Publishing, XML and indexers

Nic Gibson

Things have changed in publishing, both within publishing houses and in their relationships with suppliers and freelances. Indexers are aware of the steady downward pressure on costs and subsequent outsourcing to lower-cost environments. Many indexers will also have seen publishers turn to automated indexing software. These are all clear risks to the indexing profession. However, the author believes that the introduction of XML-based workflows and the growth of the ebook market both provide new opportunities for indexers. This article discusses some of the risks and problems inherent in these changes with reference to indexes and indexers. It also discusses the opportunities for indexers to engage with and benefit from the world of digital publishing.

A very brief introduction to XML

The eXtensible Mark-up Language (XML) was published by the World Wide Web Consortium (W3C) in 1998. XML is a meta-language; it describes a set of rules that can be used to write mark-up languages. Writing mark-up languages with a shared set of grammatical rules means that it is possible to create software that can process XML without knowing anything specific about the mark-up language content or structure.

XML is a simplification of the Standard Generalized Mark-up Language (SGML) which derives from work by IBM in the late 1960s. SGML became an international standard in the early 1970s and was taken up by organizations with large and complex document requirements (the aircraft and software industries and legal publishing, for example).

The following is an incomplete but sufficient introduction to XML.

An XML document consists of a nested set of elements identified by angle brackets. Each element consists of an opening tag, some content and a closing tag. For example:

```
<document><sentence>Hello World
</sentence></document>
```

There are two elements in the example above – document and sentence. The opening tag for an element consists of the name of the element contained between ‘<‘ and ‘>‘. The closing tag is similar except that it starts with ‘</’ instead.

The opening tag of an element may contain additional information known as attributes. Attributes are name and value pairs that provide additional information. For example:

```
<document version=“1.0″ language=“en″>…
</document>
```

This example has two attributes – version and language with values of ‘1.0’ and ‘en’ respectively.

In XML terms, content is anything that exists between the end of the opening tag and the start of the closing tag of an element. It can consist of elements or text (and occasionally a few other things).

XML has very strict rules about the relationship between the elements. Elements must be closed in the opposite order to opening and they must be strictly nested (as in the first example). Any document that follows the basic syntax rules of XML is known as a well-formed document.

The exact relationship between elements, attributes and text is defined by the language designer, and is made concrete in a document known as a schema (or a document type definition, DTD).

Most of the XML examples below use an XML language known as DocBook (see `www.docbook.org/` for information). I have chosen this language because it has a set of elements for use in indexes. You can also see Michele Combs’ article on page 47 of this issue for more about XML.

Change in publishing

Publishing is in the middle of a period of dramatic change. Since around 2006, the long-promised ebook market has finally become viable. Publishers are looking to change their processes to ensure that they can meet the challenges of a changing market. Digital-only titles are now a reality, and publishing workflows are changing for the first time in a hundred or more years. Ebook publishing causes specific problems for publishers and indexers. There is, as yet, no good way to present an index in an ebook.

Many publishers are trying to achieve publishing efficiency through what is known as ‘XML first’ publishing. In an XML-first workflow, the manuscript is converted to XML as early in the process as possible. Generally, this occurs just after an initial copy edit (to tidy up the manuscript and ensure that it will convert to XML successfully). In an XML-first workflow, the XML file is the source and object of all work processes. In the more sophisticated implementations of this sort of workflow, typesetting is an entirely automated process consisting of a software-driven transformation of the XML into print-ready PDFs, HTML web pages and ebooks.

One of the implications of this type of workflow is that pagination is not set in stone at any point in the process. Clearly, that has implications for indexing. Ideally, the index should be created from and in the XML ‘manuscript’ itself, not after the PDF has been generated.

There is no reason that an XML file must be indexed directly (where the index terms are set in the XML ‘manuscript' file as XML, not as a separate list of words). However, there are many benefits to doing it this way. By indexing XML directly, we remove the relationship between the index and the page. Having removed that relationship, we can use XML tools to create the appropriate index for the output medium. There are software applications that can take a piece of XML with index terms inserted into it and convert it to the appropriate output format.

There is no absolute reason for requiring that an index must be present in the XML manuscript itself. In fact, most publishers do not want the indexer to modify the manuscript file. I believe that a compromise approach is possible. The two examples provided as Figure 1 and Figure 2 demonstrate a technique for the creation of native XML indexes with only minor modifications to the source XML file. These examples are considerable simplifications but we hope they show the concepts well.

Figure 1 places the index terms directly into the text at the point of reference. Figure 2 shows a marker in the XML file (the anchor element) and the index references that anchor from the index term (using the ref attribute). That second example is closer to the XML indexing approach currently used by some publishers. (It is possible to create the first from the second automatically.)

The important concept is that both of these examples create an index based on the content itself, not the printed page. Processing this XML file (or files) allows publishers to create multiple formats from a single source using a single file. Additionally, translation issues may be eased as the index and the text can be translated separately without any need to update the location markers. In the United States, the technology publisher O'Reilly is using this approach on almost all its titles, giving it time-to-market and cost benefits. As soon as the printed page becomes irrelevant to the index, the opportunities for using that index grow.

