

..... |

Leveraging the Value of Translated XML Content



*Extending Reuse to Achieve Efficiencies
in Content Creation and Deployment*

Dorothy J. Hoskins

Leveraging the Value of Translated XML Content

Extending Reuse to Achieve Efficiencies in Content Creation and Deployment

Executive overview

The decision to create content in XML¹ frequently comes from the desire to save money in creating, reusing content (text, images, etc.), and in translating it. Establishing an XML content-creation and reuse infrastructure takes time and money, so there is a cost/benefit ratio that has to be assessed as favorable to make the shift to XML generate a return.

Due to the complexity of XML implementations, the return on investment may take several years to achieve. It simply takes a lot of money and time to get a robust system set up and to create enough content with it to start seeing returns.

One overlooked value of XML is the related development of translation memory. Every time an XML document is sent for translation, the text in it becomes part of the translation memory. If principles of reuse are applied, so that the same pieces of content are found in multiple documents, then the exact same translation memory can be reused. The more content that is translated and the more content that is reused, the greater the cost savings that are achieved by the system overall.

For executive management, planning for reuse within a single department will provide efficiencies. Planning for shared content reuse across multiple departments or company divisions will drive those efficiencies to greater extents and achieve a higher value return on investment.

If the concepts of content reuse are applied along with the use of translation memory, the costs of multi-lingual content development can be significantly reduced.

Consider:

- Efficiencies in reuse are multiplied by the number of reuse cases.
- Efficiencies in translation are increased by reuse with each language.

¹ Extensible Markup Language, a neutral, non-proprietary content format. Like HTML for web pages, XML is an international standard supported by the W3C. Many applications can create and use XML content, either natively or transformed into other formats.

- XML is much cheaper to translate than content in the form of MS Word documents, web pages, desktop publishing files or other application-specific formats.

In this paper, we will consider XML content development, the reuse of content, the use of translation memory technology, and how they combine for the benefit of companies with customers around the globe. In addition, we will consider what it takes to build content reuse into a business' content creation and deployment strategy to reap the maximum benefits for every reuse in every translation.

This white paper was developed by Dorothy J. Hoskins of the Novatek Communications XML Innovations Group. For more information, please contact Novatek Communications, 585 482-4070 or visit www.novatekcom.com.

XML: A Format for Content Structure and Meaningful Reuse

XML files are by nature “structured”, that is, they are created with rules to govern what kind of content pieces can contain other content pieces, in what order, and with what frequently. The rules that determine the structure are defined in a XML Schema Definition (XSD) or a Document Type Definition (DTD). The meaning, or semantics, of the structure can be developed to describe any type of content.

What makes XML particularly useful is that the elements that make up the structure can have names that mean something to human readers, but computer code can also “read” the structure of XML.

When technical writers use XML as their content format, they work with special XML editing application that help them create the structure that conforms to the rules of the schema or DTD. Having meaningful names also helps the writers understand what they are creating.

Even better, the regularity imposed by structure makes it possible to write applications that can find the path to specific parts of the XML and handle each part of it in defined ways. This handler code (called transforms if it creates a new output from the structure, or queries, if it is only used to return information about the content), is developed by XML specialists according to customer requirements.

Transforming XML also adds whatever code is needed to display the XML in a user-friendly way, as a web page, MS Word document, PDF file or other kind of document.

So XML is a special, well-behaved kind of content that can be made into lots of other kinds of content. It can also be split up, rearranged, merged and appended to provide many different outputs from the same source XML file. Also, XML files can be combined into larger documents, with the same pieces used repeatedly wherever they are useful (for example, Warnings, definitions and references, and boilerplate text).

Because XML is so flexible, it is used all over the internet in many applications (news feeds, photo-sharing, blogging, “mashups”, Software-as-a-Service) and in many types of transactions (Electronic Data Interchange (EDI), financial filings, order fulfillment, inventory analysis, supply chain management, personnel, etc.). XML is also becoming the preferred way for companies to manage large volumes of business content, such as technical manuals, training and quick reference manuals, catalogs, labels, and call center knowledge bases, where consistency and searchability are important.

