TEXOLUTION GmbH & Co. KG
Freiheitstraße 34
D-75045 Walzbachtal

An Open Source DITA CMS for Technical Authoring (and other purposes)

Dr.-Ing. Boris Horner

Abstract

With growing use of DITA in the past few years, an increasing number of software tools support DITA, including editors, CMS / ECMS systems and publishing tools. A major part of these have commercial licenses, but there are also open source products covering the entire tool chain.

Beside the obvious advantage of open source software being available without license fees and thus enabling inexpensive projects, there are some other facts making open source software attractive in business use. To determine how and when to use open source software, it is important to understand open source licenses.

This article deals with several open source software products that are useful for technical documentation and that can be composed to a toolchain for production of technical documents, based on an ECMS.

1 Introduction

1.1 Why open source?

The most obvious argument for using open source software is: there are no license fees, so a customized application costs less. This is, of course, just a general trend. It always depends on what products are actually compared with each other, and what the exact requirements are. Let's assume though that solutions based on open source products are in the average cheaper than solutions based on commercial software.

Beyond those differences with respect to license fees, there are other differences between open source and commercial software, that make open source software attractive.

Customers have an interest in sustainable service and support, and it is highly undesirable if a system supplier suddenly vanishes from the market, or decides to discontinue the product used. When buying mission critical software, large enterprises often demand escrow agreements, to ensure that the source code of the software is available to them in cases like the ones mentioned above. This way, the customer can find another service provider for support and continuation of the system.

Software escrow agreements become unnecessary when using open source software. The source code is available not only under certain circumstances, but always and guaranteed. This way, the danger of vendor lock-in is reduced.

In some dynamic environments, it makes sense to be able to quickly create sites, seats or entire repositories. With a commercial license model, in many cases, additional licenses need to be acquired and managed. This is bureaucratic overhead and contributes to the TCO.

1.2 Licenses

Open source software is software created by companies or highly skilled individuals and available without license fees, including the source code – so you can use the code in whatever way you want, provided you do not remove the authors name from the documentation. While this is often true, there are important differences between various open source licenses.

Some licenses as the GPL disallow linking the software to any module bearing an incompatible license, particularly a commercial one. Depending on the type of software, this might be a problem or not. Using a GPL library within a custom module can put the entire module under GPL license. This is known as the viral aspect of the GPL. This means, the module has to be open source as well, which is undesirable in almost all cases. Customizations are project specific and typically contain some information or algorithms not meant for the public.

There are other open source licenses like LGPL or Apache license that are more „friendly“ towards use in commercial systems, and allow linking with commercial modules without conflict. When composing a system from open source components, products with such licenses are to be preferred, particularly when customizations are required. In any case, the licenses should be carefully considered when choosing the products.

1.3 Scenarios

Let's start with a definition of the scope the open source toolchain should cover. To get an idea about the breadth of possible scopes, it is useful to look at the two extremes how technical documentation is nowadays created.

1.4 The minimal scenario

In many small and even some mid-sized companies, the toolchain is quite simple. In the worst case, technical documentation is produced using word processor software like MS Word and the resulting files are dumped into the file system. The manager keeps track of it all using spreadsheets like MS Excel. In some cases, we will find Adobe FrameMaker instead of the word processor.

Technical authors in such an environment are rarely full-time, professional technical writers, but rather development, production or service engineers who create the technical documentation as a part of their job.

Translation Memory systems are sometimes used by external Language Service Providers, but often translations into common languages are done in-house, for example by persons on foreign sites (who are not professional translators, and who do not use a TMS).

Using Content Management Systems for organizing the data is not very common.

1.5 The full scale scenario

The other extreme is the scenario of a large company producing technical products and thus producing technical documentation.

The toolchain discussed in the previous scenario might be manageable in small companies with few different products or variants of products, or a limited set of target languages.

Once the diversity or number of different configurations, markets or languages exceeds a certain limit, companies become aware of the need for modularization and re-use, and also systematic organization of the data (including versions, relations, lifecycles management, workflow automation and so on).

In a typical large enterprise, we therefore have a content management system with TMS[1], a modular data model and single source publishing to various formats, including proprietary interfaces of retrieval systems. As the data format for modules, often Adobe FrameMaker is used, some (legacy) solutions are SGML based, but an increasing number of data models are based on XML (but not necessarily DITA).

1.6 Scope definition

The full scale scenario obviously defines the functionality an open source solution has to cover, otherwise it will be unattractive for any company beyond a certain size, while limiting growth for smaller companies at the same time.