Example: embedded index terms

```
<para xml:id=“rem36”>That romance in
turn allows the metaphor to spread into
other social or political conflicts.
We wage war on drugs, on poverty, on
terrorism, on racism. There is a war
on government waste, a war on crime, a
war on spam, a war on guns, and a war
on cancer. As Professors <indexterm
class=“singular” xml:id=“rem37”><primary
xml:id=“rem38”>Lakoff, George</primary>
</indexterm>George Lakoff and <indexterm
class=“singular” xml:id=“rem39”><primary
xml:id=“ rem40”>Johnson, Mark</primary>
</indexterm>Mark Johnson describe, each
of these “wars” produces a “network of
entailments.” Those entailments then
frame and drive social policy. As they
put it, in discussing President <indexterm
class=“singular” xml:id=“rem41”><primary
xml:id=“rem42”>Carter Jimmy</primary>
</indexterm>Carter's “moral equivalent
of war” speech:</para>
```

Figure 1 Embedding an index within the XML file

Example: referenced index terms

```
<para xml:id=“rem36”>That romance in turn
allows the metaphor to spread into other
social or political conflicts. We wage war
on drugs, on poverty, on terrorism, on
racism. There is a war on government waste,
a war on crime, a war on spam, a war on
guns, and a war on cancer. As Professors
<anchor xml:id=“rem37”/>George Lakoff
and <anchor xml:id=“rem40”/>Mark Johnson
describe, each of these “wars” produces
a “network of entailments.” Those entail-
ments then frame and drive social policy.
As they put it, in discussing President
<anchor xml:id=“rem38”/>Carter's “moral
equivalent of war” speech:</para>
```

Elsewhere (possibly in another file):

```
<itermset>
<indexterm class=“singular”
xml:id=“rem98” target=“rem36”><primary
xml:id=“rem99”>Lakoff, George
</primary></indexterm>
<indexterm class=“singular”
xml:id=“rem100” target=“rem37><primary
xml:id=“ rem101”>Johnson, Mark
</primary></indexterm>
<indexterm class=“singular”
xml:id=“rem102” target=“rem38”><primary
xml:id=“rem103”>Carter Jimmy</primary>
</indexterm>
</itermset>
```

Figure 2 Referencing a separate index file. This is not a DocBook file – it simply shows the concept.

While there is no guarantee that XML is the future of publishing, it is playing a more and more important role. We believe that there is a great potential benefit to indexers in learning XML. High-quality indexing is much more important in digital publishing than many publishers believe. Search can be massively improved simply by using the index as the source rather than the text of a title.

It is perfectly possible to create the index for an ebook by creating the ebook from the XML file (and this generally creates a better ebook than the conventional practice of creating it from a PDF). Currently, there appears to be no good way to represent an index in an ebook, and for this reason many publishers do not include indexes. The ASI Digital Trends Task Force (DTTF) team has joined with the International Digital Publishing Forum (IDPF) to work on indexing standards for epub files. We think that indexers might be in a strong position to propose a solution to this problem by thinking creatively about the appearance of an index in text that has no pages (and, importantly, no page numbering).

The way forward for indexers

If indexers can offer XML-based services, publishers will use them. Learning enough XML to create an embedded XML index will be a prerequisite for joining the digital publishing revolution.

Learning XML is not hard for anyone who can think rigorously. Indexers are good at structure, and fundamentally, XML in publishing is about the imposition of structure on text. Publishers often believe that general text search will overcome the need for an index in digital publishing. This is not the case – and indexers should try to educate publishers to make sure they understand this. In the near future, many books will be published that will never appear in print. Many of those books will need to be indexed.

Nic Gibson is half of Corbas Consulting, which provides digital publishing consultancy and training to publishers. He has worked in and around the media industry since the early 1990s. He started his career in multimedia, doing hypermedia development and progressing into web development and XML. For the past ten years, he has worked on digital publishing projects including workflow transformation projects, eBook development and XML-driven publishing for a variety of trade and academic publishers in the United Kingdom and Europe. Email: nicg@corbas.co.uk

indexing specialists

Our team of experts have many years' experience in undertaking and designing effective indexes

● *Books, journals, manuals* ●
● *Multi-volume works* ●
● *From 100,000 to 100 lines* ●
● *Unfamiliar subjects* ●
● *Design of style and layout* ●
● *Short deadlines met* ●

Phone +44 (0) 1273 416777
email indexers@indexing.co.uk
website www.indexing.co.uk

INDEXING SPECIALISTS (UK) LTD
306A Portland Road, Hove
East Sussex, BN3 5LP, UK

Editorial continued from page 1

The reviews section, not by accident, is oriented towards the indexer's digital world. And lastly, Pierke Bosschieter has pulled together have a lovely section filled with important links and working groups, to help expand your knowledge in all of these areas.

Our biggest challenge is to keep track of book publishing innovations, and quickly develop ways to incorporate indexes as important navigation and metadata in each innovation we see. We add value and metadata. The index can come along for the ride into every interface as long as we understand the technology used for presentation. We can let go of the page, and think of locators as just that: locators. We do not care if they are anchor points in HTML, unique ID strings in an InDesign file, or time codes in a video. The human analysis of the aboutness at that location is our unique contribution. The technology for presenting that location and accessing its content can change, but we are still the best resource for letting readers know whether a location is worth the visit, and where else they can look.

I urge you to find out for yourself how the current crop of ereaders display indexes. Find, buy, or borrow an ereader, whether it is a Kindle, Nook, Kobo, or iPad. Look at book indexes on the readers, such as Steve Jobs's biography, the SI publication *Indexing children's books*, the ASI publication *Marketing your indexing services*, Browne and Jermey's *Indexing companion* at http://amzn.to/ynrn8Y, or the OASIS Open Office Specification at http://bit.ly/wcjsLg. Each of these chapter-like indexes was developed in a different way, and each has differing success with access: links to the page, or to the paragraph. How should we use locators? What is the simplest path for the reader? Examine the interface your reader will be using to look at your index, and think about how to make it work better. Read the resource documents at http://bit.ly/uqKwD7. We illustrate ways forward in these documents, and indexers can use them in their discussions with clients.

Let's all take a look at what we want indexes to be, and throw out ideas about how to get there. It is still early in the game.

[*See too the note from Maureen MacGlashan on page 56.*]

Jan Wright, guest editor