Making Effective Use of XML for Publishing

Introduction

XML is now acknowledged as the best format for authoring technical documentation. Its wide support, extensible nature, separation of form and content, and the ability to publish in a wide variety of output formats such as PDF, HTML or RTF make it a natural choice. In addition the costs associated with implementing an XML publishing solution have now come down significantly. Nevertheless there are some clear dos and don’ts when authoring XML – these are detailed in the following paper: Coping with Babel.

XML, thanks to its extensible nature and rigorous syntax, has also spawned many standards that allow the exchange of information between different systems and organizations, as well as new ways of organizing, transforming and reusing existing assets. For publishing and translation these now form a new way of using and exploiting existing documentation assets known as Open Architecture for XML Authoring and Localization (OAXAL).

Open Architecture for XML Authoring and Localization (OAXAL)

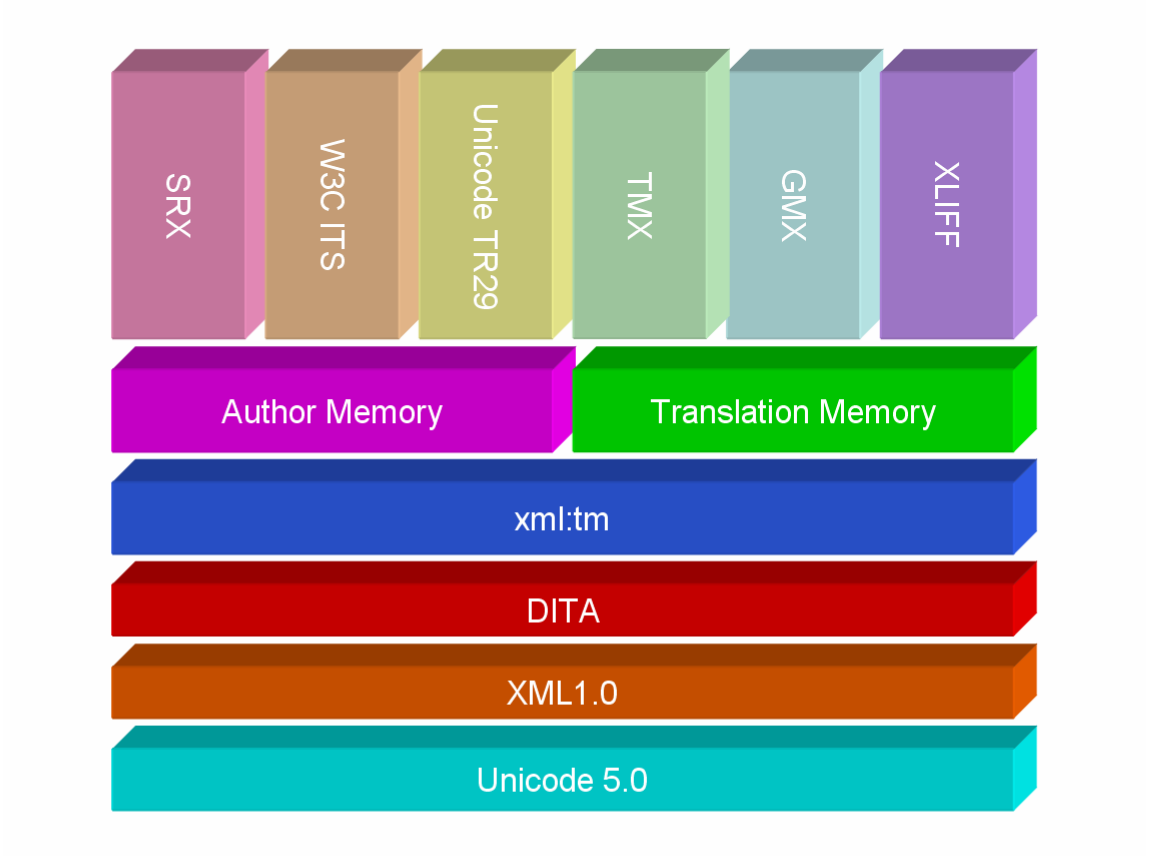
OAXAL takes advantage of the arrival of some core XML related standards:

1. DITA - Darwin Information Typing Architecture from OASIS
2. xml:tm- XML based text memory from LISA OSCAR

DITA is a very well thought out way of introducing object oriented concepts into document construction. It introduces the concepts of reuse and granularity into publishing within a very well thought our XML vocabulary. It is having a big impact on the document publishing industry.

xml:tm is also a pivotal standard that provides a unified environment within which other localization standards can be meaningfully integrated thus providing a complete environment for OAXAL. OAXAL allows system builders to create an elegant and integrated environment for document creation and localization.

