Indexers and XML: an overview of the opportunities

Bill Kasdorf

The profession of indexing and the technology of XML have only begun to realize their value to each other. First, it is important to distinguish two quite different types of indexing, closed-system indexing (such as traditional back-of-the-book indexing) and open-system indexing (generally used to classify documents in a collection), and to distinguish these from the indexing behind the 'search' function we take for granted on computers and the Web. Through its ability to delineate the structure of information, to navigate that structured content, and to associate metadata with it, XML makes it possible to dramatically improve all three types of indexing and to enable them to work together in powerful and useful ways. Beyond simple 'extracted' or 'embedded' indexes, 'integrated indexes' use XML to associate indexes produced with professional standalone indexing software and the content being indexed at a very granular level.

Indexers and XML were made for each other. This alignment of talent and technology is poised to revolutionize how we interact with information.

This revolution has only just begun. Despite the fact that XML is now a well-established, stable technology that is integral to many or most systems used today to process and exchange information, and despite the fact that indexing is a highly evolved profession based, in its modern form, on more than a century of accumulated wisdom and refinement, XML and indexers have barely started to realize how much they can benefit from each other.

Part of the problem is that indexing, more than most other parts of the publishing process after writing, is dependent almost exclusively on human intelligence. Although rudimentary forms of indexing can be automated, real indexing of the sort that is of real value to users (and I deliberately use the word 'value,' not simply 'utility') requires the judgment and sense of proportion, the sensitivity to nuance and implication, and the intellect and acuity to perceive subtle distinctions that only an intelligent and educated human being can provide. While they readily use computers to *help* them do their work, most professional indexers are skeptical of technology that promises to actually *do* their work.

Another part of the problem is the overly broad – in fact, sloppy to the point of meaningless – use of the term 'indexing.' We use the same word for the process a human being uses to compile a brilliant and sophisticated back-of-the-book index as for what Adobe Acrobat does when it compiles a list of the words in a set of PDF files. And we use that same term for assigning subject classifications to the documents in a collection. Some of these kinds of 'indexing' can indeed be automated; but the fact that some of them clearly cannot distracts us from the extent to which technology can help a professional indexer and improve the resulting index.

The obstructing paradigms are not all on the indexers' side of the equation. Anyone who has worked with programmers realizes how breathtakingly optimistic they can be. They live to solve problems and they relish coming up with elegant solutions. These surprisingly naïve Pygmalions fall in love with their beautiful creations. Not only are they blind to their imperfections (armies of testers are required to find the flaws, including the users of most versions 1.0), they think their lovely Galateas can do anything. What could be so hard about creating an index? Surely a computer should be able to do that better and faster than some pathetic human being sitting in her kitchen with a book, a pot of coffee, and a baby about to wake up.

That indexer may or may not be in her kitchen, but wherever she is, it is more than likely that she is sitting in front of a computer today. Much of the technology on which that computer is based, and much of the information accessible to that computer, uses XML, at least under the hood. The integration of that indexer's knowledge and intelligence and that technology's flexibility and power is about to start making some serious progress.

Open-system versus closed-system indexes versus word lists

Let's first be a bit clearer about what we mean by 'indexing.' The most important distinction is the one Susan Klement so ably articulated in her article 'Open-system versus closed-system indexing: A vital distinction' (Klement, 2002).

Closed-system indexing is the process of identifying and locating the concepts, and the relationships between them, within a document or a specific collection of documents. It is what we think of as 'back-of-the-book indexing.' Such an index contains terms that may or may not occur in the actual text being indexed, and it attempts to point to locations anywhere within a document (although print-based technologies have led us to accept the page as an acceptable level of granularity for this purpose). A good closed-system index is comprehensive, including all the concepts an expected user of the indexed document or set of documents might wish to find; and it is ideally rich, complex, highly structured, and well proportioned. It is specific to a unique document or set

(generally not being applicable to any other documents). It is usually created by one individual, ideally a professional indexer using professional indexing tools.

