

How to Leverage the Maximum Potential of XML for Localization  
Andrzej Zydroń, CTO, xml-Intl Ltd. & Member, OSCAR Steering Committee

XML has become one of the defining technologies that is reshaping the face of both computing and publishing. It is helping to drive down costs and to dramatically increase interoperability between diverse computer systems.

In this article, Andrzej Zydroń, CTO of xml-Intl Ltd. and OSCAR Steering Committee Member, explains how DITA and xml:tm fit into the equation and how they will take us beyond the existing XML-based localization standards that have only concentrated on the exchange of information. DITA is a flexible, topic-based architecture that provides a comprehensive model for authoring, producing and delivering of technical documentation. xml:tm is a rather radical departure from existing standards that introduces the concept of **text memory**, seamlessly integrated into XML documents.



**Andrzej Zydroń**

Editor's Note: If you, or any of your colleagues, are working to resolve XML-related challenges, don't miss the opportunity to gain insights from one of the industry's leading experts on standards related to localization and XML. You will have three opportunities to "pick Zydroń's brain" during the [LISA Global Strategy Summit](#) in Boston, May 23-27:

***(1) Open Standards: How XML and DITA Work Managing Multiple-Language Content Creation and Distribution***

***(2) XML and Localization***

***(3) DITA: An Introduction to the XML-based Open Standard for Managing Multiple-Language Content and Why It Is Becoming Important for Global Business***

We are rapidly moving towards an XML-dominated world when it comes to publishing and localization. The impact of XML has to date been mainly at the system level. Nevertheless, all of the major publishing tools have moved towards full support for XML, and companies that have adopted XML-based publishing have seen significant cost savings compared with older proprietary systems.

From the point of view of localization, XML offers many advantages:

- A well-defined and rigorous syntax that is backed up by a rich tool set that allows documents to be validated and proven.
- A well-defined character encoding system.
- The separation of form and content, which allows both multi-target publishing (PDF, Postscript, WAP, HTML, XHTML, online help) from one source, thus greatly

simplifying the localization process.

- XML-compliant tools can exchange documents easily without ambiguity.

The localization industry has also enthusiastically adopted XML as the basis for exchange standards such as the excellent ones sanctioned by LISA through [OSCAR](#) (Open Standards for Container/Content Allowing Re-use):

- [TMX](#) (Translation Memory eXchange)
- [TBX](#) (TermBase Exchange)
- [SRX](#) (Segmentation Rules eXchange) standards
- [GMX](#) (GILT Metrics eXchange) set of proposed standards (Volume, Complexity and Quality)

*Editor's Note: For background information on GMX, please read [GILT Metrics – Slaying the Word Count Dragon](#).*

[OASIS](#) has also contributed to XML-based standards for localization through [XLIFF](#) (XML Localization Interchange File Format) and TransWS (Translation Web Services).

*Editor's Note: For more information on TWS, please read [Web Services for Translation](#).*

All of the above standards and proposed standards have dealt with the *exchange* of data using XML. However, there are other ways that XML also assist in publishing and localization. Two examples are DITA and xml:tm.

## DITA

DITA represents a very intelligent and well thought out approach to technical documentation publishing.

The [Darwin Information Typing Architecture](#) is a proposed OASIS standard. It provides a comprehensive architecture for the authoring, production and delivery of technical documentation. DITA was originally developed within IBM and then donated to OASIS.

The essence of DITA is the concept of topic-based publication construction and development, which allows the modular reuse of specific sections. Each section is authored independently, and then each publication is constructed from the section modules. This means that individual sections need only be authored and translated once, and may be reused many times over in different publications.

DITA represents a very intelligent and well thought out approach to the process of publishing technical documentation. At the core of DITA is the concept of topic. A topic is a unit of information that describes a single task, concept or reference item. DITA uses an object-oriented approach to the concept of topics, encompassing the standard object-oriented characteristics of *polymorphism*, *encapsulation* and *message passing*. *Polymorphism* is the ability of an object to take multiple forms. In the case of DITA, a topic document can be used in multiple and different publications. *Encapsulation* means that you do not need to know the details of what is contained in a topic document. The details are self-contained, and the whole

document can be treated as a single object from the point of view of publishing.