Open Architecture For XML Authoring and Localization



W3C ITS is an XML vocabulary that defines the rules for translatability for a given XML document type. It states which elements have translatable text, which elements are ‘inline’ and thus do not cause segment breaks, which form subflows and which attributes are translatable.

Unicode TR29 is part of the main Unicode standard that defines word and sentence boundaries.

SRX (Segmentation Rules eXchange) is an XML vocabulary for defining segmentation rules for a given language.

GMX (Global Information Management Metrics Exchange) is a new LISA OSCAR standard for word and character counts, as well as a vocabulary for exchanging metrics information. It is part of a three part standard that will also tackle complexity and quality.

XLIFF (XML Localization Interchange File Format) is an OASIS standard for exchanging localization data in an XML vocabulary.

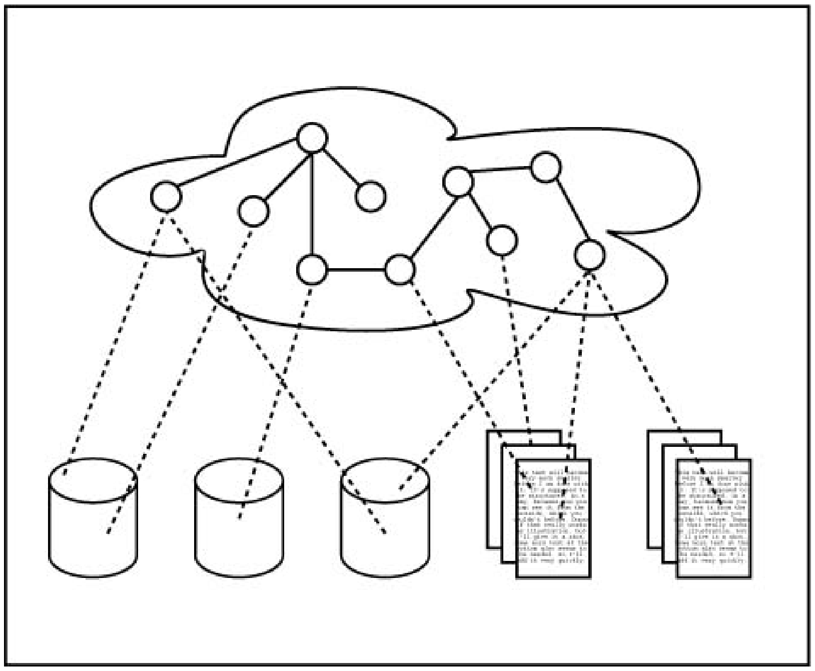
TMX (Translation Memory eXchange) is a LSIA OSCAR standard for exchanging translation memories.

DITA

The OASIS DITA (Darwin Information Typing Architecture) Standard has certainly raised the profile of XML in the technical authoring field. It has done so for two reasons:

1. It is a very intelligent and well though out approach to the authoring and publication of any technical manual.
2. It substantially reduces the costs associated with introducing a component based publishing system.

The original architects of DITA looked at the most effective way of writing technical documentation and came to the conclusion that the Topic Map approach was the best way of achieving this. Rather than writing a publication as a monolithic set of chapters they found that deconstructing a publication into a set of topics was a far better way of doing this. It meant that authors could work independently on their own topics without impeding one another. It also meant that topics could be reused across many different publications. The core messages of DITA are granularity and reuse.

