XML Authoring Tool

MIHAI TEOHARI
COPCEA LARISA
Computer Science
The Bucharest Academy of Economic Studies
15-17, Dorobantilor Street, District 1, Bucharest
ROMANIA
mteohari@yahoo.com larisa.copcea@yahoo.com

Abstract: Within the scope of publishing industry, there are various solutions which address the issues which arise from the cooperative effort accomplished by publishers and article editors. Many of them employ complex software solutions, lack of flexibility, high costs or other drawbacks. In fact the solutions must be efficient, flexible and cost effective for the publishers and in the same time, easy to use and secure for editors.
This kind of practical solution we are trying to describe in the current article, based on our own research.

Key-Words: Authoring, Word Content Control, Word Add-in, XML

1 Introduction

In a publishing company, the editorial stage demands a continuous flow of information between publishers and authors. From the moment the article is developed until the magazine or journal is sent for printing, this article takes part in a complex process of correcting, reviewing, editing again, formatting, pagination etc.
These activities belong to both authors and publishers, so there is clear necessity for that data used in the process to be error free and easy to operate with.
Activities that the publishers are responsible for (formatting data, paging etc) demand specific desktop publishing software solutions, while authors are most likely, regular computer users. This type of users doesn’t employ such complex solutions as the publishers, relying mainly on word processors.
Considering all these aspects which involve working on common data using incompatible systems, many publishers tried to standardize a common data format. While a single data format is not completely determined, publishers must improve their workflow in order to became more productive and in the same time allowing authors to focus on their work and not having to learn complex authoring software solutions.

2 Problem Formulation

Following the aspects described in the intro section, the idea of improving the data exchange process between publishers and authors stands out.
Publishers need a set of tools and templates for the applications used by the authors, so that no data lose is encountered. They rely on data stored in XML files which have a specific data structure and are already integrated in their publishing workflow (desktop publishing solutions are able to load them and manage them in their environment).
For the authors working environment, Microsoft Word 2007 (and later) is considered as their word processor.
Each author must be able to create and edit articles using Microsoft Word, but in the same time the content they are working on must be compatible with publishers’ data format. This data format must be respected in a very natural way, so that the authors mustn’t been aware of it.
Authors do not have to learn document’s structure; they only have to be guided to follow this structure. Adding content outside the specific structure would make the documents incompatible with publishers’ systems.

3 Problem Solution

The reference points extracted from the requirements were identified in the following list:
• Data exchange between publishers and authors is based on XML files;
• Article’s structure is described by an XML schema (we are considering DTD and XSD schema types) but it is not completely settled, therefore we should assume that multiple schemas may be used by various publishers;
• The user interface must be integrated with Microsoft Word for maintaining a familiar and efficient work environment for the authors.
As some of the requirements stated (ease of use and integration with the familiar user interface of Microsoft Word), the solution is basically an enhancement added to the Microsoft Word workspace in order to provide authors a comfortable but controlled medium for their activity.

This was achieved developing a Microsoft Word add-in, which basically maps the content of the articles to special controls aimed to closely represent the XML file’s tree-like structure. This way, authors will actually edit a structure defined by an XML schema but instead of XML elements, they will work with Word controls.

The most important fact is that authors aren’t aware of the details implied by the XML schema and they work with their articles based on the controls made available by the add-in.

We have mentioned a set of Word controls (in fact they are called Word Content Controls) and to clarify this aspect we have to put them in direct accordance to XML element types found in articles’ content. Each XML schema must have a set of corresponding controls in the user interface in order to allow authors to edit any type of content described by it.

When an XML schema is employed by the publishers, this schema must be analyzed and the result must contain a set of controls. Because there may be various schemas, a single set of controls is insufficient and therefore a tool is needed to generate new sets of controls.

This is the second component of our solution, a tool (more precisely a Windows application developed with Microsoft .Net Framework 2.0) which takes an XML schema as input and generates a set of controls ready to be made available to the add-in specified above.

The output of this tool must be compatible with Microsoft Word API and usable from the add-in. The best solution found was to generate a Microsoft Word template file containing all these controls.

The template file can be easily deployed to the authors’ computers as soon as a new DTD schema is made available.

