Prototype of a Web and XML Based Collaborative Authoring System

Bo HU
Universität der Bundeswehr München, D-85577 Neubiberg, Germany
bo.hu@unibw-muenchen.de

Florian M. LAUCK
Universität der Bundeswehr München, D-85577 Neubiberg, Germany
florian.lauck@unibw-muenchen.de

ABSTRACT

This paper describes a prototype of a web based, light-weight collaborative authoring system which enables a distributed team of authors to work simultaneously on different sections of the same technical document which is being held on a web server.

To enable concurrent accesses a hierarchical document model is introduced which defines a document as a collection of referable objects which are stored using an XML centric database structure. Some of these objects are text objects which can be edited using a browser embedded HTML editor which is developed in an OpenSource project „Richtext“. XML and XSLT technologies are deployed to facilitate document formatting and exporting.

Keywords: collaborative authoring system, XML, browser embedded Richtext editor, object-oriented design, content management system (CMS)

INTRODUCTION

Writing a technical document today is becoming a more and more collaborative process. Very often it is carried out by people working at different sites. In the last years many electronic systems have been introduced to support collaborative work. However, the de-facto way to support this collaborative and distributed process is to exchange draft versions between the authors via email[1].

Conventionally a technical document is treated as a single (and complex) object. That makes it difficult (although not impossible[2,3,4]) to enable multiple simultaneous accesses to a single document. Furthermore it complicates the management of different access privileges upon the different parts of a single document by different authors. Both aspects are essential to collaborative authoring.

On the other hand a technical document can also be treated as a hypertext consisting of hierarchically organized text sections, numbered figures and references at the end of the document, as shown in fig. 1. We introduced the class „referable“ as the general class for the three classes: section, figure and reference. Once a document is created referable objects can be created and saved in a database, and they can be updated, reorganized and deleted as well.

SYSTEM OVERVIEW

As mentioned above our basic concept for a web and XML based collaborative authoring system is to treat a technical document, for example this paper, as a collection of hierarchically organized text sections, numbered figures and references at the end of the document, as shown in fig. 1. We introduced the class „referable“ as the general class for all three classes: section, figure and reference. Once a document is created referable objects can be created and saved in a database, and they can be updated, reorganized and deleted as well.

A web system is predestinated for distributed users. Most of these content management systems today provide basic WYSIWYG (What you see is what you get) functions which are sufficient for the most situations of an authoring process. However, web environment is not suitable for sophisticated high quality layout works. As we understand it a collaborative authoring process of technical documents consists of two phases: collaborative writing and processing for final printing. Since the second phase needs much less collaborative support, to omit comprehensive layout functions is a reasonable alternative for a web based collaborative authoring system. Such a system should in contrast just provide some system interfaces - ideally XML based - to export all necessary data to some desktop publishing or word processing systems.

In our recent work we extend the features of an existing XML centric web Content Management System to meet the needs of a distributed collaborative authoring process of technical documents.

Fig. 1: A simple document object model

Based on this model a collaborative authoring system can be built as an extension of an existing XML based web content management system (called „CMS@inform“). Fig. 2 and fig. 3 show the use cases diagram and the system block diagram of the proto-
type of our collaborative authoring system, respectively.

Using this system a project manager can create documents and assign any further user to be a member of his/her project.

Within a project each author is assigned with privileges allowing him or her to
- read a given referable object
- browse a given referable object
- create sub-objects of a given referable object or
- change the properties of a given referable object including ownership.

The complete document structure is built up in a database. The different sections figures and references of a single document are organized hierarchically just the like elements within an XML document and stored in the database. Based on this concept a rudimentary object level version control is also implemented. Once the collaborative writing phase is finished all objects of the document can be exported for final processing and printing.

This prototype uses Moodle - an OpenSource learning management system as one of the authentication services. An OpenSource HTML editor called „Richtext“ is deployed to provide a basic WYSIWYG interface for creating and editing sections and references.

In the following the functions of each component will be discussed.