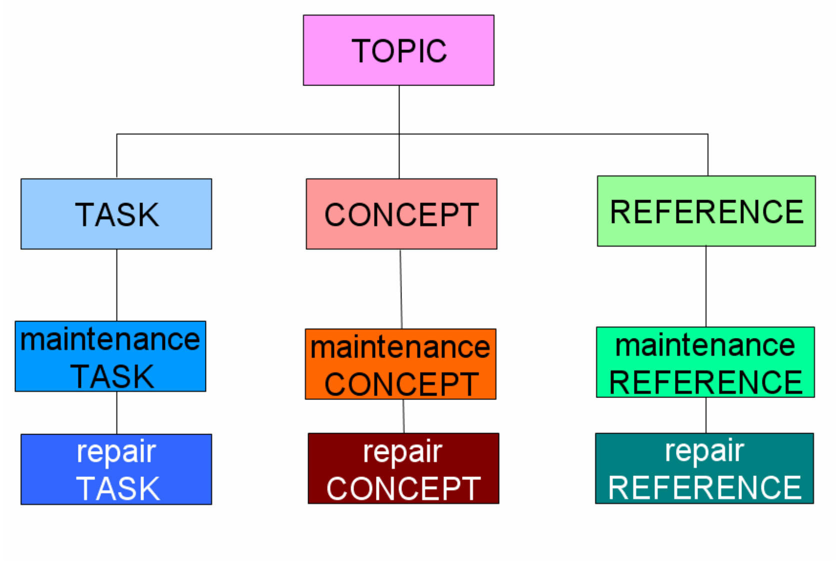


The topic concept provides DITA with a built in component model which can we readily understood and implemented: write once, translate once, reuse many times. There are many analogies to this approach in the automotive industry where the same components are reused across many model ranges. This substantially reduces costs and increases the availability of spare parts. Having a DITA document for discrete operations, such as the removal of a cylinder head for a given engine type, means that this procedure can be reused across all of the publications relating to models that share that engine.

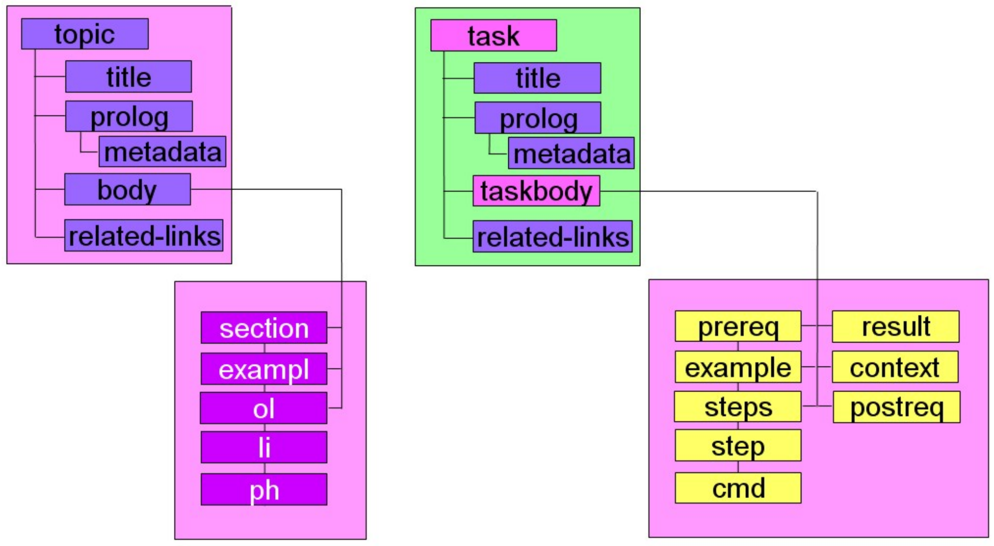
DITA also comes with a built in concept of extensibility. The original DITA architects realized that topics can have different formats, and not all topics are equal, so they worked out an extension mechanism that allows topics to be specialized. The standard DITA 'topic' is available in the form of the three most common implementations:

1. Reference
2. Task
3. Concept

These can then be in turn specialized further, so 'Task' can have 'Maintenance Task' and 'Repair Task', 'Disassembly Task' and 'Assembly Task' specializations using the simple DITA extension mechanism.