The main features of DITA are:

- a topic-centric level of granularity
- the substantial reuse of existing assets
- a specialization at the topic and domain levels
- processing based on meta data property
- the leveraging of existing popular element names and attributes from XHTML

For a comprehensive introduction to DITA, please refer to [the IBM DITA information page](#).

I predict a very good future for DITA because it represents a very well thought out and flexible architecture for content creation and publishing. From the localization point of view, it means that once you have translated a topic into a given target language, then it can be reused time and time again, as long as the source language content has not been modified.

## **xml:tm**

xml:tm is a rather radical departure from existing standards.

xml:tm is a rather radical departure from existing standards. Whereas, existing XML-based localization standards have all concentrated on the exchange of information, and DITA has concentrated on a flexible topic-based architecture, xml:tm introduces the concept of text memory seamlessly integrated into XML documents. It is designed to work closely with and to complement DITA, along with other XML-based exchange standards.

What is *text memory*? Quite simply, it is the process of allocating unique identifiers to each text unit in an XML document. A text unit is either the text content of an XML element, or the subdivision thereof into individual sentences. Text memory comprises two distinct concepts:

- Author memory
- Translation memory

Both are intrinsically linked together. xml:tm uses standard XML namespace notation as an overlay onto XML documents.

*Editor's Note: To review the code that will produce the composed page below, please refer to [xml:tm - Using XML Technology to Reduce the Cost of Authoring and Translation](#).*

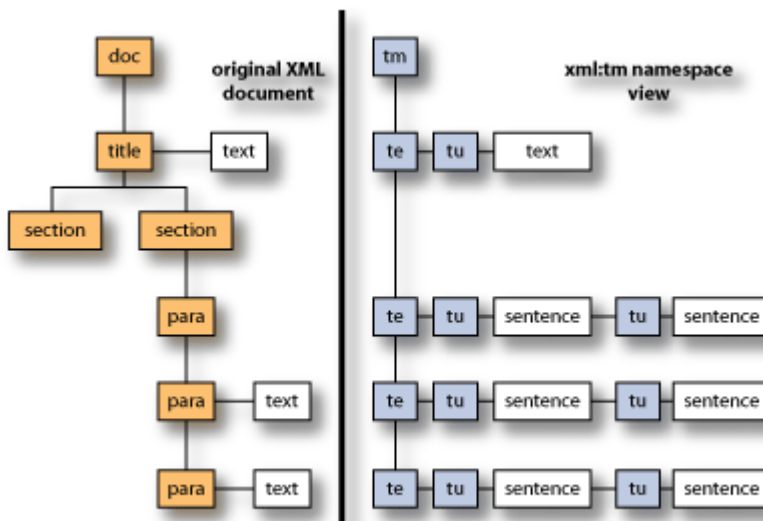
## xml:tm

Xml:tm is a revolutionary technology for dealing with the problems of translation memory for XML documents by using XML techniques to embed memory directly into the XML documents themselves. It makes extensive use of XML namespace.

The “tm” stands for “text memory”. There are two aspects to text memory:

1. Author memory
2. Translation memory

Interestingly, xml:tm provides an alternate view of the original document that is flat and text-oriented, as shown in the following diagram:



## Author Memory in xml:tm

Author memory is maintained during the authoring process with xml:tm. As a document (in DITA terms, a *topic*) goes through its authoring cycle, the unique identifiers are maintained by comparing the document before and after authoring.