**XML centric Content Management System (CMS)**

A web content management system is the basis of the collaborative authoring system. This content management system has been developed by the authors based on Windows ASP (Active Server Pages) technology. Its architecture is XML centric regarding information processing and data storage (fig. 4). The content management system on the one hand makes available all the necessary XML data for a HTML page, and on the other hand supplies the XSLT templates which should be used for generating HTML information from these data. Different HTML pages may use different XSLT templates. When a client requests an HTML page the CMS responds with XML information from the database. A XML capable browser translates this information into presentable HTML using the necessary XSLT template which it fetches from the server in a second request. When a non-XML browser is used the XML information can be translated to HTML on the server side as well. Because of its XML centric and simple data model on the database layer the CMS is flexible regarding extensions (see fig. 5). In most cases extensions can be realized by making adoptions in XSLT. Actually the database structure of the CMS has never been changed since it was first introduced in 2000 although a few new applications have been realized based on this system.
Based on its flexible architecture the CMS provides basic functionalities for managing contents and user privileges. A light version of ACL has been implemented which allows to define exactly one user group to:
- read
- write
- append or
- browse

a certain page. In other words: if a user is supposed to have full access to all objects he/she has to be a member of all groups. This simplification makes it even easier to implement the authorization using the existing CMS kernel. As a matter of fact all the user interface components are written in XSLT and thus managed by the CMS itself. Since concurrent writing processes may cause data loss a simple page locking mechanism is implemented to avoid access conflicts. The entire system which is meant for a platform of prototypes can be outlined as follows:

Fig. 6: Class Diagram of the used CMS

It is important that this CMS treats the different HTML pages as objects in a hierarchically organized structure. That is exactly the structure in which a technical document is organized. We are able to extend the features of the content management system to meet the needs of a distributed collaborative authoring process of technical documents which are, in turn, a collection of hierarchically organized referable objects. Each of these referable objects is treated as a HTML page and can be referred to using its URL.

A essential step is to implement a new function to show and to export an entire document which consists of many referable objects crossing several hierarchal layers. That means with one click of the mouse all these objects should be shown recursively on the screen and numbered properly.

Generally spoken there are two ways to export the document. The first one is to export the data in some special native target format, for example, WORD 2003's XML format or QuarkXpress tags. These are the programs with which the document should finally be processed and printed. Another possibility is to export the data into some neutral standard format, like HTML. We adopt the second method for our prototype. In this case the outline information is transformed into HTML heading tags. Using XSLT technology the entire document including hierarchical document structure can be exported to WORD 2000, as seen in fig. 7.

Fig. 7: Imported HTML with outline information

**Moodle**

Moodle is a Learning Management Systems (LMS) - a OpenSource software package designed to help educators create online courses. As the default authentication method Moodle uses email confirmation. When the user signs up choosing their own new username and password, a confirmation email is sent to the user's email address. This email contains a secure link to a page where the user can confirm his/her account. Future logins just check the username and password against the stored values in the Moodle database[8].

Our system uses an existing Moodle installation as an authentication service. This is achieved by implementing an additional PHP script which reads the authenticated username from a Moodle session and passes it on to another application requiring authentication. In this way a single sign-on is realized: Once a user has logged into Moodle he/she can use any other applications using Moodle as a central authentication service.

**RichText**

A collaborative authoring system should provide an editor with at least some basic WYSIWYG features. Some projects use special HTTP clients[9] whereas there are several OpenSource projects[10, 11, 12] aiming to develop browser based HTML editors. We have chosen Richtext which provides not only sufficient features for writing and formatting but also facilitates an XHTML export. This is essential since we use an XML centric content management system. Another reason for the usage of Richtext is that it was included in the first version of Moodle we used.

Richtext proves to be easy to integrated into our existing
system. It is a WYSIWYG HTML Text Editor which requires no additional components than Internet Explorer, for it is written entirely in DHTML and JavaScript. It can easily be embedded in a HTML form using a single HTML tag `<object ...>`. This fact is important since our CMS uses its own JavaScript functions as well.

Using Richtext an author can create and edit an HTML text object which may be a section or a reference. The author can also insert images and hyperlinks into his/her text. That means a text object is a collection of HTML elements of which some are image (<img ...>) and anchor (<a href=...>) elements. The result is saved in XHTML format to database.

Now let’s have a closer look at an anchor element. As defined above sections, figures and references are referable objects. According to our document model a document is nothing but a collection of referable objects. As shown in fig. 8 a cross reference is a hyperlink to a certain referable object within this collection.

To distinguish these hyperlinks from the „normal“ ones four new classes - „Number“, „Section“, „Title“ and „NumberTitle“ - for the anchor elements are introduced. When an author wants to use a cross reference, for example, to a certain figure of the document he/she just inserts an URL to that figure and selects one of these four classes to specify what kind of a cross reference he/she wants to use. For example, to have a cross reference to the figure below as a number an author just has to insert a URL to this picture and choose the anchor class „Number“. Instead of the URL the correct number of the figure (“8”) will be displayed when the entire document is viewed or exported.