Open-system indexing is the process of assigning terms that describe what each of a set of documents being indexed is about. Often called 'periodical indexing' or 'serials indexing,' it is most often applied to large, open-ended collections of documents that continue to grow over time. Its purpose is to direct readers to documents (whole documents, not generally to portions of them) that are about certain topics. Unlike the unique set of terms that a back-of-the-book indexer creates, an open-system index benefits from a consistent, and often controlled, vocabulary. Such indexes are often created quite apart from the creation of the indexed documents; in fact, they are often separate products created and sold by parties unrelated to the publishers of the documents being indexed. They are rarely created by individuals; because these indexes often cover extremely large collections of documents that accumulate for many years, they are usually created not even by a coordinated team but by a simple succession of indexers over time. Their vocabularies need to be monitored and modified: the term 'colored people' would no longer be an acceptable term, and in any case would mean something slightly different to readers in the United States and the UK, not to mention readers elsewhere in the world who can now easily access these indexes over the Web. Far from being comprehensive, such indexes focus on a few central topics of the documents in question, not every topic that might be mentioned within them. These indexes are typically flat and relatively unstructured, often no more than short lists of 'keywords'; complexity generally makes them less, not more, usable. They are often created by librarians or cataloguers who do not think of themselves as 'indexers.'

'Indexing' is also used to denote a third process. That is the process by which a computer program compiles a list of all the words in a document or a set of documents and records their locations to expedite finding them. The easiest way to appreciate this is to compare the functions of 'find' and 'search' in Adobe Acrobat. The 'find' function reads consecutively through a single document to locate all occurrences of the specified 'search string' and highlight them. The 'search' function appears to do basically the same thing (though it contains seldom-used advanced features), but it can search through a collection of documents, not just a single one, and it does this dramatically faster than 'find' because it is spared the task of plowing through all the text consecutively. Instead, all the words in the given set of documents have been 'indexed': the program has kept a record of every place every word occurs, so it simply zips right to the specified words in the list and highlights their locations. This is very useful, but it is not what you and I generally mean when we talk about 'indexing.' A similar process is used for 'searching' on the Web, although the leading search engines have refined the process to an amazing degree. We all love (or at least use) Google; but surely many of us cringe when what it is doing is said to be 'indexing.' Professional indexers must relish the invertebrate connotation of the term 'web crawler.'

It is this latter form of 'indexing' that makes that indexer sitting at her computer think of looking for another occupation. I beg her not to give up. She is the hub and the heart of what is needed to make these three kinds of indexing really work.

Searching, classifying, indexing – with XML

Just as the true value provided by a professional indexer is often underappreciated, so is the flexibility and power of XML. We often think of XML in terms of the structural tagging needed to define a document – heads, subheads, lists, and so on – and this is, indeed, an important dimension of what XML does. Unlike HTML, which is a prescribed set of tags, XML enables us to create whatever tags we want, and to provide a way for unknown and unrelated processing systems (such as formatting systems) to use them. For example, HTML includes the tag to delimit the beginning of a paragraph and the tag to delimit its end. XML enables us to create the tag <abstract> to delimit the beginning of a document's abstract, and </abstract> to delimit the abstract's end. The tags and pertain to the structure of a document; the tags <abstract> and </abstract> pertain to its semantics. Within the syntax specified by the XML standard – the very rules that make XML so flexible and interoperable – we can embed virtually any kind of information in an XML document. We can delineate that document's structure down to extremely granular levels (publications within collections, documents within publications, sections and subsections within documents, paragraphs within sections, and elements within paragraphs, down to single characters if we so choose), and we can embed information that may not actually appear in the published text at all, or that may or may not relate to the document's structure.

Why should this matter to indexing? Because it provides a way to transcend the limitations of virtually all previous publishing technology: it enables us to say precisely what we want to say about the content, and to say it in the way that makes the most sense.

Today's web searching is largely document based: by crawling the Web (at least, the public portions of the Web, and whatever private portions they have negotiated access to), the search engines identify what they consider to be relevant documents and return their URLs to the users. Their current ranking methodology is also document-centric: the most popular of them judge the relevance of a document by the number of other places on the Web that point to it, not unlike citation indexing for scientific literature. But the piece of content most relevant to a given seeker might be a small section of a document currently hidden behind that document's URL. We are left, figuratively speaking, to pulling that document off the shelf and flipping through it to find what we are looking for.

XML – especially with its related technologies XLink, XPath, and XPointer – provides a way to open the cover of that document and electronically find locations within it through its structural or other tagging, or even through the relationship of the content to the tagging. We have become accustomed to the simple links characteristic of the Web in its current HTML-based implementation; but as XML becomes more pervasive, more sophisticated linking technologies will become more prevalent. They enable links to

be bidirectional (pointing from either end to the other) and multi-ended (pointing to more than one location), and they enable links to point to places in the text that lack an explicit target tag, relying instead on the structural information – and even the content contained within those structural tags. They even enable the creation of 'linkbases,' collections of linking information that can be maintained apart from and independently of the documents that are being linked. What is indexing, after all, but pointing to and linking information? This will be an environment ripe for exploitation by indexers – and desperately in need of indexers. Imagine the unworkable clog of complexity that this technology will be able to produce! What is needed are the principles of distinction, proportion, and selection that are the indexer's stock in trade.