The whole solution is represented in detail in the following picture:

![Fig.1](image)

During the next chapters, each component and concept will be described in detail.

### 3.1 Important concepts

- **Exchange data format**

  XML was chose by the authoring community as the best data format which can address these requirements. From its multiple advantages, a few of them were essential to our scenario:

  - Xml files can store both structured and unstructured content (which is very common in article and book content).
  - Data format is independent of the operating systems or even solutions employed by the editors;
  - In fact, based on specific schemas XML data can be easily integrated with a Content Management system or a Database system in order to store these data in a secure and structured form.
  - XML files can be constructed following a schema which can impose a specific data structure used by the editor or even better used by the scientific community;
  - There are several wide-spread xml schemas (as DTD or XSD versions) used in the scientific word like NLM TaxPub (see [3]).

- **Content Controls**

  A Content Control is a new concept introduced in Microsoft Word 2007 (see [1], and [4]).
Basically it is a control that is added to a Word document and allows users to input content within a predefined structure. This way, the user is prevented from changing the look, feel and layout of the document.

There are several types of content controls like: text control, rich text control, picture control, drop-down list control and the calendar control.

Each of these controls defines what content it can contain and users are restricted to entering only that content (eg. plain text, rich text, date, image etc).

Also a very important feature which helps users to create a document template is the ability to specify if the content can be edited or deleted. This is essential for static content (when this content cannot be further edited) or mandatory content (when specific structures are inserted into the document and can be completed with further content but not accidentally deleted).

From the look and feel point of view, content controls are completely nonintrusive for users because their content is displayed as a regular Word document content and only when users click on it or move their mouse over the control’s content, then the content control is displayed.

This way a document can have a specific structure imposed by a template, but for users this template is not visible for the most time of working with the document.

3.2 Solution components

3.2.1 Schema Loader

The first operation that must be completed in order to operate with data consists in generating the content controls corresponding to the elements defined by the XML schema.

The content controls are stored in a Microsoft Word internal template so that they will be available to users when they edit the document.

The schema loader tool takes a DTD (or XSD) schema and iterates over all its elements generating content controls necessary for the Microsoft Word add-in. The result of this step is the word template file which together with the schema file will be deployed to the user machines.

Content controls required for the simple elements in the schema are pretty straightforward, as you can see in the following example where a ‘label’ element is represented as a content control:

```xml
<!ELEMENT label (#PCDATA | email | ... )>  
```

![Fig. 2](image)

In the picture above was chose a Rich text content control with only the Title property set with the element’s name. We chose this type of content control and not a Simple text content control, because as you can see in the element’s definition, its content can contain besides text, other nested elements like ‘email’.

There are also more complex elements which implement parent-child relationships and more important, the child elements are mandatory. When these complex elements are found, they will no longer be decomposed into single content controls similar to ‘label’ control from the previous example, but they will require composite content controls.

The mandatory element will be nested into its parent control and users wouldn’t have the ability to delete it.

In the following example, the ‘article’ element contains a mandatory ‘front’ first element, while the other elements may appear inside it or not:

```xml
<!ELEMENT  article (front, body?, back?, floats-group?, (sub-article* | response*)) >
```

This type of element will be represented in the Microsoft Word document as a composite content control named article with at least one child content control named ‘front’:

![Fig. 3](image)

3.2.2 Microsoft Word Add-in

A Microsoft Word add-in is a software application developed with Word API and integrated in the Microsoft Word solution. The reason for developing such application is to enhance Office applications with new features which suit user needs better.

The add-in technology was chose to control Word’s editing process primary because it has a very powerful integration at the User Interface level (users benefit from the standard Microsoft Word interface which is familiar to them) and also allows developers to control and manipulate content at application, document and...
The main editing process is controlled by the add-in application as well as the other two important operations: loading the Word document from an XML file and serializing a Word document (edited using content controls) into an XML file.

### 3.2.2.1 Creating and editing the Word document

In order to implement these features, the add-in works with two elements:

- The XML schema contains all the rules which determine the nesting hierarchy for the content controls;
- The Word document template generated with the tool described at the previous chapter contains all the content controls described by the XML schema.

