A Practical Implementation of DITA for Product Documentation

How PTC successfully implemented Arbortext® solutions and the DITA schema within its own operations to optimize its publishing and documentation processes–saving time, effort and money.

Overview

A long-standing problem at PTC has been our limited ability to reuse common content among different document deliverables. Today, this aspect of our documentation process causes a great deal of redundant work, and makes it costly to localize content for use in international markets.

To reduce the cost and improve the re-usability of our documentation, PTC determined that we needed a single-sourcing XML authoring environment–that is, to create one version of the content that can be used by multiple authors on multiple projects and be published in multiple formats.

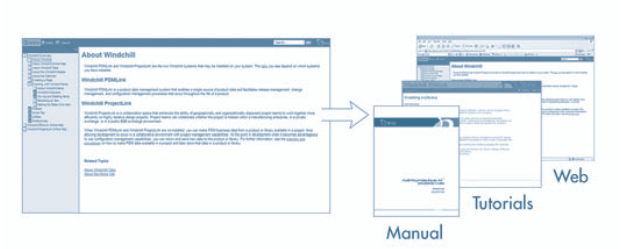
When PTC acquired Arbortext, Inc.–an industry leader in dynamic publishing solutions–in July of 2005, the documentation department at PTC was presented with a unique opportunity to implement and standardize on an XML authoring system within our own organization. Additionally, such a system would allow for single sourcing by supporting reusable content. Prior to migrating our entire documentation set to XML, we created a pilot project to test how we would both implement XML and use Arbortext® Editor™.

Recently, PTC funded a new product that integrates its Windchill® software with any of five third-party CAD/CAM applications. This project was a good fit for our XML pilot because it presented the documentation department with a requirement to create five manuals that shared about 70% common content. Additionally, this project required creating an online help system that contained much of the same procedural content as the manuals.

This project clearly offered PTC the opportunity to test the single-sourcing abilities in both XML and Arbortext Editor because of their high levels of content reuse among the deliverables. Also, since XML is able to store source content in a format-neutral state, the opportunity existed to test the reuse of content across different deliverable types, such as Web, CD, print and pdf.

In addition to our requirement to reuse content at a more granular level, at the onset of the project we also sought to resolve the following problems at the same time:

- What are the best practices for single-sourcing and collaborating?
- How do we plan to reuse content?
- How do we write and organize our documentation now that we have a greater focus on reuse?
- How do we compile multiple deliverable types from an XML source?
- How do we manage all of this data?



By using Arbortext authoring tools, it allowed for a single source of content that was shared and reused across multiple product documentation outputs.

Alternatives Analyzed

After investigating the benefits of XML technology, we realized that the use of XML solves our specific technical need for reuse in our documentation, and fulfills a corporate goal to use our own tools. XML has the following characteristics, which are particularly suited to this documentation project and our overall needs:

- Easy to code–writers with experience in editing HTML will find XML just as easy to code.
- Hierarchical–XML's hierarchical structure is suitable for the required manual and online help creation.
- Structured–XML intrinsically enforces structured writing through the use of small, functional, repeated component parts. Adherence to structured authoring ensures consistency, so that a user-procedure is always formatted the same way, and a functional overview is always formatted the same way.
- Highly Portable–XML is completely compatible with Java and any other application that can process XML, and can easily be transformed into HTML or PDF by using XSL.
- Extendable–XML can be customized to fit our exact needs. If we need a special tag, we can create it and incorporate it in our DTD.
- Tailored Output–XML can be constructed and processed in such a way to allow for conditional output.

XML Document Types Considered

What's a Schema?

When working with XML, 'schemas' are very important because they define the 'legal' building blocks of an XML document.

An XML schema controls the way content is created by doing the following:

- Defines elements and attributes that can appear in a document
- Defines 'child' elements, as well as the order and number in which they can appear
- Defines whether an element is empty or can include text
- Defines data types and default values for elements and attributes

Currently, two of the most popular XML schemas are Docbook and DITA.

Docbook

Docbook is a tried-and-true schema that has been around since 1991. Prior to introduction of XML, Docbook was an SGML standard. Docbook has the following benefits:

- Lends itself very well to book-structured documents
- Proven architecture
- Authors with experience in SGML may have experience with Docbook

DITA

The Darwin Information Typing Architecture–DITA–was originally designed by IBM and then released as a standard that is now maintained by OASIS. OASIS–Organization for the Advancement of Structured Information Standards, is a not-for-profit consortium that drives the development, convergence and adoption of open standards for the global information society.

DITA divides content into small, self-contained topics that can be reused in different deliverables. The extensibility of DITA allows you to define specific information structures and still use standard tools to work with them.