In addition, XML gives us dramatically expanded abilities to describe and classify information, and to make those descriptions and classifications more accessible and useful to those seeking information. This is the realm of the 'unseen' information in XML documents: metadata. Metadata – or information about the information – can be embedded in XML documents in any number of ways. Journal articles, for example, now begin with lengthy 'headers' that provide bibliographic information in the form of metadata about the article; these headers are the basis for registering the articles in such services as PubMed and CrossRef and for assigning them DOIs (Digital Object Identifiers, which are persistent names that uniquely identify them within a networked environment). Metadata is also critical to such important standards as the OpenURL (which provides a mechanism for associating metadata with a URL that helps identify the content at that location) and the OAI-PMH (the Open Archives Initiative Protocol for Metadata Harvesting), which provides a standard by which metadata about content can be made freely available on the Web, whether or not the content it describes is free. Metadata can be incorporated anywhere within an XML document – either as specific elements embedded in the content, reflecting hierarchical relationships as appropriate (e.g., `<subj1>History<subj2>British <subj3>Edwardian era</subj3></subj2></subj1>`) or as attributes attached to elements (e.g., `<div id="12345" subject="Impressionism">`, in which 'id' and 'subject' are attributes of the element `<div>`). Alternatively metadata can be maintained in a document or database separate from the related XML documents and associated with them through a technique known as 'metadata binding.' The latter is an especially powerful and flexible technique: by assigning unique IDs at a very granular level in the content, an increasingly rich set of metadata can be associated with those pieces of content without requiring the XML documents themselves to be updated.

Despite my efforts above to distinguish searching, classifying, and indexing as three quite separate processes and mechanisms, it should be apparent now that XML makes it possible to combine them in very dynamic ways. Imagine how much more effective searching will be when it can be focused on appropriate documents or on appropriate portions of documents. (XML gives us ways to help future search technologies know where to look.) Imagine how much more valuable classification will be when it is applied not just to whole documents but to the relevant portions of documents. (XML enables us to associate such metadata with structural and semantic features of documents.) Imagine how much more useful a 'back-of-the-book' index will be when its entries point to the precise portions of text to which they refer. Suddenly, in the digital era, the page (while still relevant) has become an unfortunately coarse, imprecise way to specify a location, one which bears little or no relationship to the content itself. A recent study, 'Index versus Full-Text Search: A Usability Study of User Preference and Performance' (Barnum et al., 2004), showed that although users were able to find what they sought more effectively using an index, they preferred using full-text search in an environment where the index was only able to point to a page (in this case, using PDF). The frustration of having to scan over the page to find what they were looking for was just too much for these users, despite the fact that this is precisely what they have always done in print. Time marches on. Pointing to pages is no longer sufficient.

Extracted, embedded, and integrated indexes

Even within the context of XML indexing, there is confusion between three types of indexes, which I call extracted, embedded, and integrated indexes. Just as all three indexing methods – search, classification, and indexing – are useful, each of these three types of indexes has its place, and they can even peacefully coexist. But they are not at all the same things, and they are often confused.

Extracted indexes are indexes created by extracting terms from the text, organizing them, and pointing to their locations. This is the kind of index that is often first envisioned as being easily automated and reasonably sufficient. When all that is needed is a concordance to the actual terms in the content, the process can, in fact, be automated. But many publishers naively assume that it is possible to create a serviceable index for print in this way – not all the words in the text, but just selected, meaningful terms, organized in a useful manner. This can only be done in the most schematic of situations, when what is needed is a specialized index that does, in fact, need to point to specifically identifiable terms in the text – for example, an index of genus and species cited in a book of natural history, or an index of artists' names in an art history reference. The former is semi-automatable in conventional digital files (presuming that genus and species names are italicized and not much else is); the latter is hopeless unless some added tagging and processing are done. XML could be used to make both tasks more feasible by tagging genus and species and artists' names as such. But the trouble does not end there, because there are often too many inconsistencies or abbreviations in the content, not to mention ambiguities that must be resolved by someone with subject knowledge (e.g., that 'Leonardo' and 'Da Vinci' are the same artist). The presence of special characters such as accented letters, which are represented by entities or Unicode in the XML files, further complicates things. Admittedly, extracted indexes are possible and are quite valuable and practical in the right circumstances; but the

naïve assumption that 'the index can just be computer-generated' is almost always mistaken.

