The Digital Scholar's Workbench project: Final report

Ian Barnes

The Australian National University

Ian.Barnes@anu.edu.au

December 2007

1. Background and context

The Scholar's Workbench project is about providing better ways to work with word processed documents. These documents are not suitable for long-term preservation as they are; they need to be converted into suitable preservation formats before being placed in a repository.

2. What did we want to do?

We wanted to investigate the issue of preservation of word processing documents, and provide prototype software solutions for solving this problem. More specifically we wanted to select an appropriate preservation format for text documents and provide automated conversion from popular authoring formats (particularly Microsoft Word's .doc format and Open Document Format) into that preservation format.

We also wanted to provide automated conversion from the preservation format to popular and useful viewing formats like XHTML and PDF. As well as this, we wanted to provide simple ways for authors to insert their documents into institutional repositories. This involves not just providing a software bridge between authoring software and repository software, but also providing for automated or semi-automated extraction of metadata from documents. (There is no point inserting a document in a repository without good metadata; unless it can be found, it will sink without a trace.)

3. How did we try to do it?

The first step was to investigate the suitability of different text file formats for long-term preservation.

For the conversion and metadata extraction steps, we decided to use the XML-based Open Document Format as a starting point and to use XSLT to transform the contents into more suitable formats, and from there to viewing formats. We chose the Cocoon web application framework[1] as a container. We made an early decision to make the task much easier by only accepting documents that have been prepared using a particular set of formatting styles, namely those in the ICE template[2] from USQ[3]. We decided to leave until 2008 the task of normalising documents prepared without using styles.

We also aimed to get this software to a state where it could be tested by a focus group of early adopting users, and then modified based on their feedback and readied for mass use. Unfortunately it turned out that this goal was completely unrealistic. We underestimated the difficulty and complexity of the tasks involved, and overestimated how much work one developer/researcher can do. In the end, this project has been much more of a research

1 See http://cocoon.apache.org/2.1/ and also http://en.wikipedia.org/wiki/Apache_Cocoon.

2 Available from http://ice.usq.edu.au/svn/ice/downloads/latest/templates/.

3 See http://www.usq.edu.au/.

project, a feasibility study, than it has been a software development project that produces a finished product.

4. What did we actually do?

The first phase of the project involved research into sustainability of various types of text documents. This work is summarised in the report Sustainability of Word Processing Documents[4].

The second phase of the work was developing software for doing automated conversion of suitably formatted word processing documents into DocBook XML (an archival format) and from there into XHTML and PDF (viewing formats). This software, together with a web application interface, is called the Digital Scholar's Workbench[5].

The third phase was to develop the bridge between the Scholar's Workbench and the APSR RIFF submission service[6]. This involved extracting metadata, coding it using MODS[7] and packaging it, together with appropriate links, into a METS[8] metadata wrapper file. As this feature is new, it is worth going into a little more detail about its workings.

How the "Archive this!" button works

This assumes that the author already has a finished document, created using styles from the ICE template, and is viewing it inside the Digital Scholar's Workbench web application.

1. The author clicks in the "Archive this!" button.
2. The software presents the author with a metadata entry form, with as much of the information as possible already filled in with values extracted automatically from the document.
3. The author corrects and adds to this information if necessary, then clicks on the "Save" button.
4. The software uses this information to prepare MODS descriptive metadata for the document and wraps it in a METS wrapper conforming to the Australian METS Profile[9].

---

4 Available at http://www.apsr.edu.au/publications/preservation_of_word_processing_documents.html or http://www.apsr.edu.au/publications/word_processing_preservation.pdf.

5 A demonstration installation of the software can be viewed at http://workbench.anu.edu.au:8888/workbench/.

6 See http://www.apsr.edu.au/submission_service/index.htm.

7 See http://www.loc.gov/standards/mods/.

8 See http://www.loc.gov/standards/mets/.

9 See http://www.nla.gov.au/australianmetsprofile/ and also http://pilot.apsr.edu.au/wiki/index.php/METS_profile_development .

