Some comments on the NLM XML plugin for Word 2007

I have been very slow getting to this, and I missed catching up with Microsoft people at Open Repositories 2008 but I wanted to make a couple of comments on the new Microsoft Word 2007 plugin for authoring journal articles in the (USA) National Library of Medicine XML format. I saw this originally via Brian Jones at Microsoft.

I have several questions about this plugin. Actually, they're concerns more than questions.

I will look briefly at the plugin here and then try to put it in historical context and then as always I will suggest another approach using the Integrated Content Environment, which we're probably not going to have time to try out in the short term.

Microsoft's Pablo Fernicola says:

> The goal is to simplify several activities in the publishing workflow, from authoring to publishing and archiving, with this last step including conversion to the XML format from the National Library of Medicine. The current process of getting an article from the authors to a journal (increasingly electronic only) is a bit complicated and many times lossy, especially in relation to the metadata related to the article, we hope that the add-in will help simplify and improve the process.
>
> http://blogs.msdn.com/exscientia/archive/2008/03/20/Technology-Preview-Launch.aspx

There's a video. Which shows Pablo creating a new document from a template and adding some stuff to it. After watching the demo I downloaded the software and tried it out. You need Word 2007 so that meant installing Windows XP in a virtual machine on the MacBook, along with Office and other stuff. I'm pretty sure this plugin will never be coming to Mac Word or to Linux which means cutting out a growing number of authors.

I'm not sure if the Add-In is really working for me. I get a template but I can't find any new user interface items or figure out how to add metadata. But then I can't find anything in that new Word 2007 interface because I have not invested time in learning it.

A few of the questions that occur to me:

1. Why does the demo show putting a table in the abstract? I checked, and it is allowed by the DTD, but I don't think that articles usually have tables in the abstract. Also, I don't think the demo shows valid NLM XML; shouldn't the table be wrapped in a section element?
2. What's with the 'lets go ahead and make some stuff bold and underlined' in the demo – are there some more semantic elements that could be used instead?
3. How come I can pick up the introduction and move it around my document? If I can do that what's the difference from a style-based system where the structure is implied by styles? Also, I'm a bit confused by the empty paragraphs between sections. Are they meant to be there? What if I type in one of them?
4. How would preservation people rate the resulting Word document? It's a kind of mishmash of two standards and as far as I can see. So to make sense of it you're going to need lots of user interface code.
5. What's the development cost on an effort like this versus the likely saving in time at the journal?
6. Are there any examples of similar efforts that work with decent ROI? Has anyone yet made a usable DocBook editor in Word? What about XHTML? TEI? I suspect the answer remains no, just as it was back in 2005 when Word 2003 had been around for a while. This suggests to me that Word is still not a good platform on which to build an XML editor. Happy to be proved wrong, though.
7. How do I repurpose my article for other journals when it's rejected? How about copy and paste with my other documents?
8. How are the pilot authors liking it?

9. How much support do we expect authors to need?

Now, lets go back in history a bit.

In 2005 I corresponded with Brian Jones of Microsoft's Word team about this kind of use for Word as an XML editor. I questioned just how the feature in MS Word that lets you mix in foreign XML schemas is supposed to work. Back then I called it the bizarre feature that lets you mix schema-controlled XML content in amongst Word's own structure. See Brian's reply in his comments with emphasis by me:

> **The first point is that our main scenarios weren't about turning Word into an XML editor.** As you can imagine, we have a fairly large user base, and investing the amount of resources we did into our XML support just to target the XML editor market wouldn't have made a lot of sense. The XML support is really for a much broader set of scenarios.
>
> There is a huge market that exists today for custom Office solutions. People customize the Office applications in all kinds of ways to try to get more out of their documents. By adding the support for custom defined schemas, we made it much easier to **build semi-structured solutions on top of Word**. Rather than rely on hacks with styles or bookmarks, folks could create a simple schema and add some XML tags into their existing document solutions.
>
> We provide a fairly rich object model on top of the XML functionality, as well as the ability to save an entire Word document as XML (using the WordprocessingML schema). These tools make it much easier to build document generation and consumption solutions, as well as more reliable add-ins that act on the document while it's being authored.
>
> http://blogs.msdn.com/brian_jones/archive/2005/07/08/436973.aspx#452483

So Word's XML schema integration is going to work at the level of small structural chunks like metadata but is not likely to work for schemas with several hundred elements in them.