Finally, XML supports metadata, which is information added to the content that tells who created it, when, what revision it is in, what type of product or service it relates to, and other important business information.

Reusable Content Models

The amount of reuse that can be achieved depends heavily on two factors: the writers' consistent use of writing styles, and the model (DTD or XSD) of the XML. Without writers adhering to consistent wording, there can be numerous little variations in sentences. It takes discipline, coordination and compromise to achieve reuse from the writing standpoint.

As for the content model, there are three main options: adopt an existing standard model that is already in use, adjust an existing model for your own purposes, or create an entirely new model. The first approach usually gets an XML team going faster, because many XML editing applications are preconfigured to work with approved, standard XML models such as DocBook and DITA (Darwin Information Typing Architecture). For technical content, such as a user manual, quick reference guide, training manual, or Help files, many writing groups formerly used the DocBook model. Now, many are adopting the DITA model. DITA was designed to let writers create content as discrete topics, and then share the topics by collecting them together for the final document delivery. There are tools that can take collections of DITA topics and create sets of linked web pages, help files or a PDF "out of the box."

Both DocBook and DITA have a large number of elements, which can make them hard to learn to use and unwieldy in the XML editing applications. Accordingly, sometimes these standards are "subset", that is, the elements that writers don't need are suppressed or removed.

The second approach to content model development is to adjust an existing XML standard model for your own needs. Rather than "reinvent the wheel", developers can borrow from the best models they can find, and then adapt them.

The model described in the case study later in this paper took this adaptive approach. The model was designed using the existing standard XHTML tags (also used in DITA) for foundation elements for paragraphs, links, lists and tables. The basic building-block elements were then wrapped in higher-level structures specific to troubleshooting codes. The main custom elements are the troubleshooting code name, its description and related pairs of causes and actions. This model makes it easy for anyone who has ever worked with web page code development to understand the basic building blocks of the causes and actions. A great deal of thought went into the modular design of the various elements and how they would be processed and translated. As a result, it is a highly efficient XML model designed to be lean and get a specific job done.

The third option, creating a new model from scratch, can sometimes be the worst case for XML models: a "home grown" model, where the elements of the structure are arbitrarily named and the future use of the model by other groups or for other content is not taken into consideration. Home-grown models that are created without consideration of long-term use may make it almost impossible to reuse content across multiple projects and between departments, such as between training and technical publications groups.

Translating XML

A major benefit of XML as a file format is that the presentation of information is handled separately from the structure and meaning of information. XML structure is a framework holding the text. Typically there isn't any formatting (information about text size, color, position, etc.) in XML, only structure and text.

In non-XML documents, by comparison, formatting is mingled with the text. An example would be a Word document with colored text, bolded and italicized words, and embedded images. For translation, most of the formatting information is irrelevant and is removed. The text is translated in specialized applications that allow translators to view English text segments and translated text segments side by side. Then the translated text is put back into the original file format. However, the translated text has to have formatting reapplied to it to match what was in the English file. This takes time and quality control, so translation companies include this effort in their cost estimates.

Since XML content doesn't contain formatting, it is immediately cheaper to translate. The translated text is just inserted into the XML structure. The new document can be manipulated in all the same ways that the source XML file can be, making it possible to write code that switches text language without changing anything about its functionality. For example, an Adobe Flash application can give the user a method to set a language preference, and then load the correct translated XML file without impacting anything else. This approach is often used for screen shots and simulations where the background images remain the same, but the text changes to different languages.

As an example, here is some "raw" XML code in two languages. The same elements are in each example, although the order is different in places, due to differences in language syntax. The inline-library-reference elements pull in commonly-used words from an XML resource file. These resource files of common words and phrases, such as material and part names, are a major way to achieve reuse in XML content.