This METS file contains links to (and file sizes and MD5 checksums for) the original Open Document Format version of the document, plus derived DocBook XML and PDF renditions. (It would be simple to add XHTML to this as well.) This metadata file is saved to disk.

5. The software presents the author with a RIFF Document Archiving Submission form, pre-filled with most of the necessary information. The author has to choose a repository and a collection within that repository (although this will be pre-filled with that author's usual default so that in most cases no action will be necessary). The author may also have to provide some authentication information (username and password).
6. When this is complete, the author clicks the "Submit" button. This sends an HTTP Post request to the RIFF submission service, containing authentication, repository and collection details, plus the URL of the METS metadata file.
7. The RIFF submission service retrieves the METS metadata file.
8. The RIFF submission service finds the URLs in the METS file and retrieves the different versions of the document (and checks that the file sizes and checksums match).
9. The RIFF submission service deposits the document in the repository, together with its associated metadata.

5. What stage is the Scholar's Workbench at?

The Scholar's Workbench is a working prototype. It has not been rolled out to real users and is not yet at a suitable stage of development for everyday use. It can successfully convert documents in Open Document Format into DocBook XML, XHTML and PDF, and deposit them in a DSpace repository using the RIFF3 service. It can extract metadata from documents and then allows the author to edit and add to that metadata, before creating a METS wrapper file ready for sending to the submission service.

What did we learn?

- This is a much larger project than we first realised, and developing a full working application to the point where it could be rolled out to users was totally out of reach for a single developer.
- XSLT is a very good language for a lot of this work, however for the most involved stage of the conversion, a general-purpose programming language like Java or Python may well prove superior both in terms of performance and of code maintainability.
- These goals are feasible: word processing documents can be converted into suitable formats for preservation and inserted into repositories with minimal human intervention.

What's next?

- Extending the RIFF submission work to handle documents with multiple authors and with associated images or other resources.

- Merging the work done into the ICE project from USQ[10]. This would give users immediate access to useful features like document version control, large document management and IMS package generation for courseware. The ICE project covers similar but complementary concerns, and is further along in development, with a moderately large number of active users. By combining our efforts we can achieve more.
- Possibly changing the preservation format from DocBook to a new format consisting of XHTML plus mixins from other namespaces for specialised parts, plus microformats for class attributes and rules for the use of the div element to capture structure. Perhaps XHTML2? Perhaps using CSS3 and CSS Print for styling?
- Extending the work on metadata extraction to unstyled documents, using heuristics and possibly also text mining software to come up with keyword lists automatically.
- Experimenting with automated document normalisation: taking unstyled word processing documents and deducing the structure.
- Extending this approach to Powerpoint presentations. One possible target is the XHTML-based Slidy application for doing presentations inside the web browser.
- Extending this to large and complex documents. This might involve bypassing the facilities for this that are built in to the word processing software (master documents etc) and providing a web interface for organising large documents like books or theses. (ICE already has ways of doing this.) It would also be good to extend the repository submission service to handle large and complex documents.
- Working towards full interoperability of all major text file formats by choosing and working within a well-defined common subset of features. This would involve building converters between a chosen preservation and interchange format and all of the popular authoring and viewing formats: Word, ODF, HTML and PDF at least.
- Improving the quality of PDF output for printing. At the moment the Scholar's Workbench uses XSL-FO, which gives fairly poor results. ICE uses the PDF export built in to the word processors, also not ideal. However there are two possible high-quality typesetting options available. The first is to convert documents into LaTeX format and then run the PDF-LaTeX processor on them to produce typeset PDF. The other is to convert documents into a form suitable for import into Adobe InDesign, and prepare suitable InDesign templates so that most formatting of the document is done automatically in InDesign, and a typesetter/graphic designer only has to do the final fine tuning of the document before printing.

10 See http://ice.usq.edu.au/.