What are the benefits of using xml:tm for authors? Firstly, you have a detailed record of your authoring process. You also have all of your sentences segmented for you. You can then use this information to store all sentences in a phrase reuse database. This allows the creation of a phrase reuse system for authoring. If a sentence has been authored and then translated, encouraging reuse allows for a much higher percentage of leveraged memory in the future. Recent internal studies of automotive manuals have shown that up to 80% of a repair manual

can be created from reused phrases.

When first learning about xml:tm, people often think that the presence of the xml:tm namespace will add considerable clutter to the document in terms of authoring, etc. In fact, the presence of the xml:tm namespace is totally unnecessary in the document being authored. The usual technique is to strip out the namespace when reviewing the file for authoring. This is done with two lines of [XSLT](#). On return, the freshly authored document is segmented for xml:tm, and the differences are worked out from the document in the repository. The identifiers are then updated in the new document, which is then stored. In this way, the authoring software does not have any contact with the xml:tm namespace.

## Translation Memory xml:tm

No proofing is necessary.

Translation memory in xml:tm is radically different from the traditional concept of translation memory. 'Traditional' translation memory resides in an external database and is used for leveraged matching during the translation process. However, in xml:tm, the memories are embedded in the previously translated target version of the equivalent source document. The first time that the source document is sent to translation, the extraction is done into XLIFF using the xml:tm namespace. Therefore, there is no need to segment the sentences for translation, since it has already been done for you by xml:tm. In fact, you can extract straight from xml:tm into XLIFF using a simple XSLT transformation. When the translated sentences are merged to create the target version of the document, the latter carries all of the identifiers at the sentence level, as does the original source version. In other words, you will have two identical documents in terms of their XML structure and xml:tm identifiers.

Subsequently, when the source document is revised, those sentences that have not changed will have the same identifiers as before, so you can guarantee that the translation will not have changed. This is the concept of *perfect matching*, as opposed to *leveraged matching*, and it is one of the key benefits of xml:tm. With leveraged matching, a translator must still proof the matching to ensure that it is correct because no assumptions can be made that relate to the context of the match.

The owner, not the service provider, is in total control with xml:tm.

Of course, xml:tm can also be used very effectively to populate traditional leveraged translation memory databases. The other major benefit of xml:tm appears in workflow, since it is ideally suited to completely automating the process of translation memory matching and extraction within an XML-based workflow. It allows the content creator to own and control the translation memories associated with his/her documents and to fully automate the translation process. There is no need for specialist project management or translation memory specialists. The owner is not reliant on a translation supplier to hold translation memories and decide how to use them. In other words, the owner is in total control when using xml:tm.

# xml:tm and Other XML Standards

xml:tm is designed to work directly with other XML based standards:

- [SRX](#) – this is mandated for xml:tm segmentation.
- [GMX/V](#) – all xml:tm-based word and character counts should use GMX/V
- [XLIFF](#) – xml:tm is designed to work directly with XLIFF. xml:tm effectively pre-digests documents for XLIFF so that you can create XLIFF documents by means of an XSLT transformation.
- [TMX](#) – xml:tm simplifies the creation of TMX documents based on xml:tm source and target documents.
- [DITA](#) – xml:tm is a ‘perfect match’ for DITA, allowing further reuse down to the sentence level, as well as automating the translation matching process and allowing document creators to leverage greater value from their documents.

A full and detailed technical explanation of the inner workings of xml:tm can be found in an article about xml:tm published on the influential xml.com web site:

<http://www.xml.com/pub/a/2004/01/07/xmltm.html>.

## xml:tm – The Benefits

The main benefits of xml:tm are as follows:

- Infinitely scalable architecture
- Seamless integration with any XML document
- Automation of the translation process
- Automatic embedding of translation memories into the actual XML documents
- The concept of perfect matching, as opposed to leveraged matching. This means that no proofing is required.
- Total integration and interoperability with all existing XML-based localization and document architecture standards

## xml:tm and LISA

xml:tm was created and is maintained by [xml-intl](#), and the full specification is available [here](#). It has been offered to the LISA OSCAR steering committee for consideration as a LISA OSCAR standard.