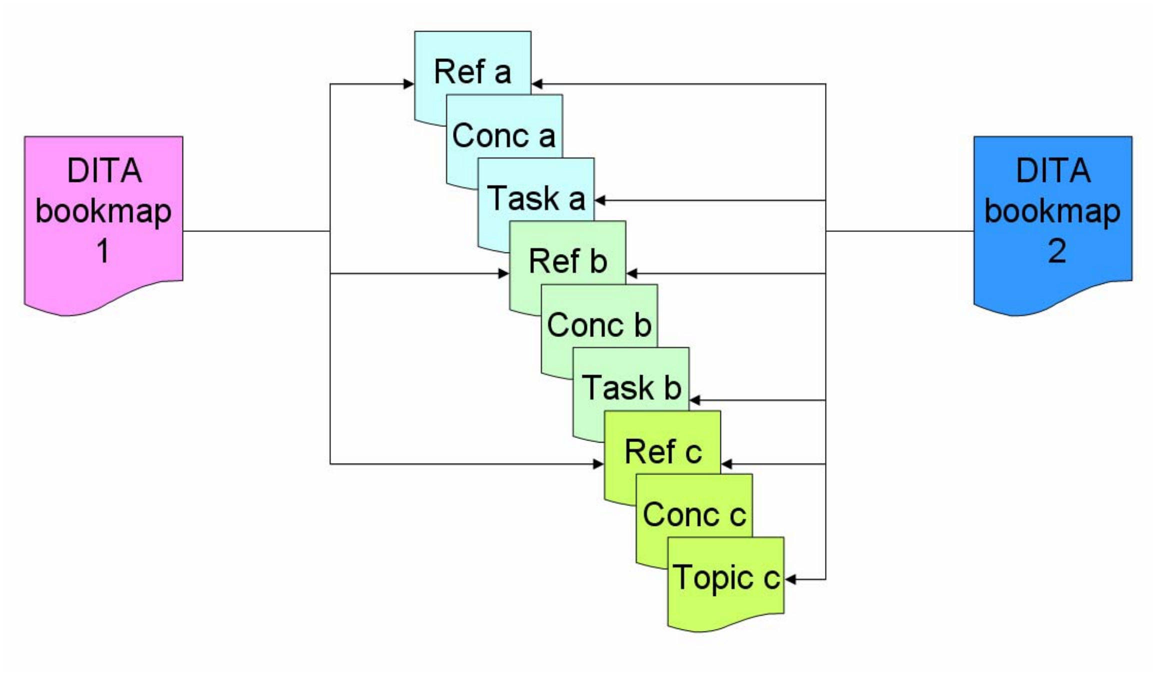


Elements in the new specializations can share typographical characteristics of their forebears even though they may be called by a different name.



By providing a ready built toolkit for the construction of technical documentation, DITA substantially reduces the cost of implementation. DITA comes with a prepackaged set of tools to construct books and manuals as well as complex web pages structures for free.

In order to put together a publication from individual topics DITA uses the 'bookmap' concept. A 'bookmap' allows the publisher to decide which topics go into a given publication.



DITA also has some very powerful mechanisms for conditional processing depending on environmental settings for such items as model name etc. The following example shows how conditional processing can be used:

```
<p audience="administrator">Set the configuration
  options:
 <ul>
  <li product="extendedprod">Set foo to bar</li>
  <li product="basicprod extendedprod">Set your
  blink rate</li>
  <li>Do some other stuff</li>
  <li platform="unix">Do a special thing on Unix
  systems</li>
 </ul>
</p>
```

DITA provides allows conditional processing depending on the attribute values for given elements. In the above example the `'p'` element will only be published if the environmental variable `‘audience’` has the value `‘administrator’`. Likewise the `‘li’` elements will only be published if the relevant `‘product’` and `‘platform’` attributes have the required settings.

One of the main benefits of DITA from the standardization point of view is that most of the work regarding the XML Schema construction has already been done. Previously introducing a custom built XML Schema would entail lots of work and the associated costs involved with hiring a consultancy or specialist. DITA provides a ready made pre-built generic XML environment for authoring and publication. The full DITA Open Source Toolkit is available online which provides transformations for HTML, PDF and RTF output as well as many other utilities and examples. The other benefit of standardization is that there is extensive support now available from XML editors and content management systems for DITA making implementation simpler and cheaper. In addition there has been a rapid build up of Documentation, Training and Consultancy resources for DITA. All of these make for easier and cheaper implementation.