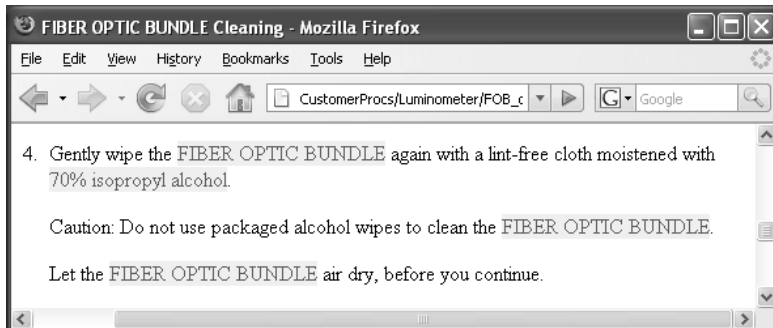
English XML:

```
<step>
  <cmd>Gently wipe the <inline-library-
reference class="- topic/inlinelibraryreference "
href="../../../../Common_XML_files/PartNames/Lu
min_FOB-long.xml"/> again with a lint-free cloth
moistened with <inline-library-reference class="-
topic/inlinelibraryreference "
href="../../../../Common_XML_files/PartNames/alc
ohol-long.xml"/>.</cmd>
  <info>
    <note type="caution">Do not use
packaged alcohol wipes to clean the <inline-library-
reference
href="../../../../Common_XML_files/PartNames/Lu
min_FOB-long.xml"/>.</note>
  </info>
```

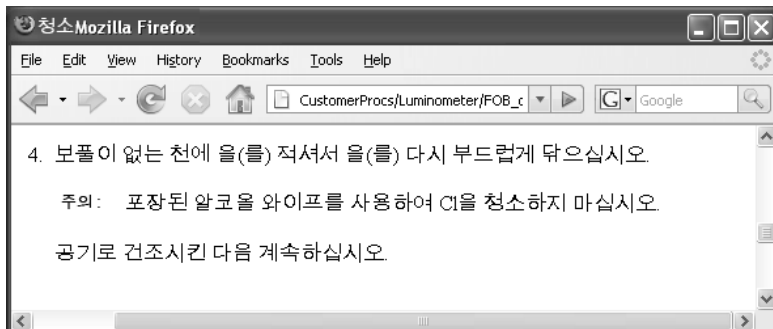
Korean XML:

```
<step>
  <cmd>보풀이 없는 천에 <inline-library-
reference class="- topic/inlinelibraryreference "
href="../../../../Common_XML_files/PartNames/alc
ohol-long.xml"/>을(를) 적셔서 <inline-library-
reference class="- topic/inlinelibraryreference "
href="../../../../Common_XML_files/PartNames/Lu
min_FOB-long.xml"/> 을(를) 다시 부드럽게
닦으십시오.</cmd>
  <info>
    <note type="caution">포장된 알코올
와이프를 사용하여 <inline-library-reference
href="../../../../Common_XML_files/PartNames/Lu
min_FOB-long.xml"/> 을 청소하지 마십시오.</note>
  </info>
```

Each of these XML files can be transformed into a user-friendly format such as a web page. The structure of the web pages will be the same, because the structure of the XML is the same. But the appropriate words in each language will be displayed.



English HTML output from XML (highlighting the inline-library-references in gray)



Korean HTML output from XML

Translation Memory: A Tool for Leveraging Reuse in Translation

Translation memory is “a database that stores segments that have been previously translated. ... The translation memory stores the source text and its corresponding translation in language pairs called “translation units”. ... A translation memory consists of text segments in a source language and their translations into one or more target languages. These segments can be blocks, paragraphs, sentences, or phrases.”

”...Translation memories work best on texts which are highly repetitive, such as technical manuals. They are also helpful for translating incremental changes in a previously translated document, corresponding, for example, to minor changes in a new version of a user manual.”²

Reuse of Translation Memory (TM) within a file or set of files

When translation memory is applied to translations, content that appears repeatedly does not have to be translated over and over as if it contained all new words. Rather, the translation tools recognize the repetition and when identical text is encountered, the translation memory simply reuses the previously translated text. When the text is completely identical, down to

² http://en.wikipedia.org/wiki/Translation_memory

the level of spaces and punctuation, the match is considered “100%”. If there is any deviation between text, then the tools find various degrees of “fuzzy matches”. For example, the sentence

“Open the door, then pull the handle and turn it to the right.”

is not identical with

“Open the door, then pull on the handle, turning it to the right.”

