CASE STUDY

# Moving from Single Sourcing to Reuse with XML DITA

*Lori Fisher, Program Director for Data Management User Technology, IBM® Corp.*

The concepts of single sourcing and information reuse have been talked about for many years in the technical communication community, but few (if any) companies have really succeeded in the full sense of these terms. There are two inhibitors to successful reuse:

- lack of training in how to design our information to be readily reusable
- lack of authoring tools and technology to effectively assist in the mechanics and implementation of reuse

In this article, I examine both of these issues and address a potential solution available by integrating information architecture with an XML DTD in a publicly available offering called DITA (Darwin Information Typing Architecture).

> ***"Reusable information is of course single-sourced. …but not all single-sourced information can be reused…."***

## Some Definitions

Let me begin by defining my use of the terms *single sourcing* and *reuse*. In this article, I am referring to two distinct but related concepts; both have suffered from the lack of design knowledge and the lack of tool support that are essential to successful implementation.

*Single sourcing* refers to authoring a piece of information once and being able to derive multiple output formats from that single source. For example, many companies author information in Adobe FrameMaker™ for their books and process that same source material using various tools such as WebWorks™ from Quadralay to produce online information delivered as HTML help systems or using Adobe Acrobat™ to produce electronic PDF books. Within IBM, we have successfully single sourced book content, online help files, and electronic books for many years from our SGML source files. These files can be delivered in different output formats with no modification of the content, given the right transform technology and given the right structure of online help systems. We are able to do this because the files are being used in the same context and for the same audience (for example, Product A User's Guide, Product A Help, Product A PDF book), where "Product A" is the context for the information, no matter the output format. This achievement is much easier than reusing content across multiple contexts.

*Reuse* refers to authoring a piece of information for one intended context and audience and being able to reuse this information in a distinct and different context. For example, a company might be developing a set of online topics for software Product A that also covers tasks related to the base operating system; if the company can reuse information originally written to document the operating system *without changing that operating system documentation for the new context* in the Product A help system, this is successful reuse. Ideally, in the future, information will be written that does not assume any initial context or audience and can be reused in multiple contexts without modification. Such reuse requires more than smart tools to transform output formats; it requires a significant change in the way we design and author information.

So, stated more generically, reuse refers to designing a piece of information so that it can be reused in multiple contexts and for multiple audiences without modification. Reusable information is of course single-sourced. It is authored once and delivered in multiple formats. But not all single-sourced information can be reused—across multiple contexts and for multiple audiences.

The more difficult goal, then, is to create reusable information from a single source. Therefore, for the remainder of this article, I will refer to information reuse as the strategic

Lori Fisher
Program Director for Data Management User Technology
IBM® Corp.

lorif@us.ibm.com

Lori Fisher works at IBM's Silicon Valley Laboratory as Program Director for Data Management User Technology, an organization of information developers, human factors engineers, graphic and visual designers, and user assistance programmers working on data management software products. She teaches two of the core courses in a certificate program in Managing the Development of Technical Information at University of California Extension. She has served on the STC Nominating Committee, chaired the STC Quality SIG, and judged in various STC competitions. She is a Fellow of STC and currently serves as an officer in the Silicon Valley Chapter.

goal we are trying to achieve as information-development managers.

### The Advantages of Information Reuse

Why do we need to move beyond single sourcing to reuse? One factor, of course, is cost. Companies have growing volumes of documentation yet need to improve the efficiency with which they create customer documentation for their products. Often, new products are built around pieces of an existing product line, but the information can rarely be reused intact for the common componentry. Even within the information for a single product, we may want to reuse information for varying audiences (system administrator versus programmer, for example) or across tasks. Also, more and more companies are creating integrated suites of products, or customized solutions, which combine existing product elements into new offerings for the customers, requiring information to be integrated and combined in flexible ways.

> *"Designing for reuse is a critical prerequisite for being able to deliver… a dynamic help system, because users will control in what sequence and what context they view the information."*

Quality is also a factor. Information that can be reused across multiple tasks, products, or offerings but updated from a single source is more likely to be accurate—as well as cheaper to maintain. The information is more likely to be accurate because the source is being updated in only one place (another traditional advantage of single sourcing), thus reducing the likelihood of human error or process errors that do not find all instances of a particular fact needing updating. The information is likely to be cheaper to maintain because there is a reduced volume of information to be managed internally and potentially less volume to be distributed to the customer (which can mean lower product cost because of reduced storage and media costs).

Most important, from a reader's point of view, information explicitly designed for reuse will be clearer and more usable than information designed for one context and force-fit into another. Today's help systems that are single sourced from linear book-model files are often awkward and not optimized to the current task context. Information designed for reuse can be more specifically mapped to user tasks in new contexts, can be more appropriately mapped to the audience, and is more likely to be accurate and up-to-date.

In the future, there may be additional advantages to reuse, by enabling us to deliver and display information to customers in new ways. For example, the ability to deliver information as a database of topics would allow readers to dynamically customize the information they see. Designing for reuse is a critical prerequisite for being able to deliver such a dynamic help system, because users will control in what sequence and what context they view the information. The advantage for customers is information that is personalized for their attributes and customized to their patterns of use.

### The Inhibitors to Successful Reuse

Even companies that have successfully single sourced their information for years have had difficulty taking the next step to true reuse of information. There are two key reasons for this:

- We have not adequately designed our information to be readily reusable.
- Authoring tools and technology have not been available to assist in the mechanics and implementation of reuse.

Let me briefly address both of these inhibitors.

### Designing Information for Reuse

You have probably struggled with even the simpler task of single sourcing books and online help. Why is it difficult to create both books and online help from a single source? A book (other than pure reference material) is by definition meant to be read sequentially. It assumes in most cases that one has read the prior paragraph before beginning the next paragraph, which therefore also implies the use of transitions and references forward or back. It also assumes that the reader is familiar with

mechanisms such as a table of contents, an index, or an appendix.

In contrast, online help does not necessarily require sequential reading of the content, so transitions from prior topics or chapters may no longer be meaningful and references may need to be more carefully defined as links or specific pointers. Book metaphors, such as an appendix, may not necessarily have meaning in an online help system. However, an online help panel may include a mix of information types, such as a conceptual introduction, a list of procedural steps, and a table of reference information. To single source both help and books from the same content means accommodating these differences and trying to optimize for the delivery medium without changing the source material.

Initially, we all know companies responded to the need to single source by "dumping books online." The information was not optimized for an online environment, but at least the books were available electronically! With more sophisticated tools and the use of features such as conditional text, most writing teams over time have been able to do a much better job of single sourcing book-metaphor and help-metaphor information while optimizing the content based on the output format. To do this, writers use conditional tagging within the source files to chunk the information into pieces that can be viewed as a network of help panels, given that the navigation paths are known and the audience is constant (for example, users of Product A), and therefore the context is constrained to a reasonable set of variables. Generally, our help systems still assume a certain level of sequential context, based on menu hierarchies or field-sensitive context. This dependence on sequence is an indicator that the information is not necessarily reusable across contexts, although within the above constraints, it can be single sourced.

Another indicator that our information is not reusable can be found in the opposite scenario. That is, when information has been consciously designed and optimized to be delivered as an online help system and the writing team is then asked to produce a printed book (or a PDF or an HTML book—the key is book-metaphor as the requirement). This attempt often fails miserably because the information was written for one specific context—a panel written as task help for a specific field, as an example—cannot easily be reused in the context of a traditional user's guide *without significantly modifying the content*.

Now, imagine the next step beyond single sourcing: designing truly reusable information. Not only must the information stand alone within a narrowly defined context (as it might in a help system), it must also stand alone when taken out of that original context and used for a different audience or different technical context beyond Product A. Now, there is no way to rely on what other information is available to the users, no way to rely on how they might have navigated to this piece of information, no way to know the sequence of tasks they might be trying to achieve when reading this particular chunk of information, no way to know if this information will be delivered as a book or a help system or a wizard or embedded in the user interface.

Not only does the information need to be authored to be context-independent, there must also be a way to identify the scope, content, and purpose of a particular piece of content, or a topic, so that it can indeed be reused. This identifying information is often referred to as information metadata. The metadata must also indicate the relationship of this topic to other topics. (Are there prerequisite topics? Is this information supported by other topics, such as a definition?) The metadata must also indicate the type of information. (Is this a reference item? Or a procedural step? Or an optional explanation of a concept?) Without extensive metadata to describe a piece of information, that information is unlikely to be efficiently reused.

In the past, we have not designed our information to be neatly categorized and described in these ways. We have not created distinct, independent topics carefully distinguished from one another by scope, type, and relationship, nor have we attempted to identify them in these ways. We have neither designed nor cataloged our information in such a way that it can be reused. Our content has been very closely tied to the form in which it would be presented—as a book or as specific help panels. The design of most of our current information relies on a specific context, sequence, or navigation path to be effective.