![Fig. 8: Document Object Model (actually implemented in prototype)](image)

**DISCUSSION**

Although the development is still an on-going process it may be useful to review the benefits and limitations of the current version of the web based collaborative authoring system.

**Benefits**

**Web based system.** A web based system is predestined for distributed users. The current system does not need any local installations. Using this web based collaborative authoring system authors will not be forced to work on different local copies of a single document anymore. No time consuming check-in/check-out operations of the whole document have to take place. Instead of this each section can be locked for editing separately. The document can be viewed and edited in HTML based on XML technology (see fig. 9 and fig. 10).

**Referable objects.** A technical document is treated as a collection of hierarchically organized referable objects - sections, figures and references. Figures can be uploaded and managed by the system as referable objects. Due to the nature of our web based, i.e. browser based system the only formats supported now are JPEG, GIF and PNG. A subsystem for managing user privileges is realized on the level of referable objects. Based on this concept our system provides also basic features for cross references and automatic numbering of sections, figures and references. The system also facilitates a rudimentary version control on the object level which enables to undelete objects and to view the change history of an object.

**Basic WYSIWYG features.** Using the browser embedded HTML editor „Richtext“ our collaborative authoring system provides basic WYSIWYG features for creating and editing text objects. Richtext and XSLT make it easy to achieve homogenous formatting of a document written by different people.

**Reusable document parts.** Additionally standard components of technical documents can be managed easily: Each document has only to have a hyperlink to such components which do not need to be copied to each document.

**XSLT based HTML export.** A special XSLT template enables the export of the entire document into HTML.
format which can be interpreted by word processor or DTP systems for final processing and printing.

Comments and status of referable objects. Co-authors are able to add personal contributions (e.g. ideas, opinions, hints) to other document parts. By picking up these comments, the individual writing process of authors can be improved. Furthermore this way of exchanging information can also be used to give coordinating directions to subordinate authors or to initiate alteration of text passages by a superior author. Additionally an referable object can be set to have some specific processing status and priority. These features are used as means of communication, coordination and collaboration.

Fig. 10: Editing a document in a browser

Limitations

Because of the nature of this browser based system and because it is a prototype there are some substantial limitations.

Processing graphics. In contrast to a standard word processor a web system has limited capacity for processing graphics of which the formats are limited to GIF, JPEG and PNG.

Export of numbering of the referable objects. The numbering of the referable objects - sections, figures and references - can be exported to other word processors or DTP programs for final processing and printing. However these numbers are exported as text constants. That means that they can not be renumbered automatically during the final phase.

Cross references to page numbers and footnotes. Because our web based collaborative authoring system does not support page breaking at all there is no possibility to support cross reference to page numbers or footnotes.

WYSIWYG. The current system supports only limited WYSIWYG functions.

Spell checking. The current system does not support any spell checking functions.

Prototype. To reduce the project time on the first stage we have designed a prototype presented in fig. 8. The original class diagram does include further elements like annotations, tables and even appendices.

Fig. 10: Editing a document in a browser

CONCLUSION AND FUTURE WORKS

Having implemented this prototype of a web and XML based collaborative authoring system we are confident that the web is the right electronic platform for office automation and especially for distributed collaborative works. The most important step for this prototype is to treat a technical document as a collection of referable objects. We are looking forward to future works including:

Enhancement of the export interface. It is planned to provide an export function into native formats of WORD 2003 and QuarkXpress’ tags. The better the export interface the more work can be carried out within the collaborative environment.

Realization of an import interface. An interface which enables imports of more than one referable object at one time may increase the usability of the collaborative system especially for people which prefer non-web working environments. Additionally it may provide a backup and restore solution together with the existing export interface.

Development of a dedicated collaborative system. The current system is built on top of an XML based web system which is a platform for multiple co-existing projects. A dedicated system may increase the performance, enable easier maintenance, integrate more collaborative features and provide a more user friendly interface.

Enhancement of graphics support. Particularly several vector graphics formats are useful and should be supported by a collaborative authoring system. This can be realized using some browser plug-in’s.

Deployment in teaching and research. Last but not least: It is most important to integrate such collaborative systems into the daily teaching and research practices to gain more experience and more impulses for the further developments.
REFERENCES