Having the content controls and also the rules which manage them, editing the document according to a common structure is reduced to a much simpler task. There is no need for the users to have any knowledge about the XML schema and the validations involved. All these are resolved internally by the add-in in the following way:

- first of all, the location in the document is determined – it is identified the content control which has the focus;
- from the content control is inferred the element type in the XML schema; this step is particularly important because the element definition contains information about the child element types allowed for this element and their cardinality;
- from the rules extracted at the previous step, a set of child content controls is displayed to authors;
- the selected content control is copied from the Word template and inserted at the current location;

According to the algorithm described above, users know every moment what type of content controls are allowed to add to the document and more important these controls are controlled by the XML schema. When creating a new document, the process is the same, but according to the XML schema, the only content control allowed is the ‘Article’ control – which acts as a root control.

Let’s consider an example of a small XML fragment:

```xml
<fig-group>
  <caption>
    <p><bold>Figure 3.</bold> Distribution and habitat association of Nixonia masneri and N.mcgregori. Biome map from Rutherford et al. (2006).
  </caption>
</fig-group>
```

Its representation as a Word page developed using content controls is shown in the picture below:

### 3.2.2.2 Exporting a Word document

When the author needs to submit the article to the editors, the add-in must ‘translate’ the article from the Microsoft Word format to an XML format used by the editor. This XML format is of course described by the XML schema used in creating or editing the document. The document’s content controls hierarchy is parsed from the deepest control to the root, every content control being mapped to an XML element. The mapping process consists in creating an XML element and loading it with the data entered in the content control. The result of this recursive traversing of the document is the final XML document. Because the content controls were enabled for editing according to the XML schema, further validations are not required. At this stage we have a valid XML document ready to be sent to the editors.

### 3.2.2.3 Loading an XML document

The processing involved in this operation combines routines used in the previous operations of editing the document and then exporting the document to an XML file. More precisely, when the XML file is loaded, the XML nodes tree is traversed recursively similar to the loading routine, but the mapping works in the opposite direction: the XML element is mapped to a content control.
At this moment, the process of loading content controls from the Word document template is automatically accomplished for each element in the XML file.

3.2.2.3 Storing data and integration with other systems

Even if documents storage and management are not covered in this solution and also the documents are handled in a disconnected mode, a few considerations are necessary.

All data flow between publishers and author is accomplished using XML files as we already specified. These files contain mostly structured data and precise rules that handle this structure.

In the last years, XML data format has become widely spread around all IT community. This fact is reflected in the support offered by most database systems and document management systems.

Also, from the limited support offered by the first implementations (which used to map data from XML documents to relational databases), at this moment probably all major database management systems offer specific XML data types. These XML data types allow storing whole XML files and validate their content against a schema defined at the moment when the XML data type is declared (see [2]).

Due to the existence of XML schemas for the files used in this solution, the XML data type mentioned before is perfectly suitable for storing these files. So, storing solution’s files is rather easy and regards only publishers’ preferences for a certain database type.

The add-in may be easily integrated with other types of systems, so that the articles are stored in a central repository and users benefit from advanced features like versioning control, check-in / check-out, notifications, approvals etc. These types of systems are document management systems and provide much more specific features (like the ones we’ve mentioned before) than databases.

Integrating our solution with the systems described above would require little effort. Existing web services could be consumed in order to achieve a secure and controlled environment for loading XML files from the central repository, editing these files and sending back the updated versions.

4 Conclusion

Through the article we have address a common problem found in the publishing industry – the necessity of implementing a simple and comfortable solution for text editing by both authors and publishers, but in the same time restricting user actions to a set of rules which guarantee a common content structure.

The solution we have found answers these requirements and more than it is flexible enough to allow further modifications on the data structures used by publishers.

Regarding the storage possibilities for the articles involved in the process, the fact that we didn’t address this aspect regards the variety of implementations existing at every publishing house. These implementations may vary from custom solutions developed over relational database management systems, to complex document management systems.

From this point of view, a common approach to manage the storage of the articles on the publishers’ repositories is difficult to be found. Nevertheless, as a future development, the solution can be transformed in a connected application, but the implementation would be specific to each publisher’s existing systems.

References:

