DITA Detours: Bypass Four Costly Roadblocks

Executive Summary

The emergence of Darwin Information Typing Architecture (DITA) as a general-purpose XML schema for technical documentation represents a significant step forward and opportunity for organizations that create extensive amounts of technical documentation and other related content. DITA offers organizations a ready means for structuring product-support content and the rapid adoption of DITA by XML technology vendors provides a number of options for implementing DITA productively.

Indeed, few vendors have hesitated to announce their "support" for DITA, even when that support is still not fully understood or defined. Moreover, there is a temptation for DITA to be positioned as a comprehensive and complete solution (just add water and stir!) In fact, the reality of implementing DITA is still evolving. At the same time, a community of knowledgeable users is beginning to emerge around DITA, complete with its first experienced practitioners.

While there are many compelling reasons to adopt DITA for developing and structuring product-support content, one of the most interesting aspects of DITA is *specialization.* Information developers can use specialization to customize the core DITA schema; this way, new information types can be introduced to a DITA application by extending the current application and maintaining core tools, processors and data structures.

DITA provides a significant new opportunity for XML's role in content management, where XML can become the underlying structure for a much higher percentage of an organization's content. Until now, organizations have implemented XML-structured content as a focused solution to a specific problem within an organization—for structuring documents in highly regulated industries such as the FDA or in a vertical market where an XML approach already exists, such as the aviation or automotive industries. With DITA, organizations can now look at applying XML structure to a broader array of content, beginning with technical documentation and other kinds of product support content, but potentially expanding to many other kinds of documentation and content.

However, even though DITA is clearly ready to step into a larger role, not every organization is ready to embrace out-of-the box DITA solutions. In fact, many companies need to conduct a thorough due diligence to ensure that their workflows, resources and technology are ready before they can reap the full benefits of a DITA implementation.

This paper discusses DITA in general—its background, current status, and applicability for product-support content and its ability to support this next step forward in the widespread adoption of XML. But the key thrust of the paper is the significance of specialization, and what the ability to develop DITA specializations means for organizations. We will also highlight four "roadblocks to success" with any DITA implementation. Failure to at least consider the implications of these roadblocks could lead to project delays and cost overruns, which in the long run would erode the cost and time savings of deploying DITA.

Introduction: Understanding DITA and Topic-Based Information Development

The Darwin Information Typing Architecture, or DITA for short, has quickly become a subject of keen interest to the technical documentation community. An XML-based approach to authoring and distributing topic-based materials, DITA was published in May of 2005 as a standard by OASIS, the Organization for the Advancement of Structured Information Standards. Even prior to its publication as a standard, many organizations began using DITA for developing documentation and other product-support content such as Help systems. As of October 2005, major companies such as Adobe, Autodesk, IBM, and Nokia are actively using DITA in their day-to-day operations.

Background

Why has DITA gained so much traction so quickly? Simply stated, DITA provides ready mechanisms to two business challenges facing companies that produce complex products, which require substantial and complicated supporting documentation and content. Producing such content efficiently means companies require at least two things:

- A means of structuring the content in a media-neutral format that enables ready *repurposing* of the content to print, Help, HTML and other formats.

- A means of identifying content chunks or modules for *reusing* the content in several different products and contexts.

Such *repurposing* of content in multiple formats has become a well-established practice in technical documentation groups. Even commercial authoring solutions like Adobe FrameMaker provide tools for creating the print, HTMLand Help versions of the content from a single file. But *reuse* has been a more difficult problem to solve. Because reuse requires a content structure that allows chunks of content to be readily combined and recombined, it immediately exceeds the capabilities of a commercial authoring solution like FrameMaker, which is most often used to store content in large, chapter-length files.

When organizations have looked closely at their requirements for reuse, they have often concluded that they need to do two things differently than before:

- They need to adopt a content structure that affords the ability to flexibly create and manage reusable chunks of content. This often leads them to look at *the eXtensible Markup Language, or XML.*

- They need to approach the creation of the content in a new way, where the content is developed as single units of information that can then be combined and recombined into different products. This often leads them to *topic-based approaches to content development.*

This is precisely where DITA comes in, by providing an XML-based approach to topic-based content development.

The DITA FAQ defines a topic as "a chunk of information organized around a single subject. Structurally, it is a title followed by text and images, optionally organized into sections." If this sounds simple, it is. The core DITA tagging is meant to be straightforward and extensible. The initial set of topic types is oriented toward the types of information most often created for product support: *tasks,*

reference, and concepts. These types of information are familiar terrain for technical communicators and others involved with product support.

- The *task* topic type is used to create procedural materials such as step-by-step instructions.
- The *reference* topic type is used to create content that provides quick access to facts.
- The *concept* topic type is used to provide general-purpose background information.

Together, these three content types provide a comprehensive starting point for many organizations that produce product-support content. Also, because of the discrete chunking of the content and the XML markup, this approach enables reuse much more readily. As recently noted in an article in *CIDM Information Management News*[1], if the content is properly authored as stand-alone topics, the topics can be reused in different contexts and topics from multiple components can be integrated into a solution. According to the authors, "Reuse flows from the topic-based paradigm."

History

While DITA is relatively new, as well as the specific terminology around "topic-based writing," the goal of reuse has a lengthy history in product support, especially in certain vertical industries. Beginning in the 1980s, several large industries began using structured markup to create product support documentation. The Standard Generalized Markup Language (SGML), the precursor of XML, was used to create specialized tag sets for the U.S. Department of Defense, the Air Transport Association, and the automotive industry. Some of the largest companies in these industries—Boeing, General Motors, United Technologies—have created and still maintain millions of pages of technical documentation that have been encoded in SGML. Since then, many of those pages have been migrated to XML.

These large companies had the same goals for SGML that companies now have for DITA: create content in a neutral format that can be readily *repurposed* into different formats and create chunks of content that can be *reused* in different products and in different contexts. In aviation, for example, companies like Boeing have successfully created a single task (for example, for removing a part) that is then used in several different manuals and for several different aircraft. Simple elements, like a standard warning that needs to be appear when a certain tool is used, can be created once, edited once, and used as many times as needed.

Yet whereas these efforts were successful, the conventional wisdom was that such approaches were only reachable by the largest organizations that made the capital investment necessary to achieve these results. Moreover, companies in selected industries had the benefit of an agreed upon SGML tag set that had been created specifically for them. The U.S. DOD, for example, had specific SGML Document Type Definitions (DTDs) that vendors such as United Technologies and Raytheon were free to use. Outside of these and a few other vertical industries, product support professionals did not have a ready mechanism for working with SGML and later XML. One general-purpose DTD did emerge in the 1990s for technical documentation. The DocBook DTD was created in 1991 as an SGML DTD and is now available as an XML DTD.[2] However, even though DocBook has been widely adopted, it is not readily adaptable to topic-based authoring.

---

[1] "Topic-based Authoring: It's Not All About the Technology; Why Content is King and Technical Editors are Critical to Topic-based Authoring," Michelle Corbin and Jana Jenkins, IBM Corporation, *CIDM Information Management News,* June 2005

[2] DocBook, like DITA, is an OASIS standard, first formally published in 1998.

A Brief History of DITA

Information Development at IBM

High-tech manufacturers such as IBM are among the most prolific creators of product support documentation. IBM created one of the earliest systems for creating complex documentation, Bookmaster, and many of the key developers of SGML also came out of IBM. Document processing tools and approaches that were used at IBM became commercial tools and standard practices in hundreds of other companies.

Like many other groups, however, IBM saw opportunity in the greater flexibility of XML, and the many tools and technologies that were growing up around XML—notably the Extensible Stylesheet Language (XSL) family. In 1999, IBM formed a cross-company workgroup to look at further exploiting XML in its product-support content. They considered approaches like converting their existing SGML DTD to XML, or using an existing XML DTD such as DocBook. In the end, the group focused on how to best effect reuse, where a single source of generically tagged modules of content could create multiple content products that could be published in multiple formats. The focus on reuse and topic orientation led to the first designs of DITA, which was published on IBM's developerWorks site in March of 2001 after a year of internal prototyping.

DITA and OASIS

In March of 2004, IBM donated the DITA technology to OASIS, and simultaneously handed over governance of DITA to an OASIS Technical Committee (TC). The TC put out a call for participation that same month, with founding members representing IBM, Nokia and OASIS. Innodata Isogen was also a founding member and continues to serve on the TC. The formal standard was then published in May of 2005, and the TC is currently working on requirements and design for subsequent versions of the standard. The TC has since grown to include companies from different industries and additional countries, including BMC Software, Idiom Technologies, Intel, Sun, and Boeing.

DITA in Action

As mentioned previously, a number of large companies have already adopted DITA. Significantly, a number of major projects have already been completed using the DITA DTD:

- Adobe Systems recently completed a massive documentation project using DITA. Adobe's Creative Suite 2 is their flagship retail product, and ships it worldwide in 14 languages, with over 110,000 pages of printed documentation and Help. The Instructional Communications group at Adobe used their own product, FrameMaker, as the DITA authoring and publishing tool and a content management and globalization system from Idiom Technologies.

- Autodesk has moved the majority of its user documentation to DITA, though some teams are still in transition. Documentation deliverables include Online Help, PDF books, Online and PDF tutorials, Installation Guides, Getting Started Guides, and Command References. The company is also using Structured FrameMaker as the authoring tool and the content management and globalization system from Idiom Technologies as the underlying database.

- Graphics hardware giant ATI Technologies has licensed TEXTML Server from IXIASOFT to manage their technical documentation. ATI will be using TEXTML to manage DITA-encoded content and will be using XMetaL Author from Blast Radius for XML authoring.

- Software vendor Sybase is also adopting DITA, and has licensed XMetaL Author DITA edition as a specialized tool for authoring and publishing DITA-encoded content.

- IBM, of course, uses DITA widely throughout its product support operations. As it continues to evangelize DITA, IBM has provided speakers at numerous conferences and professional meetings, and continues to publishing tools, background, and training material on DITA.

Understanding DITA Specialization

What is Specialization?

Designed as an "XML-based, end-to-end architecture for authoring, producing, and delivering readable information as discrete, typed topics," DITA is intended to be built on and customized for more specialized uses. While DITA includes the core information types of *concept*, *task*, and *reference*, the designers of DITA fully expect people to customize DITA to their own needs. This is where *specialization* comes in.

Specialization allows a developer or organization to extend one of the existing information types by adding or modifying elements, or to create whole new information types. An often cited example in discussion of DITA specialization is programming documentation, where technical writers have to document an application programming interface, or API. Documenting an API often requires the technical writer to create a highly specific reference document jammed with code examples, processing rules and other technical data. Such details are not explicit in the generic definition of a *reference* in the DITA DTD. The writer or organization would likely want to—and need to—specialize the *reference* information type or create a new *API reference* based on the DITA structures. They could then add the additional, necessary structures.

In addition, specialization enables DITA users to define structures specific to their information requirements and business processes but that is still explicitly mapped to the core DITA types. This ensures that all conforming specializations of DITA base types will still reflect the core DITA-defined structures. Moreover, specialized DITA information can be interchanged not only with users, but also with base DITA processors.

Why Specialize?

There are many reasons to do specialization, but specialization is usually done for one or more of the following reasons:

- *Provide better authoring support.*
  For example, an API reference topic must have the subsections "parameters," "return value" and "notes," in that order, the easiest way to ensure these subsections is to define an XML structure that requires a "parameters" element followed by a "return value" element, followed by a "notes" element. The base DITA structure for reference topics simply allows "section" elements, so specialization is necessary to define the needed rules.

- *Enable specialized formatting or other processing* that is not part of the default DITA processing

- *Support other information processing within an enterprise.*
  For example, you might have a non-documentation-related information system that extracts important information from the documentation source.

Within an enterprise, specialization also enables different groups to create specialized document types that meet their specific needs in a controlled and managed way. In a sense, the cost of doing the specialization can be determined with some accuracy and compared against the immediate and long-term business value of doing the specialization. The specialization is controlled because it must conform to the DITA-defined rules for specialization, which ensures that specialized information will still be interchangeable with other DITA information and DITA-aware processors.

Two things often happen when large enterprises adapt XML for technical documentation:

- Different groups create their own XML document types and processing systems with little or no coordination with other groups, creating incompatible silos of information.

- Or, the enterprise creates one overarching XML document type that tries to satisfy all the requirements of all the groups.

Both approaches have serious problems. The first approach creates chaos, represents duplicated effort, and impedes future interchange and interoperation of information assets. The second approach leads to either huge, bloated document types that become almost impossible to extend, understand and implement or highly generic document types that don't satisfy any group's requirements very well, leading to user dissatisfaction and misuse of the markup.

IBM's BookMaster offers a textbook example of this first approach. Sophisticated, but huge with more than 600 distinct element types, BookMaster often required six months or a year to get a new element type approved and implemented. This experience ultimately led to DITA's specialization mechanism.