The translation company is paid to decide if these two sentences really mean exactly the same thing, or if each needs a separate translation.

The more consistency there is in the way that text is written, the better the matching and the higher the reuse of the translation memory. If the same sentence is used many times in many files, the translation is just inserted every time a match occurs. This automation achieves greater speed in translation, with higher accuracy, and therefore lower costs.

Even better, over time, as more translation memory is created from a larger body of related content, the translation memory contains many more 100% matches. When translation memory is created for many languages from consistently written and structured content, the savings can become dramatic, as our case study will demonstrate. The cost and time savings in combining reuse and translation memory has also been documented in the Rockley Report. Here is one example:

“Total words published per year 750,000 words

Cost for 100% matches: \$0.07/word

Assumed % of words that would fall under 100% match criteria: 30, 40, 50, 65 [first column]

%	Words eliminated	Dollars saved per language	Total (14 languages)
30%	225,000	\$15,750	\$220,500
40%	300,000	\$21,000	\$294,000
50%	375,000	\$26,250	\$367,500
65%	487,500	\$34,125	\$477,500

The writer also notes that

“...Reuse of your text will also shrink your timelines. The ability to reuse text will allow you to translate only new blocks of content reducing translation turnaround time.”³

³ “Content management systems and translation memory: creating management buy-in,” by Peter Argondizzo, in *The Rockley Bulletin*, vol. 2, issue 4 (copyright, The Rockley Group, Inc.)

A Story of “Extreme Reuse” with XML in Translation

A few years ago, leaders of the technical communications department of a medical device company embarked on an assessment of how to reduce their time and costs to create an important type of content, troubleshooting messages. The product that they supported was large and complex, with multiple software modules and sophisticated hardware. The product was being engineered to integrate and update parts of two existing products into a single, more powerful unit. Thousands of different troubleshooting messages could show up on the product’s touch-screen interface (in HTML format).

The schedule for creating the troubleshooting content was very aggressive, given that there would be thousands of messages generated by the integrated product, and these messages had to be delivered in 17 languages. A rough word count based on the messages for the two existing products was about two million words. The initial estimates also provided a writing and translation schedule that would extend far beyond the integrated product’s launch date. Obviously, the production of the content needed to be streamlined to meet the company’s aggressive delivery schedule.

The technical communications department was already bringing other types of content into XML format from legacy formats, so they had some experience with creating XML content and transforming it into PDFs. They also had a Content Management System (CMS) to provide version control, review and approval workflows, and translation workflows.

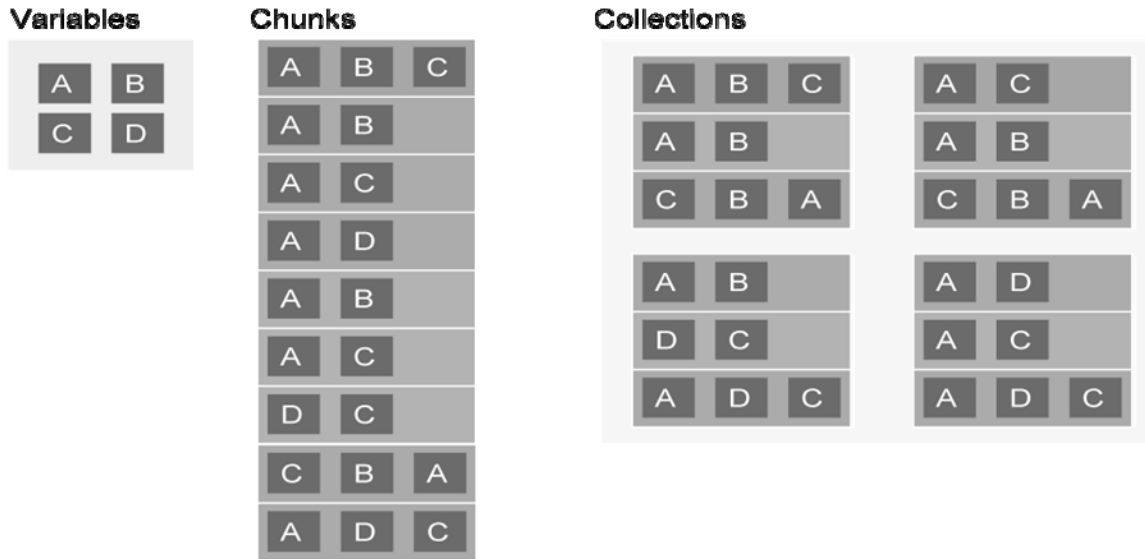
A critical part of their decision-making process was how to maximize the value of their content creation process. What would it mean to “write once, use many times”? How big or small a unit of content could they reuse?