## Authoring Tools and Technology to Assist with Reuse

Since the 1980s and early 1990s, traditional WYSIWYG authoring tools have reinforced the design issues above by focusing on the form of the information, how it looks, versus the content. This focus ties information more closely to a given delivery or output mechanism, such as book format or help panel format. WYSIWYG authoring tools emphasize formatting elements, such as fonts and page breaks. This emphasis on formatting can make even single sourcing more difficult when content is moved from one medium to another because these elements may not transfer to the new medium without requiring changes.

In the 1990s, some information development teams began to look at SGML technology as an alternative authoring solution. SGML is a markup language that emphasizes the structure of information as opposed to the form. Authors identify structural elements, such as "procedure step" or "example," instead of indicating how those elements look, such as font or emphasis (bold, italic) or bulleted list. This was a step in the right direction but required information development teams to invest significant amounts of time to tag their existing information with accurate descriptions of the structural elements.

At the same time, as the Web evolved, writing teams were moving to HTML technology (a subset of SGML), sometimes in place of or in addition to their WYSIWYG tools and sometimes in addition to trying SGML. With the explosive growth of HTML, many companies moved to the simpler HTML-based editing tools as their primary authoring environment. Although HTML is also a tag language, it allows writers to use it to identify WYSIWYG-like elements, such as bold or bulleted list, instead of forcing the identification of content by its structure.

Those information development teams who persisted and implemented a rigorous SGML approach to their information are better positioned for information reuse than those still working in a WYSIWYG or HTML authoring environment. The separation of form from content, which SGML enforces, is a key advantage for information reuse because it forces the author to write content that is not tied to a particular output or display mechanism. In other words, it reinforces a medium-independent approach to information, which is the first step toward information reuse.

*"Those information development teams who persisted and implemented a rigorous SGML approach to their information are better positioned for information reuse…."*

However, even SGML does not automatically enable reuse if the information content is not designed with reuse in mind. SGML also does not inherently provide metadata tagging, or a mechanism for mapping information types and their relationships to particular contexts of reuse. The development of SGML DTDs has also proven costly and time-consuming, so most writing groups reverted to generic DTDs (such as DocBook), which then minimize the value that more customized markup might bring.

Content-management systems for file management have also had limited effectiveness for writing teams trying to manage large amounts of chunked information across multiple authors and directed to multiple output targets. Features such as version control and file history tracking are well-developed, but functions related to associating metadata with information or information typing for the specific purpose of supporting reuse are often missing.

Information development teams have used conversion programs, transforms, and a myriad of home-grown tools to try to address the problem of optimizing single-sourced content for new media or new contexts. Some of these efforts have been successful for narrowly defined repurposing of content, but the resource investment required is often daunting and certainly prohibitive for small writing teams.

## XML DITA As a Potential Solution

XML offers some promise as an authoring technology to help move us toward informa-

tion reuse. XML reinforces the separation of form from content and provides a way for writing teams to describe that content (the metadata). It is flexible enough to allow writing teams to customize the markup to describe very specific kinds of content. Describing the content specifically is necessary to then intelligently applying presentation styles or output formats. For example, a step in a procedure might be formatted differently than an ordered list used to catalog the parts in a package. The metadata differentiates these numbered lists as two different kinds of content. The metadata may also be used so that a certain kind of annotation is displayed as hover help online but as a figure callout when printed.

By the way, the metadata aspect of XML also enables more intelligent search by users; as a simple example, if a user needs to find a step in a procedure with the variable *account number* in it, the search can find all the information tagged as a `<step>` with the `<variable>` *account number* in it, which would yield a much narrower set of choices than finding all text instances with the words *account number* in them. The user doesn't need to know how the information is tagged, but the search can take advantage of this and other metadata to return smarter results.

But XML technology alone would not be enough to ensure the creation of reusable information. The information must not only be tagged and described (a la XML), but it must also be *designed* to be reusable. What is needed is an underlying, structured information architecture as the design principle for the content. DITA enables the development of modular information optimized for reuse. DITA includes a set of design principles for creating specific types of information in a highly modular structure. DITA is a set of XML DTDs, with a base that defines tags common to topics and with additional DTDs that build on that base. Together, these DTDs express principles for authoring modular information and for delivering that content in various ways, such as in online help systems or in Web portals or as PDFs. The base XML DTD included with DITA has about a hundred markup elements—not very complex—and most of them are very familiar to anyone who knows SGML or HTML.