Let's therefore summarize what we need:

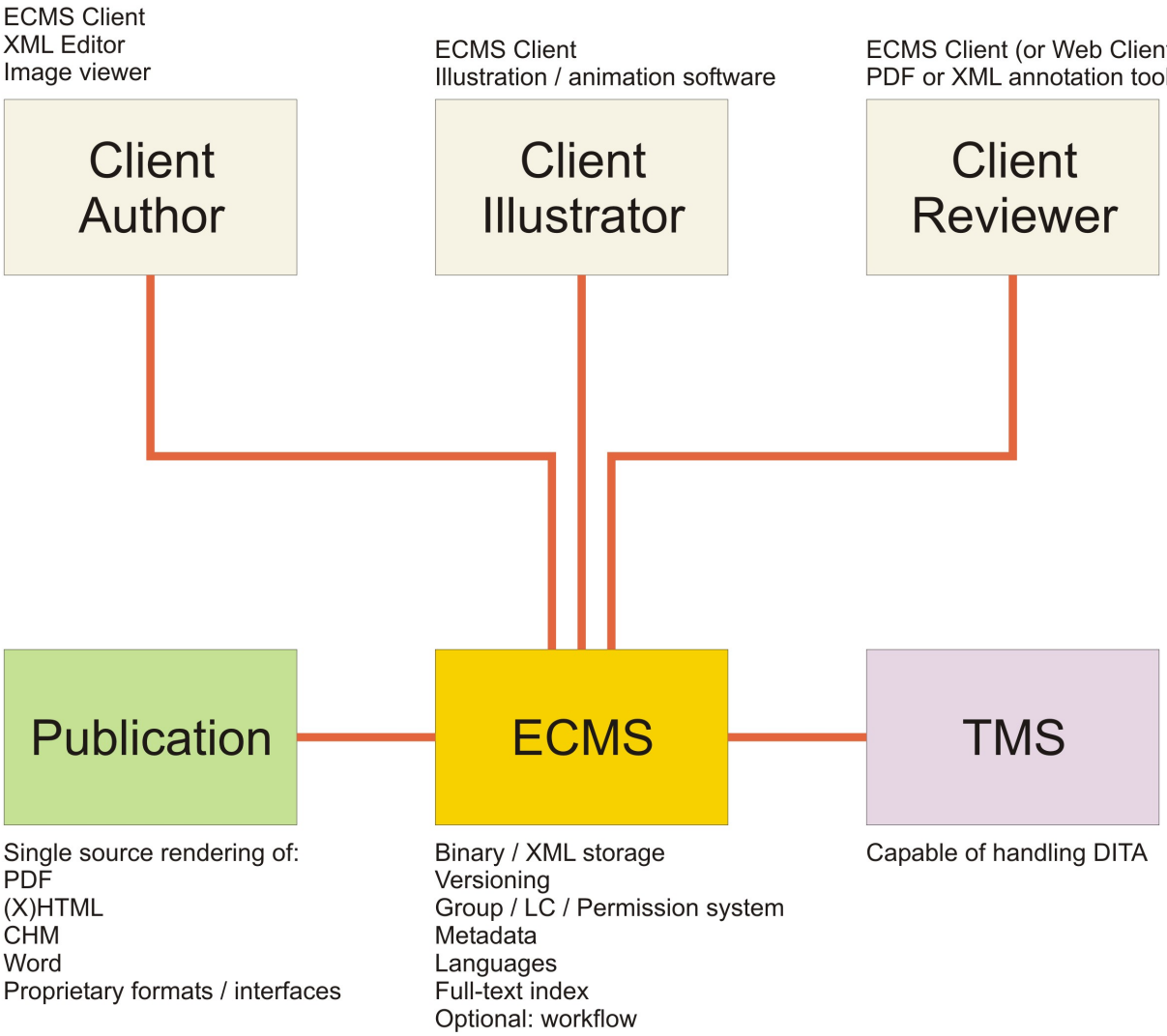
- An ECMS (Enterprise Content Management System)[2] with these features:
  - Versioning, allowing references to specific versions of objects.
  - A role / group / permission / lifecycle system.
  - Full-text indexing and search.
  - Means for managing languages and variants.
  - Flexible ways of assigning metadata to objects.
  - Workflow automation (optional).
  - Client (desktop or web-based) providing a GUI to interact with the ECMS.
  - Programming and customization interfaces allowing the modification of the system to fit the specific needs, and to create interfaces to other systems (ERP, PPS, ...).
- An editor for the data format used, in this case DITA. This is typically a native XML editor, but there are web-based solutions and others based on Adobe FrameMaker or Microsoft Word.
- Editors for other data formats (illustrations, CAD, video etc.).
- Translation management capabilities allowing use of or integration with a TMS.
- Publishing modules to create the target formats that are required.