With DITA-style specialization, organizations start with a fairly generic, but useful, base: the DITA-defined core information types and domain elements. DITA specialization then conducts the usual sort of information requirements analysis to determine what specializations are appropriate for a particular application, such as creating precisely-structured API reference documentation. From this analysis, organizations can then determine what specializations are required, setting the stage for a fairly accurate estimate of the cost of creating and implementing specializations on top of the core DITA infrastructure.

The Measurable Benefit of Specialization

Specialization is also measurable. When new requirements appear, such as a new product that needs slightly different documentation, managers can accurately determine the cost of satisfying those new requirements with additional specializations and then make an informed business decision. For example, a large enterprise might provide a centrally defined and maintained DITA-based system that satisfies most of the requirements for the different groups, but some groups might need more. Each of those groups could make individual decisions on whether to do additional specializations, knowing that their specialized information will still be interchangeable with other groups, because it is all defined in terms of core DITA types.

Another important aspect of specialization is that it allows the implementation itself to be distributed. The modular nature of DITA, coupled with modern programming techniques and languages, enables the supporting infrastructure to be modularized in a way that mirrors the specialization hierarchies in DITA.

This ability to modularize the supporting infrastructure enables both direct re-use of existing code modules and the ability to locally and unilaterally extend DITA processing. In the case of the centrally defined documentation system, a group that requires additional specification doesn't have to go back to the central support group and request the added functionality—they could simply implement the required features locally. Because the cost of this type of extension can be accurately estimated and measured, the group can make a well-informed business decision about whether to do the specialization and whether to bear the cost of implementing the additional features they need.

By contrast, the silo approach requires each new group to bear the full cost of developing their system from scratch, with little opportunity for higher-level oversight or cost control. On the other hand, the overarching document type makes it less likely that groups cannot get new requirements met, either in the time required or at all.

Specialization is also incremental. Organizations are always adding to an existing base, rather than starting a new, from-the-ground-up implementation. To implement a specialization, organizations normally only need to add to an existing DITA-based system, which makes the incremental cost of satisfying new requirements much lower than it would otherwise be.

Toward a DITA Community of Practice

While DITA has emerged quickly and gained impressive traction in a short time, it is still a new technology, with emerging approaches and best practices. As with many new technologies, there is a temptation among the technology vendors to overpromise and overstate the technology's capabilities.

As one writer famously quipped about XML several years ago, "It is the software industry's latest answer to world hunger."

So just as XML didn't solve world hunger, DITA does not solve every problem that organizations face in creating technical documentation, online Help, and other kinds of product support documentation.

Significantly, most organizations still need to do the key things discussed earlier:

- Adopt a content structure that provides the ability to flexibly create and manage reusable chunks of content.

- Approach the creation of the content in a new way, where the content is developed as single units of information that can then be combined and recombined into different products.

It is clear that the industry sees an answer to this second challenge in topic-based authoring, though this is still a new arena for many organizations. Some larger organizations—IBM, Adobe and others—are well on their way, but other organizations are still training and providing re-orientation to staff, while others have yet to even take up the task.

Adoption of DITA is at a similar stage. Some large organizations are deeply immersed; others are well underway; many others are just beginning to prototype or pilot an effort. What we know so far is that whereas some organizations are using the core DITA DTD with little modification, most are finding

reasons to specialize it. Both Adobe and Autodesk, for example, specialized the DITA DTDs for their own usage.

When is DITA not a fit?

While DITA has much to recommend it, there is a cost in adopting it. Not all technical writers are willing or able to write in the modular, context-free way that DITA tends to require. Using DITA can require more sophisticated authoring and content management tools than are typically needed for doing non-modular books. Thus, for some enterprises, DITA may be overkill.

In the case of legacy XML and SGML systems that cannot be easily mapped to DITA, it may be better to apply DITA ideas of modularity and specialization but not try to conform to the specific DITA structures, at least not right away. This approach provides much of the benefit of using DITA (modularity and specialization) without requiring you to rework all your existing data. In particular, the specialization mechanism can be applied with little or no change to existing documents. The modularization approach can be applied as needed, making as few structural changes as necessary.

In addition, even when legacy information structures can't be directly mapped to DITA structures, it is still relatively easy to implement transforms to DITA to supply DITA-conforming XML information to others if necessary.

There are several other reasons why DITA might not be a good fit for an organization:

- Companies have existing information structures and practices that cannot be easily redefined in terms of DITA's base types. This is actually quite common because the current DITA structures impose arbitrary constraints that are not universal.
- Organizations are essentially printing books that don't require re-use or modular delivery.
- Companies have no requirement to interchange modular information with other enterprises or groups within a larger enterprise.

