenumber attribute.
All tags within the <person> tag require a source attribute.

With this in mind we have created our own DTD with the specific needs and limitations of Carpenter's data in mind.

Issues:

Crosswalks – inter-archive searching

The Carpenter project team has found that EAD and XML provide a powerful environment for the work we have to do. In a larger context we can hope that ours is work upon which others can build. Our use of EAD is the first step towards making our work useful to a larger universe of users. Looking to the future we can wonder if our data model or template could be used by other collections of folksong and folk drama, or for collections of other folk genres. The creators of EAD had the foresight to include facilities for fledging technologies such as XPointer and XLink. Dare we hope that these tools can be used for linking encoded folklore catalogs across this country and abroad?

Might we dream of a folklore meta-site along the lines of the archives-hub in the UK which searches thousands of encoded finding aids across hundreds of repositories? The EAD *Best Practice Guidelines* recently puyblished by the Reasearch Libraries Group (2002) are helpful in this direction – providing models for standardizing the use of EADs tags and uniform presentation of the data within them.

Conclusion

As the Cataloging phase of the Carpenter project nears completion we can see that the decision to use EAD and XML has had fascinating if unexpected consequences. We can hope that the lessons we have learnt from bridging the gap between Greek and Geek, and ways we have embraced the extensibility ot these new technologies will offer others useful concepts they can employ towards solving their own research problems.