Embedded indexes are a step in the right direction. These are indexes where the entries are embedded within the XML file at the locations they mean to identify. This overcomes one fundamental limitation of extracted indexes: the proper index term does not have to appear as such in the text. Embedded indexes enable the identification of the subject 'Nixon, Richard Milhous' with a paragraph that only uses the term 'Tricky Dick.' Most embedded indexes are actually little more than keywords, accommodating only a single level (although it is certainly possible, with some effort, to reflect entries and subentries) and generally indicating a point, not a range, in the content. As such, they resemble 'open-system indexes' except that they usually point to content below the document level. Nevertheless, the embedded entry 'Nixon, Richard Milhous' rarely tells us whether it is referring to a phrase, a sentence, a paragraph, or more. Limitations like these are very frustrating to professional indexers, who have a much more sophisticated and nuanced understanding of the content and its structure. Even more, they are frustrated by their inability, when creating such an embedded index, to use the professional indexing software that most of them rely on – stand-alone indexing programs like Cindex, Macrex, and Sky. These programs cannot be used to create embedded indexes because they have no direct access to the text being indexed. Instead, indexers find themselves scrolling through tagged documents, perhaps using XML editing software like XMetaL or XMLSpy, tools which are very useful for tagging and processing XML documents but which have none of the sophisticated features of professional indexing tools. They have no good way to assess the density and proportion of the index being developed, to manage the structure of entries and subentries conveniently, to handle double posting well, and so forth.

What is needed is a way to enable indexers to interact effectively with XML files without sacrificing the use of their professional indexing tools, and to be able to express a richly structured back-of-the-book index within the indexed XML files. Publishers and software developers are working on this problem; currently, most of the solutions are proprietary. The one I am familiar with is one developed by my company and which we have used successfully on several occasions. Known as Impressions Integrated Indexing™, it enables indexers to use their professional software tools to develop indexes as they normally do, with all the power of Cindex or Macrex or Sky at their disposal, and to associate each entry and subentry very precisely with the indexed content. The resulting integrated index is a document separate from, but closely related to, the XML documents being indexed. Those documents include a special system of tags that associate the index entries with ranges of text – words, phrases, passages, paragraphs, pages, or more – in a method that allows for unlimited nesting or overlapping of such cited ranges. When the XML documents are used to produce print pages, the system generates the page references and embeds them into the separate XML-encoded index file used to produce the typeset back-of-the-book index. When that same XML file is used to produce a new set of pages – either for a revision or for a new page format – the new back-of-the-book index can be generated automatically. And when that XML file is used to create an electronic product, the precise range of text that each entry or subentry refers to can be linked and highlighted.

Conclusion

As the universe of information available to us continues to expand, the ability to identify and locate exactly the information we need becomes ever more important. No single technique or technology is sufficient for this task. Librarians may disparage Google, but 'to Google' has become a verb whether we like it or not. We need full text search, we need metadata, and we need intelligent, well-structured, well-proportioned indexes that direct us to very granular portions of documents. Extracted indexes have their place; embedded indexes – when done as a sophisticated form of open-system index at a very granular level – can be extremely useful; and we are beginning, through techniques like integrated indexing, to bring the full power of professional, closed-system, back-of-the-book indexing to the digital world. In all of these cases, XML is the technology that will enable it to happen, and professional indexers are the best people to make it happen.

Acknowledgments

Thanks to Martin Tulic for directing my attention to the two cited articles, and to Meg Hannah, Senior Scientific Editor and Indexer at Impressions Book and Journal Services, who, with her colleagues over the years, has helped me to understand and appreciate the art and science of professional indexing.

References

Barnum, Carol, Henderson, Earvin, Hood, Al, and Jordan, Rodney (2004) Index versus full-text search: a usability study of user preference and performance. *Technical Communication* **51**(2), May, 185–206.

Klement, Susan (2002) Open-system versus closed-system indexing: a vital distinction. *The Indexer* **23**(1), April, 23–31.

Bill Kasdorf is General Editor of The Columbia Guide to Digital Publishing*, a major print and online reference work published in 2003 by Columbia University Press. Bill is Vice-President of Apex CoVantage, and formerly President of Impressions Book and Journal Services, a Madison, WI composition and publishing services firm that designs, edits, and produces books and journals in print and electronic forms. Impressions has developed a national reputation for the effective and practical application of technology, especially SGML, XML, and PDF, to the production of both books and journals. Past President of the Society for Scholarly Publishing, Bill is a frequent speaker and seminar leader for SSP, as well as for other publishing industry organizations such as Seybold Seminars, BookTech, the Council of Science Editors, the Library of Congress, STM, and the Association of American University Presses.*