If your existing documentation is already in a ‘topic’ or ‘task’ type format then moving to DITA should be relatively simple as long as there is direct mapping between the existing elements used and DITA elements. It may even be possible to develop a specialization of the existing DITA into the incumbent element names and attributes. Another factor that makes the move to DITA easier is if your existing documents use the CALS table model which DITA supports. Conversion between the J2008 table model and CALS is relatively simple.

DITA is not only for technical manuals. It can be easily adapted to other subject matter such as eLearning, warranty or service bulletin publications.

As with all good things, there are some aspects that need to be considered carefully. DITA does have some potential issues to bear in mind:

1. Element nesting – DITA has unfortunately followed the HTML principle that nearly every element can appear in another element. This can cause horrible problems with publishing and translation, where segmentation and typography can go badly awry.
2. Many DITA elements can occur inline with text which can cause substantial problems for both composition and translation if not restricted.
3. The linking mechanism for conditional text can cause problems with localization.
4. Topic based publishing requires the use of a content management system, specially if you are also translating into other languages otherwise control of versions and synchronization with different language version becomes very difficult.

XML based Text Memory – xml:tm

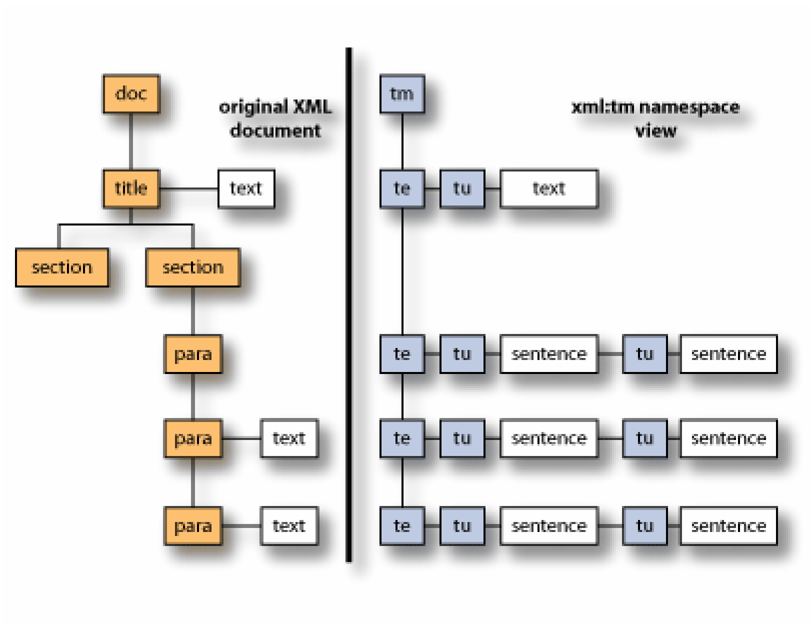
xml:tm is a new proposed LISA standard that is a directly compatible with DITA. It takes the DITA principle of reuse and applies it at the sentence level. Integration with DITA is seamless.

Whereas traditional translation memory systems have only concentrated on the translation aspect of the document lifecycle, xml:tm goes deeper into the lifecycle process establishing the concept of "text memory". Each sentence (text unit) in the document is given a unique identifier. This identifier remains immutable for the life of the document. xml:tm uses the XML namespace mechanism to achieve this.

Xml:tm comprises two core concepts:

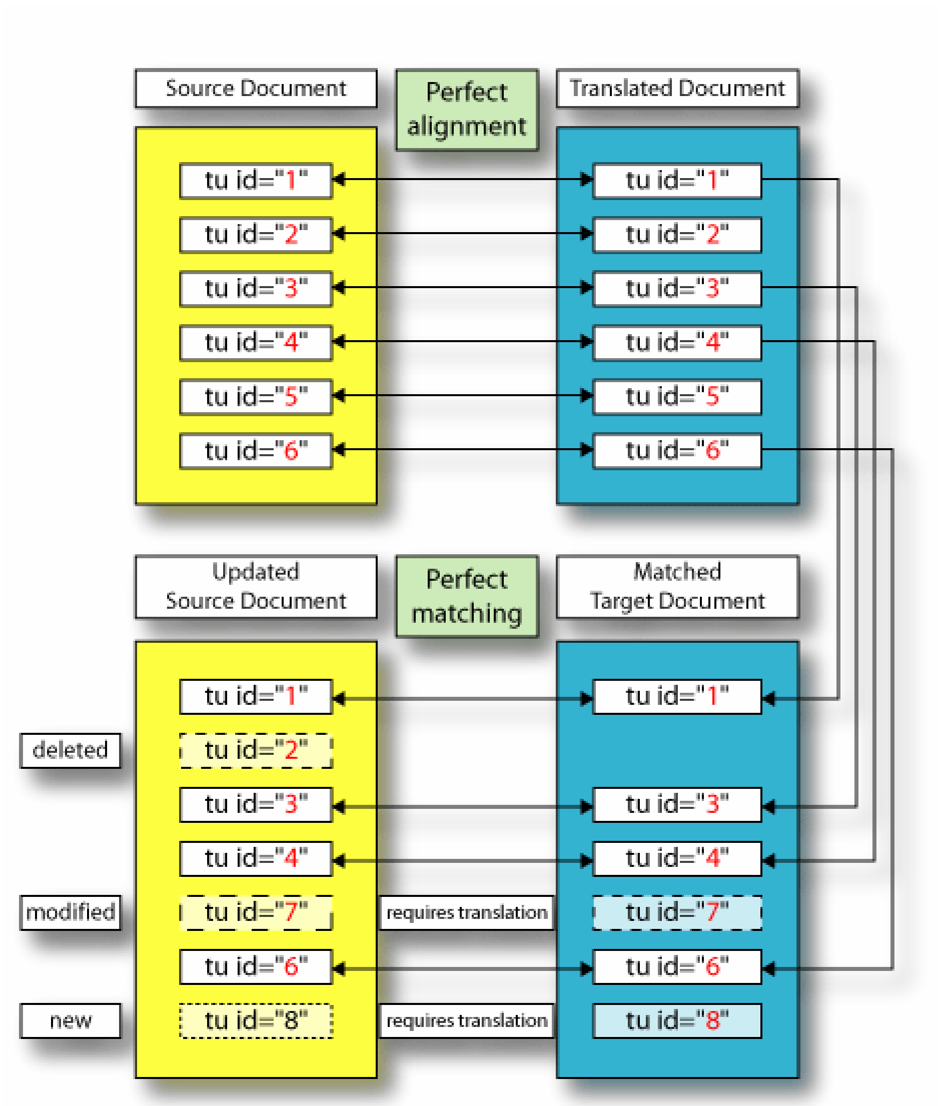
1. Author memory
2. Translation memory

The following diagram shows how the xml:tm namespace coexists within an XML document:



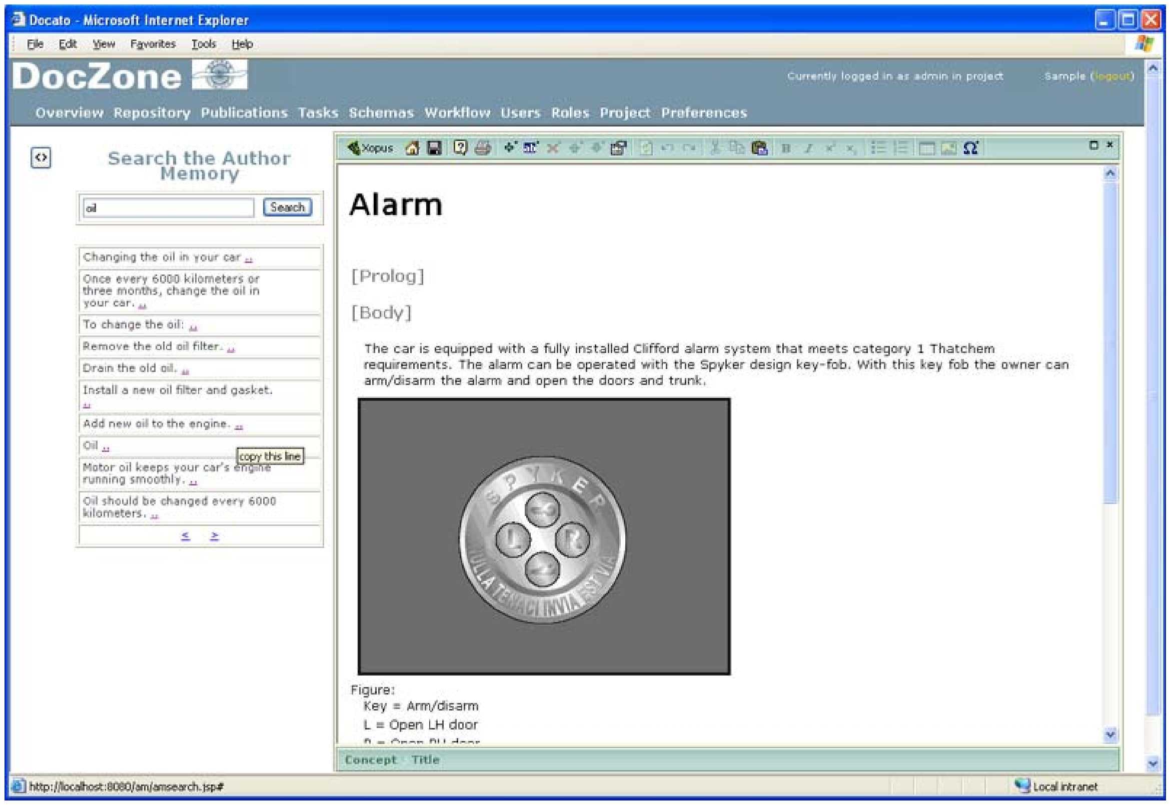
XML namespace is used to map a text memory view onto a document. This process is called segmentation. The text memory works at the sentence level of granularity - the text unit. Each individual xml:tm text unit is allocated a unique identifier. This unique identifier is immutable for the life of the document. As a document goes through its life cycle the unique identifiers are maintained and new ones are allocated as required. This aspect of text memory is called author memory. It can be used to build author memory systems which can be used to simplify and improve the consistency of authoring. A detailed technical article about xml:tm has been published on O'Reilly's xml.com web site.

The use of xml:tm greatly improves upon the traditional translation route. Each text unit in a document has a unique identifier. When the document is translated the target version of the file has the same identifiers. The source and target documents are therefore perfectly aligned at the text unit level. The following diagram shows how perfect matching is achieved:



In xml:tm terms this type of perfect matching is known as ‘in context exact’ (ICE) matching. The xml:tm concept has been successfully proven within very large scale automotive publishing applications.

For authoring xml:tm represents a systematic way of identifying and storing all previously authored sentences. This repository can then be appropriately indexed and serve as input to authoring tools, to encourage authors to reuse existing sentences where appropriate rather than coming up with different ways of saying the same thing.


Docato - Microsoft Internet Explorer
File Edit View Favorites Tools Help
DocZone
Currently logged in as admin in project Sample (logout)
Overview Repository Publications Tasks Schemas Workflow Users Roles Project Preferences
Search the Author Memory
oil
Search
Xopus
copy this line
Concept Title
http://localhost:8080/am/amsearch.jsp#
Local intranet

Changing the oil in your car

Once every 6000 kilometers or three months, change the oil in your car.

To change the oil:

Remove the old oil filter.

Drain the old oil.

Install a new oil filter and gasket.

Add new oil to the engine.

Oil

Motor oil keeps your car's engine running smoothly.

Oil should be changed every 6000 kilometers.

Alarm

[Prolog]

[Body]

The car is equipped with a fully installed Clifford alarm system that meets category 1 Thatchem requirements. The alarm can be operated with the Spyker design key-fob. With this key fob the owner can arm/disarm the alarm and open the doors and trunk.

Figure:
Key = Arm/disarm
L = Open LH door

The author can enter key words and search the author memory for appropriate sentences. If a sentence has been already used, then it is highly likely that it has also been translated, in which case a leveraged translation match will be found further reducing translation costs.

xml:tm provides a pivotal role within OAXAL allowing all the other related standards to interoperate within one elegant architecture.