For better overview, the following diagram shows the composition of the system components.

Further clients for yet different user roles can be easily imagined, here are just two further examples:

1 TMS = Translation Memory System.

2 Let's not argue about this term. There are web CMS based solutions as well as other systems that are rather called DMS or RDBMS-based solutions by their manufacturers. I use this term representing the variety of possible ways of storing structured and binary content.



- translators with TMS client software
- subject matter experts who contribute background information for authoring or review
- customer or end user feedback management (for reporting errors in the documentation, for example)

The software should therefore be flexible to the degree that different user types can have different client capabilities.

2 Open source building blocks

2.1 A word in advance

In this section, some open source products are given as examples for building blocks of the type of system drafted before.

I need to stress though that the product examples are not intended to be a complete list of all open source software products available. They are a small subjective choice. Not showing up on the list is definitely not meant as negative judgement.

2.2 Platform

For completeness, the platform should be considered. Some server software runs on Windows only, some on Linux only, and some software runs in both environments (like many Java-based systems). The platform chosen is often a decision based on the overall IT strategy. The same is true for supported RDBMS.

2.3 ECMS

There are many commercial CMS for technical documentation, and they often support DITA. There are, however, not so many open source CMS platforms available.

I am not aware of an open source system that covers the whole technical documentation process in its entirety. There are several commercial products with such functionality.

An open source based system will therefore be a composition of several components, based on an ECMS. Such systems are not specifically designed for technical documentation purposes, but are rather general purpose management systems for assets of all sort, technical documents being just a part of them. ECMS are flexible enough to be customizable for all technical documentation tasks.

Commercial systems of that category are Documentum or LiveLink.

Two open source systems with ECMS functionality are

- Cinnamon (http://www.cinnamon-cms.de/)
- Alfresco (http://www.alfresco.com/de/)

Both systems have licenses that are fully compatible with commercial use.

2.4 TMS

There are several commercial TMS systems (like SDL Trados, Transit, Across) and also some open source systems. The best known open systems are:

- OpenTMS (http://www.opentms.de/?q=en)
- OmegaT (http://www.omegat.org/)

OmegaT is GPL licensed, OpenTMS is licensed as EPL. The license issue discussed above is less severe for the TMS. If the TMS is used without any programmatic add-ons, both licenses are sufficient.

2.5 Publication

Depending on what the format and structure of the source data is and what target formats are required, there are several ways of converting the data into the target formats. Even if only DITA is considered, there are several options.

The obvious approach is the DITA Open Toolkit. The DITA-OT is capable of rendering bookmaps to PDF, XHTML, CHM, RTF and other formats. The transforms and style sheets that come with DITA-OT can be customized to meet the requirements of the corporate layout.

While publication to PDF and XHTML works quite nicely, there are some issues with special characters in other target formats.

WebWorks ePublisher can create several target formats from DITA and other sources. It is a commercial alternative.

Another common approach is FrameMaker based rendering, that means, a rendering system implemented using FrameMaker programming interface to compose publications from structured source data.

There are PDF libraries like iText, allowing programmatic composition of PDF. There are two versions of iText. An open source version under the GPL (forcing the custom rendering code under the GPL as well, which is undesirable) and a commercial license, not imposing such restrictions. However, the commercial license is quite expensive, even compared with commercial solutions. iText was originally available under the LGPL which allows rather easy integration into commercial solutions. As of 2009, it is dual licensed under the Affero GPL (which most likely prevents all commercial use), and a very expensive new license.

Other more exotic methods include rendering in OpenOffice using the API, or expressing the layout in TeX.

What method(s) to use surely depends on the layout requirements. It probably makes sense in most cases to estimate first how much effort it would be to customize DITA-OT. This gives a rough figure to see if commercial or open alternatives can compete.

2.6 XML Editors

Beside alternative products that are also used for XML editing (or DITA editing, in particular), such as Adobe FrameMaker or Microsoft Word, and beside web-based products, the major XML editors suitable for technical authoring are:

- Arbortext Editor
- XMetaL
- Syntext Serna

The former two commercial products are well known, while Serna is not as widely spread.

Serna is a product to be considered though. It is available in a free edition under an open source license and a commercial one. There are only few differences between both, the most important of which is the absence of a programming interface in the free edition. CMS integration (for example: check in a module from the editor's menu) is only possible with the commercial version.