I keep an eye out and I have not been able to find a single example of a full DocBook, or XHTML, or TEI or whatever application that's built on Word 2003+.

Going back further in history, who remembers BladeRunner? Never even made it to market.

What about Microsoft's own SGML author, terrible thing, more than ten years dead.

With my limited imagination I can't see how you can mash-together a generic word processor and an XML editor and get something that's going to be widely usable. Adobe Framemaker had a sort-of XML editing mode but it was not the sort of thing I'd expect ordinary authors to deal with, and setting up new documents required us to hire a consultant for a week just to get started. WordPerfect had a structured mode, but the way I remember it, it was very much segmented off from the ordinary word processing part of the application.

And even if this approach did work it represents a new kind of vendor lockin. Instead of having a standard format for your document as with OOXML or the NLM XML format you have an unholy mishmash of the two which requires custom code for users to edit it. The custom code runs in Word, so you have just lost one of the main benefits you're supposed to get from a standard format which is the ability to switch editing applications. Yeah, I know, the only thing on the planet that can edit full OOXML is MS Office and that's not likely to change in the near future, but if you take care about which features you use then you can get reasonable interop between word processors which you will loose with this kind of mishmash.

In my opinion one of the great things about word processors, and particularly Word, is that they allow you to manage structure by implying it. Word has a great outliner which lets you structure your work and move stuff around, while still allowing lots of freedom to copy and paste. It's simply not true what a lot of XML people say, that word processing files are 'flat'. Out of the box using headings, your Word document has an outline. Just because it's not serialised into nested XML doesn't mean the structure doesn't exist.

You can turn something into a heading by adding a style and turn it back into ordinary text the same way.

That kind of operation can require some serious gymnastics if you try it with a validating XML editor, cos a heading is not just a heading it's typically a magic element that's part of a bigger element which is part of an explicit hierarchy. Me I prefer working with implicit hierarchy and WYSIOOTMFYG (What You See Is One Of The Many Formats You Get) which is why I choose a word processor over an XML editor or a text editor even though I'm pretty sure I'm smart enough to use Emacs + DocBook if I wanted to.

I promised an alternative suggestion. So here goes.

Why not work out a generic set of word processor styles and microformats that can be used for academic authoring, which can be assembled into downloadable templates with the right look and feel for various journals and conferences, so authors need only learn one system of styles. Write some software that can render these documents as high-quality HTML, and PDF using the word processor itself, for low-end publishing. For higher-end work transform from XHTML to the XML DTD of your choice or just pump the content into Adobe InDesign or similar and forget the DTD like I said nearly four years ago in support of Tim Bray.

As an author, I'd prefer to have a generic way to write structured documents that I can use for all my writing. I'd really dread being expected to learn several variations on the XML embedded in Word theme. Or adapt to several different templates all with a different name for a block-quote, or a bullet list, or deal with templates that don't use styles at all, which is pretty much the way things are at the moment for academic authors.

Actually, the ICE team have done lots of work on this already, including a proof-of concept we knocked up in FrameMaker that used its structured mode to render XHTML back when most of us worked for the ill-fated NextEd.

We're working with a research group at USQ to test out these very ideas – they will work in ICE, either using MS Word or OpenOffice.org Writer, and we'll try to help out by providing import and export from the various word templates that journals provide.

I hope we can find the time and resources to explore this idea with the National Library of Medicine XML format as well – it would be nice to contrast the development cost and usability of a cross-platform ICE-based authoring environment versus the work that Microsoft has been doing on their Word 2007 solution.

Another nice to have would be a Word version of our word-processor to XHTML code. At the moment we use OpenOffice.org as a conversion-hub, but it has some limits to its Word support. A native implementation would be better for Windows users who want to use Word. At the moment we recommend serious ICE users use OpenOffice.org or a derivative but we do support Word.