Novatek’s XML specialist worked with the department to gather the requirements for content development, reuse, review and approvals, translation and transformation to the final output formats. The system was developed in prototype and tested iteratively until a 3-tiered modular content development design was achieved.

Multiple levels of content reuse

The content development methodology that the department adapted was innovative and arduous. It required in-depth analysis of the words contained in those thousands of messages, and combination of the minimum essential words and phrases to create the most messages. An important consideration was writing with the goal of translation in mind. Sentences had to be simple and declarative, with unambiguous terms and actions. In particular, verb-subject relationships had to be clear in English, and wording needed to be standardized.

The content model that they developed starts from a vocabulary of nouns and phrases (variables) that occur frequently across a set of troubleshooting messages. These words are used in sentences that are formulated in standard ways. The sentences are used to construct content pieces (“chunks”) for descriptions of the problem reported by the system, followed by pairs of possible causes and actions to take to fix the problem. These sets of troubleshooting messages are bundled into groups according to the part of the product to which they relate.



Modular content development: variables are used to construct parts of chunks, which in turn are used in “mix and match” collections.

The diagram shows how variables can occur in a collection of troubleshooting messages. Some may be used in many messages, while others occur occasionally. In addition, many cause-action pairs might use sentences that are grammatically the same, with different variables, a “fill-in-the-blank” approach to sentence construction.

Typically, a Cause sentence could read “The _____ is obstructed.” The related Action might be a series of steps to perform:

- “1. Open the _____ and remove the _____ from the _____.
2. Blow compressed air into the _____.
3. Replace the _____ and close the _____.
4. Reset the _____ by touching _____ > _____ > _____.” (where the arrows indicate a sequence of touch-screen buttons).

As you see, the Cause text is very generic. Many nouns or phrase could be inserted to indicate what part is obstructed. The Action sentences are as generic as possible; although step 3 might not be used in many procedures, steps 1, 2 and 4 could take a variety of subjects in the blanks.

The level of granularity in developing the content is at the heart of the extreme reuse achieved in this content creation project. It took months of analysis by subject matter experts and an experienced writer to determine the names of parts, buttons, link text, etc. that should be used in the blanks, and to weed out all the variations of the sentences to achieve the most reusable wording. This effort resulted in large cost savings, as we’ll see.

When the sentence blanks were filled in, the content had to be reviewed and approved. XML content was transformed into PDFs that were routed via email and returned with comments. After approval and edits, the XML content was ready for translation.

From the translation vendor's point of view, while it is important that the same nouns are used repeatedly and predictably, a translator needs to see every noun in the context of its use. So the level of translation is generally a sentence or a phrase. As the translators receive more content that follows the same pattern, they find more 100% matches and higher percentages of near-matches (where the translator might only have to replace a different word within a sentence that is otherwise identical to one they have already worked on).

Thanks to the "fill-in-the-blank" sentences and modular structure, the XML content for the troubleshooting codes was so regularized that entire blocks of content could be reused. As the troubleshooting codes poured in for a set of related messages, the reuse continued to expand. The net savings of this reuse and translation technology was that the cost per word dropped from about \$0.22 per word to about \$0.09 per word, ultimately resulting in about \$2,000,000 savings for translating the complete set of troubleshooting messages.