> ***"What is needed is an underlying, structured information architecture as the design principle for the content."***

The key to the DITA architecture is authoring information structured as specific information types, in highly modular chunks called topics. A topic is a discrete piece of information, independent of other topics, covering a specific subject. Topics are categorized as specific information types—task, concept, or reference information. In contrast to an online help panel that might include a mix of information types, such as a conceptual introduction, a list of procedural steps, and a table of reference material, a topic includes only one of those information types. It can be combined with other topics, however, to provide the reader with the variety of information contained on a traditional help panel. The key here is that this topic can then be recombined with other topics in a slightly different context or for a different medium (in a PDF for example), yet the source is written only once. DITA also allows the writer to describe relationships between topics. For example, a writer can describe chronological, frequency, or priority-based relationships among topics.

The second advantage to DITA, beyond the information-typing architecture, is the support for easily modifying and extending the underlying XML DTD. The "D" in DITA for Darwin refers to the ability within DITA to "specialize" and "inherit" aspects of the design of topic types. Starting with the base DTD, information development teams can create a customized topic type that is defined relative to an existing topic type, so you need to define only the *delta* markup and rules for that type, while inheriting the markup and rules for the remainder from the standard type in the DTD. This simple concept is a powerful advantage and can significantly reduce the resource needed by information development organizations to customize an XML DTD that allows you to truly optimize your information for specific characteristics of your audience or technical

### References

An excellent introduction to DITA and the concept of information architecture was published in the August 2001 issue of *Technical Communication* (Vol 48, No 3), titled "DITA: An XML-Based Technical Documentation Authoring and Publishing Architecture" by Michael Priestley, Gretchen Hargis, and Susan Carpenter.

An overview of DITA, along with sample code and a sample DTD, is published at <www.ibm.com/developerworks/xml/library/x-dita1/index.html>


context. This concept also allows writing teams to change and adapt the DTD to their needs over time, without requiring expensive, complicated rewrites of the entire DTD. The ability to customize the DTD to create content-specific markup within your topics is also key to providing advantages to your readers, such as intelligent search (for example, search only the procedure steps within this document or topic).

In addition, DITA supports the use of both generalized and specialized XSLT transforms to process the information topics for delivery in various output formats. XSLT is a programming language specifically designed for use with XML stylesheets to create rules for how to process particular XML elements, for example to define what a heading looks like in a help system versus on a Web site. As with the DTD, DITA allows the writing team to reference a base transform and customize it for specialized processing of topics by defining only the delta changes and inheriting the rest of the existing transform. Again, this specialization can be a key timesaver for any information development team and also provides flexibility over time, as new deliverable mechanisms are needed, with minimum effort in recreating entire XSLT transforms.

### Summary

To create reusable information will require hard work by technical communicators. We will need to rethink the way we design our new information, and we will have to re-architect existing information into small, independent topics structured by information type. We will need to describe the content of our topics in ways that allow it to be easily reusable, using technology such as DITA's XML DTD. And finally, we will need to leverage other tools and technology, such as XSLT transforms, to perform the mechanics of delivering our reusable information in various output formats and for various media.

The reward for this effort is more accurate information that is optimized for the reader's context and for the output mechanism and that requires less maintenance and less rewriting by the information development team.



## Mark Your Calendars for the Best Practices 2003 Conference

Join Dr. JoAnn Hackos, the managers and staff of member departments of The Center for Information-Development Management, and the community of industry experts she brings together yearly at the 5th annual Best Practices Conference.

### Conference

September 22–24, 2003

### Post-Conference Workshop

September 25–26, 2003

### Location

Seattle, Washington
The Edgewater <www.edgewaterhotel.com>

### Theme

**Innovation: Making It Happen** based on Malcolm Gladwell's book, *The Tipping Point* (Back Bay 2002).

Around this theme, we want to focus on introducing innovation into our organizations, which means

- learning from the best practices of others
- uncovering new ideas for innovations (what practices would improve our service to customers, reduce costs, gain time for creativity)
- persuading management to support innovative practices
- encouraging staff members to embrace innovations rather than fight them
- measuring the value that an innovation may bring to customer quality or cost reductions

Watch for additional information about the conference at **www.infomanagementcenter.com**.