2.7 Content awareness

There are several fields in such a solution where content (= DITA) awareness is relevant.

- Searching

  Information in the DITA content should be available for search.

- Editor display

  The layout in the editor should be rendered in a reasonable way.

- Reference mapping (inbound / outbound)

  The system must have the information, where in the XML structure references can be found.

- Publication

  If a DITA-aware publication solution is used (like the DITA-OT), this requirement is covered by the publication solution.

In Cinnamon, reference mapping and the search index can be configured to match arbitrary XML grammars. All three major XML editors are DITA-aware, and DITA-OT and other publication tools are DITA-aware.

If Documentum is used, the searching and reference mapping points are covered with equivalent solutions.

Other ECMS need to provide similar mechanisms to be usable as a platform.

2.8 Annotation

There seem to be no open source tools allowing PDF annotation, however there are some free (but not open) tools that claim to enable PDF annotation. Adobe reader has also some basic annotation features. Still, all free products with annotation capabilities seem to be stripped down versions of larger software suites with a price tag.

If requirements are above basic level, commercial products either from Adobe or alternative vendors should be considered.

If review does not need to be PDF-based, web based annotation is a good alternative. The content is rendered in HTML format and annotations can be entered directly. There are several open source solutions for HTML based review.

3 System composition

3.1 General remarks

With the various tools that we discussed in the previous section in mind, we can now compose a system. In the best case, it is possible to use only open source components, but in reality, open and commercial components will be mixed. Perhaps, a relational database, editors or the publication module could be commercial while the other components are open source.

The system described here is composed entirely of open source components. I'd like to repeat that if I omit a product from this list, it is not necessarily inferior. In most cases, there exist one or more valid alternative products.

On the other hand, the configuration described here has been implemented several times with some variations, and is in production use at several companies. It is therefore a good starting point that can be modified as needed.

In the following, the components making up the server and the client will be listed in tabular form.

There is a reference implementation of most parts of this system available as a free DVD with a Linux-based server VM and a .net based Windows client software. The system is preconfigured with DITA. Please contact the author for details.

3.2 Components

3.2.1 ECMS Server

Component type	***Reference implementation***	***Alternatives***
Operating System	Ubuntu Server 10.04, 32 or 64 bit	The software runs on Windows as well. It probably runs under other OSes as well.
RDBMS	PostgreSQL 8.x	MS SQL Server or most other RDBMS systems.
Java	Java 6	---
Servlet Container	Apache Tomcat 6.0	Basically, other containers work, but no compatibility matrix is available.
ECMS	Cinnamon ECMS	The same general concept applies to other systems as well. Other ECMS software like Alfresco or the commercial Documentum are possible.
Index Engine	Lucene is internally used by Cinnamon for metadata and full-text indexing	Like Cinnamon, other ECMS products typically bring their own indexing engine.

3.2.2 Publication Engine

Component type	***Reference implementation***	***Alternatives***
Operating System	Ubuntu Server 10.04, 32 or 64 bit	The software runs on Windows as well. It probably runs under other OSes as well. Some available rendering tools are only available under Windows.
Java	Java 6	---
Render Server	Cinnamon Render Server	If another ECMS is used, the rendering solution of that system should be used.
Rendering Software	DITA Open Toolkit Render PDF and XHTML (customizable) Render CHM and RTF (fix required in DITA-OT)	FrameMaker based rendering or WebWorks ePublisher. For proprietary interfaces, typically, custom software is required.

3.2.3 Client

Component type	***Reference implementation***	***Alternatives***
Operating System	Windows XP / 2003 or newer	Mono is not yet supported. .net 4.0 has the OS requirements specified here.
.net Runtime	.net 4.0	---
ECMS Client	Cinnamon Client 1.3	If another ECMS is used then the client software of that system must be used.
Editor	Syntext Serna Free Edition	For everyday users a commercial editor should be used. Either the commercial edition of Syntext Serna, or XMetaL, or Arbortext Editor.
TMS: Java	Java 6	---
TMS	OmegaT	If OmegaT is not sufficient, commercial TMS software should be considered.

4 Summary

In this article, I showed an example for a DITA-aware, ECMS-based system for technical authoring and publishing. The system is built from open source components and can be

freely distributed. If budget is low or there are concerns about vendor lock-in, open source systems should be checked in any case. But also if these limitations do not apply, money not spent on license fees can be spent on better customization, or simply saved.

The presentation on the NL DITA 2011 will add to the information in this article by showing the system from a user's perspective.