Four Roadblocks to Success with DITA

At the same time, companies that do have a place for DITA need to take steps to address four potential roadblocks that could prevent them from achieving the full benefits of DITA. Failure to at least consider the implications of these roadblocks could lead to project delays and cost overruns, which in the long run would erode the cost and time savings of deploying DITA.

1) *Inefficient processes*

Automating an inefficient process will often fail to deliver the expected benefits and cost savings from the overall DITA investment. Organizations that implement DITA need to take a good look at their existing processes to ensure that they don't inadvertently replicate redundant workflows or other business processes that create unnecessary work or costs. In addition, they also need to review their content management models to ensure that information is easy to locate and that they continue to leverage existing materials.

2) *Organizational boundaries*

In many companies, departmental or divisional content solutions are often not integrated into a unified approach. This often leads to:

- unnecessary duplication of effort
- technology systems that are not integrated
- increased costs and turnaround times
- expertise that is only leveraged within a single department

Organizations can use DITA to focus on controlled specialization within the organization to provide a framework by which these problems can be avoided. For example, DITA provides an architecture that supports group-specific solutions that are still interoperable with the overall corporate solution and with other group-specific solutions. Another way to approach this roadblock is to implement pilot projects at the departmental level, but always stay focused on the "big picture" for the enterprise.

3) *Technology limitations*

Many companies may be underestimating the level of expertise and effort involved to implement DITA smoothly. Many vendors claim to have "out of the box" DITA solutions, however, they should not be considered complete with respect to the specific requirements of a particular enterprise.

This means that organizations will need to customize DITA, starting with a complete content or information requirements analysis to determine what additional specializations or refinements are needed on top of the core DITA base types.

In other words, organizations will still need to:

- Perform format analysis and style sheet development
- Engage in legacy data conversion
- Integrate DITA modules with other necessary tools, including digital asset management applications and Authoring Support Systems

4) *Resource Utilization*

Organizations also need to use DITA as an opportunity to rethink how to best utilize skill sets of their staff members. Within many organizations, companies are asking subject matter experts to perform inefficient tasks, such as formatting technical documents. In addition, different groups may be creating similar content, or maintaining "reused" content in separate repositories. And other times, companies may be using expensive resources to perform copyediting and production, functions that could be outsourced with a minimal impact on the workflow.

In addition, DITA will also require organizations to focus on indexing content for search and retrieval, particularly descriptive metadata for topics and elements. This is much more sophisticated than just indexing because it may require abstract thought that needs to be performed by an experienced team member.

Moreover, many companies will find that maintaining reused content is a much greater burden than most vendors or users realize or want to believe. In many cases, a reused module of content may require slight differences in their content based on context. For example, a Ford Taurus and Mercury Sable may have identical features, but there are branding differences between the two vehicles that must be handled when producing an owner's manual.

Conclusion

Even though DITA is clearly a major step forward for companies considering the adoption of XML, it is still not a panacea for all content management problems. As an out-of-the box solution, DITA is still too general to satisfy the requirements of most users, especially authors. In addition, many core DITA types require specialization to be made useful and sensible. Using DITA also does not change the fact that organizations will require business-specific authoring, content management and publishing systems. While various tools are starting to provide some DITA support out of the box, this support cannot satisfy all specific requirements. Therefore, just as in any other XML deployment activity, organizations must plan for the development and maintenance of their authoring, management and publishing infrastructures.

As a rule of thumb, the 80/20 paradigm applies here. An out-of-the-box DITA tool kit can provide, at most, 80 percent of the functionality required to use DITA in production, which means that companies still need to acquire or develop the remaining 20 percent. This will also likely represent the majority of the system's implementation cost.

As a result, organizations should not expect the use of DITA to provide dramatic savings in initial system infrastructure development. However, DITA will provide dramatic savings in the flexibility of the final system to accommodate new requirements and DITA-conforming information structures. In the long run, most DITA-based systems will pay significant dividends by enabling much more cost-effective re-use, interchange and re-purposing of information assets.

To that end, organizations that have been considering XML for the authoring, management and production of sophisticated technical documentation – or those that recognize the need to make existing SGML- or XML-based technical documentation better suited to reuse and modular delivery – need to take DITA seriously. However, as in any implementation, organizations need to do so after a careful consideration of whether DITA first makes sense for their organization, and after an extensive review of their existing processes and resources to ensure that they do not incur excessive costs in deploying the full production system.