System Description: A Semantics-Aware LaTeX-to-DOCX/ODF Converter

Lukas Kohlhase and Michael Kohlhase

Mathematics/Computer Science
Jacobs University Bremen

Abstract. We present a LaTeX-to-Office conversion plugin for LaTeXML ML that can bridge the divide between publication practices in the theoretical disciplines (LaTeX) and the applied ones (predominantly Office). The advantage of this plugin over other converters is that LaTeXML conserves enough of the document- and formula structure, that the transformed structures can be edited and processed further.

1 Problem & State of the Art

Technical documents from the STEM (Science, Technology, Engineering, and Mathematics) augment the text with structured objects – images, mathematical/chemical formulae, diagrams, and tables – that carry essential parts of the information. There are two camps with different techniques for authoring documents. The more theoretical disciplines (Mathematics, Physics, and Computer Science) almost exclusively LaTeX, while the more applied ones (e.g. Life Sciences, Chemistry, Engineering) use Office Suites almost exclusively. Transforming between these two document formatting approaches is non-trivial: The TeX/LaTeX paradigm relies on in-document macros to "program" documents, empowering authors to automate document aspects and leading to community-supplied domain-specific extensions via LaTeXpackages. Office suites rely on document-styles that adapt visual parameters of the underlying document markup either document-wide or for individual elements.

This incompatibility of document preparation approaches causes friction in cross-paradigm collaboration as each camp deems their approach vastly superior and the other's insufferable. In this paper, we will discuss the transformation from TeX/LaTeX to Office documents.

copy from PDF	paste (libreoffice)
$h_{\mu_\varphi}(f) + \int_X \varphi d\mu_\varphi = \sup_{\mathcal{M}(f,X)} \{h_\mu(f) + \int_X \varphi d\mu\},$	$h_{\mu_\varphi}(f)+\square\ \varphi d\mu_\varphi = \sup\{h_\mu(f)+\square\ \varphi d\mu\},$

Fig. 1. Copy & Paste in Word Processors

There are several methods to transform papers from LaTeX to an office word processor. The first method is to just generate a PDF file and then open this

file in Word/LibreOffice. This achieves the goal of looking like the desired PDF document, just in Office. There are two problems with this route:

1. mathematical formulae are not preserved (see Figure 1)
2. even if the result looks OK the results have lost their links (e.g. for citations/references or label/ref), or become difficult to edit, because they do not conform to the styling system of the word processor.

The fundamental problem is that it converts the appearance of the document and loses meaning due to macro expansion. This is especially blatant when looking at the math in a document. Either it is treated as text, with no meaningful way to distinguish between math and formatted text that happens to contain some mathematical symbols, making automatic treatment of this kind of math difficult, or it is represented by an image of the relevant formulae, which makes editing extremely impractical if not impossible. The same holds true for references, they are essentially treated as parts of text with a linked number in front of them, complicating adding new references substantially.

The other way of transforming LaTeX to Word, by transforming the LaTeX source file directly, avoids these problems. `latex2rtf` [L2R] is a widely used system that uses a custom parser to convert a non-trivial fragment of LaTeX to the RTF format understood by most office systems. The system works well, but coverage is limited by the LaTeX parser and the aging RTF format. TeX4ht [T4HT], which uses the TeX parser itself and seeds the output with custom directives that are parsed to create HTML has a post-processor that generates ODF. Its coverage of LaTeX is unlimited, but the intermediate format HTML somewhat limits the range of document fragments that can be generated.

Here we present a similar approach, only that we extend the backend of the LaTeXML system to generate DOCX and ODF. Like `latex2rdf` LaTeXML directly parses LaTeX source files, but the coverage of TeXis complete (including macro definitions) and semantics-preserving bindings for the most important LaTeX packages are provided. The main difference to TeX4ht is that LaTeXML generates an XML representation that is structurally near to the LaTeX sources and preserves the author-supplied semantics for further processing.

2 Implementation

Both docx and odt files share a very similar structure and are almost interchangeable, except for slight differences in syntax and different names. They both consist of zipped up XML files. The main content, such as text, placement of images, tables etc., is written in document.xml. The other important file is relations.xml, which contains information about where in the docx/odt file other supplementary files such as images are contained. Finally the archive contains various other objects such as style files, setting files and images.

To create the .odt/.docx files we first transform the .tex file to an intermediate XML-based format using LaTeXML..

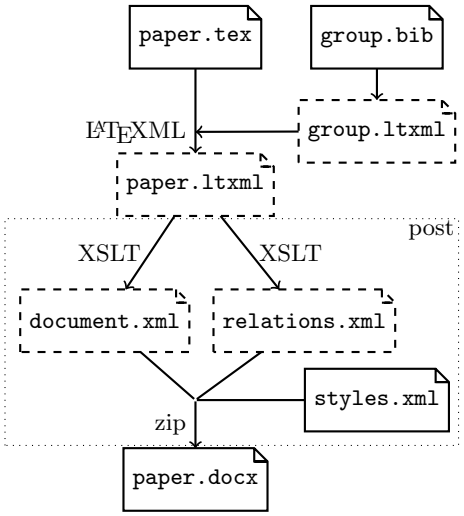

Fig. 2. The Transformation Process

Then we use an XSLT stylesheet to generate document.xml from the .ltmxl file. For Word files, we use a Microsoft stylesheet to transform the MathML generated by LaTeXML to the docx math format. The other file we generate from the ltxml file using XSLT is relations.xml.The other supporting files such as images are placed into the correct file structure the post-processor. As the penultimate step some static files, that don't change depending on the input document, are also placed into the correct directories.The main file of interest here is styles.xml, which contains the style information of the document. We had to create this ourselves to recreate the feel of the PDF files generated by LaTeX. Finally the document is zipped to create the docx/odt file.

[1] [2]

EdN:1
EdN:2

3 Conclusion

We have presented a LaTeXML plugin that transforms LaTeX papers into Word/Office documents in a one-line system call. With the recent web front-end of LaTeXML, it will be simple to extend this to a web service. The LaTeXML Word Processing plugin is public domain and is available from GitHub at [L2O]. The conversion makes crucial use of the fact that LaTeXML preserves more of the document and formula semantics than other systems that process LaTeXdocuments, this ensures that the core process in the transformation – the translation of LaTeXML

[1] EdNote: Screenshot einfuegen
[2] EdNote: Bin eigentlich nicht zufrieden hiermit TT

XML to Office XML (DOCX or ODF) has enough information to generate the respective target document structures.

In the future we want to develop an office package for LaTeX, which allows the direct markup of higher-level structures – e.g. document metadata in LaTeX documents, so that it can be transferred to the office documents. Similarly, we want to extend the transformation to carry over even more semantics from the sTeX format into semantically extended office formats like CPoint or CWord [Koh08]; this would finally give us a way to cleanly interface the currently LaTeX-based document methods in the KWARC group to applied STEM disciplines.

References

[Koh08] Andrea Kohlhase. "Semantic Interaction Design: Composing Knowledge with CPoint". PhD thesis. Computer Science, Universität Bremen, Apr. 2008. URL: http://kwarc.info/ako/pubs/AKo_Promo.pdf.

[L2O] GitHub repository. URL: https://github.com/KWARC/LaTeXML-Plugin-Doc.

[L2R] *LaTeX to RTF converter*. URL: http://sourceforge.net/projects/latex2rtf/ (visited on 01/08/2010).

[T4HT] *TeX4ht: LaTeX and TeX for Hypertext*. URL: http://www.tug.org/applications/tex4ht/mn.html (visited on 01/08/2010).