The investment in making all the content so reusable continues to pay off. When new sentences or revision are sent for a troubleshooting message that is already translated, the amount of the overall XML file that has to be retranslated or newly translated is tiny. Typically, over 90% of the text is pulled from translation memory, so the company only pays the translators for the less than 10% translation that they need.

This case, as mentioned, is an extreme example. Most XML reuse is more modest. But when the XML model is well planned for reuse, translation costs savings of 30% or more over non-XML content is not uncommon.

Getting Reuse Beyond Departmental Walls

Once a company has decided to use XML as their preferred format for source content, the concept of reuse will naturally arise. It is fairly easy to decide on using an XML model, whether an established standard like DITA, or a custom one, depending on the task at hand. But the larger-scale reuse opportunities will come from planning out how to reuse content beyond the department where it originates, combining analyses of a company's business processes and its methods of delivering information.

It can be argued that all of the business' content used by customers, internal personnel, partners and resellers is also part of the company's "reuse opportunity". Content may be written

- to support sales and marketing (product brochures, datasheets, catalogs, online order forms)
- to support customer use of products and services (web site FAQs, user manuals)
- to support technical service personnel (call center knowledgebases, training materials, quick references, service manuals).

All of this content is fair game to consider for reuse and potential cost savings.

Creating Content for Reuse

To illustrate the reuse concept, consider what may be present in a typical customer document, such as a Quick Start guide for using a newly-acquired product. Here are some content pieces that we are likely to find:

- product descriptions (features, benefits, physical characteristics, use requirements)
- visuals (photographs of the product from various viewpoints, diagrams, illustrations)
- procedures (how-to steps, instructions for installation)
- warnings, tips and notes
- tables, lists and reference material
- links among content and to other resources (“see also”).

Typically, accompanying these are business marketing and branding content (logos, slogans, copyrights, business contact information, etc.).

Businesses commonly will apply their branding across all types of content, so the reuse of content is understood in this context (making sure that every piece of content is recognizable as belonging to the company). What is less understood is the reuse of pieces of content in multiple contexts. How many of those content pieces need to be different due to their context? How many could be identical, if everyone could agree on their wording and structure?

Some types of content, such as admonitions (Warnings, Danger, Caution, Important) and boilerplate (disclaimers, copyrights, corporate logos) should be cases of identical reuse to comply with regulations or avoid litigation, but that is hard to achieve because the materials may all be in different document formats. Creating the content in XML, translating the stored XML as the “content of record”, and transforming it into the output format as needed, can reduce chances of errors and omissions.

For example, a Warning that appears in the Quick Reference is likely to be used in the service manual and the call center’s knowledgebase as well. It will improve the company’s return on the effort of writing that Warning to reuse it exactly the same in all the places possible. If that warning is translated, then keeping it identical in all contexts will return its value in translation memory also.

Logistics, learning and leadership for effective reuse policies

Gaining the cooperation to coordinate writing that can be reused in documents for different departments is not easy. There can be surprising differences in the words used for general text (synonyms such as “lid”, “hatch” and “cover”, “select” and “click”, “type” and “enter”). So one of the first tasks must be forging agreements on terminology. There can be writing style issues such as what voice or tense is used. There are also logistical issues such as managing the content (“Who “owns” it?” “Who needs to be consulted in the event of a change request?” “Can there be multiple versions in use for different product releases?”). There are issues of cost-sharing for the XML development, storage and translation.

Teaching writers to use XML content-creation tools, and to write in topics or other types of “chunks” that can be used to build many different deliverables, is also a considerable investment. It may take several months for the writers to get comfortable with collaborative writing and using their new XML tools.

XML reuse is ideally championed by a senior executive who can bring the interested parties together and forge the agreements needed to achieve cross-departmental success. A business case can’t always be made if the goal is only to get more consistent and flexible content. But if the goal is to gain better cost control across global content use, leveraging XML content and translation memory together will achieve efficiencies that neither can achieve alone.