DITA allows for a number of topic types, such as Task, Concept and Reference. Within DITA, a Task topic is intended for a procedure describing how to accomplish a task. It lists a series of steps that users follow to produce a specified outcome. Concept information is more objective, containing definitions, rules, and guidelines. A Reference topic is for topics that describe command syntax, programming instructions, other reference material; it usually contains detailed, factual material.

Solutions Chosen

After reviewing the available alternatives, we decided to use Arbortext Editor with the DITA schema.

Why DITA?

We decided to go with the DITA schema for the following reasons:

- The topic is an organizing principle in DITA. As such, sharing and reuse are more easily accomplished with this type of modular content.
- DITA promotes the reuse of whole and partial topics through the use of content references.
- Through specialization, DITA accommodates new topic and element types, as needed, for specific projects or other departments internal to PTC.
- DITA uses extensive metadata to make topics both easier to find and easily leverageable in our online help output.
- DITA uses universal attributes for areas such as audience and platform to enable profiling and other information-processing techniques.
- DITA uses element names and structures similar to HTML.

Challenges Faced

In the process of implementing Arbortext and the DITA schema, PTC encountered a number of challenges:

- Training–Writers were unfamiliar with the XML family of markup languages and Arbortext applications. Solution: Instructor-led training was provided in these areas.
- Planning Deliverables–Planning is critical to writing in a modular style. Solution: Writing in modular style required that we plan out our content in a much higher level of detail than we had previously. We found it helpful to create extremely detailed outlines and then use them as guides when writing and organizing our content.
- Collaboration Strategy–We found that multiple writers who worked on this project needed to collaborate on a much more basic level. Solution: Our collaboration strategy included:
  - Establishing ownership of each topic.
  - Providing writers with easy access to all files.
  - Conducting frequent peer reviews to familiarize writers with each other's work.
  - Creating a file naming convention that easily relates the topic type, where the topic is used, and whether the topic is reused.
- Reuse of Content Across Multiple Products–In many cases where we reuse content, the content refers to specific product names. These product names may change depending on where the content was reused. Solution: We resolved this issue by using the profiling functionality in Arbortext Editor. Profiling is analogous to conditional text.

Challenges Faced (continued)

- Managing Source Files – Because DITA is topic-based, the authoring process generates a tremendous amount of files. Ideally, one would use a content management system (CMS) in conjunction with an XML publishing system. However, due to time limitations on the implementation of our own brand of content management system, we were unable to take advantage of a CMS solution in time for the pilot. Solution: We used Rational ClearCase to manage our files. ClearCase provided change control functionality, ensuring that no two writers were working on the same file at the same time.
- Creating XSL Style Sheets – Since our source content is in XML and our output formats are both HTML and PDF, we were required to create XSL style sheets to transform the XML content into these formats. Solution: Arbortext Styler provided the capability to create and edit XSL style sheets.
- Organizing and Structuring Deliverables – To maximize reusability in DITA, deliverables had to be structured differently. Solution: Before structuring our deliverables, we first had to define the granularity of our topics. Would a topic represent a chapter, or would it represent a paragraph of text? Next, we used a special DITA file, called a DITA map, to organize and create structure in our deliverables. Additionally, DITA allows for the embedding of topics within other topics to create structure while authoring.
- Handing-off Content to Localization – XML and DITA were new document types for our localization department to process. Solution: Localization of documentation is a costly effort. Our localization department was generally pleased with our decision to do a pilot project in XML. We worked very closely with Localization from the start, and provided early content handoffs that allowed them to adjust their process for this new document type.
- Writing Style – PTC's documentation typically has a linear style. This means that a traditional technical manual would have a slight narrative to it. This type of writing style is sequential and creates difficulties as content is often reorganized when it is reused. Solution: Developed a new modular writing style that supports content reuse.

Results

The PTC pilot project has successfully launched our documentation to a new level. As a result of this project, our departments now enjoy these benefits:

- Sharing content across multiple deliverables and deliverable types.
- Reduced localization costs – content only has to be localized once for each language. The localized XML source is then recompiled into the appropriate deliverable.
- Developed best practices for writing and delivering product documentation in XML. These practices may benefit our customers looking to optimize their publication processes.
- Maximized the reuse of content.
- Developed best practices on project collaboration.
- Used DITA and XML to create deliverables with the same look and feel as those authored in traditional tools, such as RoboHelp and FrameMaker.
- Created new internal standards for 'chunking' information.
- Adopted a new, modular writing style.
- Paved the way for content reuse among all of our products.

To learn more about how Arbortext can help your company create and delivery high-quality product information, please visit our website at: http://www.single-sourcing.com/

single-sourcing solutions, inc.
www.single-sourcing.com

2855-DITA-CS-1007