EuDML
The EUROPEAN DIGITAL MATHEMATICS LIBRARY

ICT PSP
ICT POLICY SUPPORT PROGRAMME
part of the Competitiveness and Innovation Framework Programme CIP

DEMO

Project Acronym:	EuDML
Grant Agreement number:	250503
Project Title:	The European Digital Mathematics Library

D10.2: Accessibility Toolset – Deployment in Live System

Revision: 1.0 as of 15th March 2012

Authors:

Volker Sorge	University of Birmingham
Josef Baker	University of Birmingham

Project co-funded by the European Comission within the ICT Policy Support Programme		
Dissemination Level		
P	Public	✓
C	Confidential, only for members of the consortium and the Commission Services	

Revision History

Revision	Date	Author	Organisation	Description
0.1	9th March 2012	Josef Baker	UB	Submitted first revision to SVN.
0.2	9th March 2012	Volker Sorge	UB	Added conclusions.
0.3	10th March 2012	Thierry Bouche	UJF	Peer review.
0.4	12th March 2012	Krzyś Wojciechowski	ICM	Small corrections, added links.
0.5	12th March 2012	Petr Sojka	MU	Biblio corrections and hyperref setup.
1.0	15th March 2012	Volker Sorge	UB	Added example and final revision.

Abstract
In this document we describe the release of the accessibility toolkit for creating accessible mathematical documents in a wide variety of formats, designed to be compatible with a variety of software. We give a brief overview of the processes and output of the toolkit, together with some initial evaluation.

Contents

1 Introduction

As described in our review of the state of the art, D10.1 [10], the goal of work package 10 is to enable accessibility to the mathematical documents within EuDML for print impaired users. In addition, by improving these documents, we allow all users to interact with the mathematical content in far more ways than available with a standard PDF, for example, copying and pasting, searching for, or reading out loud formulae.

The road map in the review, presented two tools that should be considered for integration into the EuDML work flow to produce accessible content and give added value; the UMCL library and MaxTract. This document presents the integration of one of these tools, MaxTract, together with an initial evaluation, along with our decision for not integrating the UMCL library.

MaxTract is a tool to translate mathematical PDF documents into suitable markup like MathML or LaTeX and is developed by UB. MaxTract has been tested over EuDML content and extended to produce a wide variety of accessible output and has been integrated into the EuDML work flow. The content produced by this tool is not yet available through the EuDML system, as it is a part or release v1.3, so an online demonstration page has been created at `http://www.cs.bham.ac.uk/research/groupings/reasoning/sdag/maxtract.php`. Details of the MaxTract process and output are detailed in sections 2 and 3 and the online interface is explained in section 4.

UMCL is a library that aims at providing Braille transcription for mathematical applications [3]. It was developed at the University Jussieu, Paris, France. It takes mathematical markup in LaTeX and MathML and translates it into several Braille dialects in different European languages. It was tested extensively by MU on the EuDML corpora [8]. As a result the UMCL library was found to produce extremely poor results, primarily due to its pre-processing routine that tries to put the input markup into a canonical form, thereby introducing a host of errors for complicated expressions [2, 8]. After discussions with the developers, who are no longer developing the tool thus unable to fix the bugs we identified, we decided to abandon attempts at integration.

2 MaxTract Process

We use image analysis over an input file rendered to `TIFF` to identify the precise bounding boxes of the glyphs consisting a page. These are mapped to the character and font information extracted via PDF analysis to produce a list of symbols. The extraction is completed by parsing the content streams and font objects comprising a PDF file with a bespoke PDF parser we have written, based upon the PDF specification [1]. A symbol consists of a name, bounding box, base point, font size and font name. Projection profile cutting is then used to identify lines of symbols which are passed to a linear grammar as lists of symbols.

The linear grammar creates a parse tree based upon the two dimensional relationships of the symbols and their appearance. The grammar has been designed to produce a parse tree rich enough to be translated by specialised drivers to produce a wide variety of output in various markup. The extraction process, analysis and grammar are explained in detail in [4, 5].

3 Translation

We use three main output drivers to produce markup, namely a LaTeX, MathML and plain text driver. We combine these drivers in a number of ways to produce various output formats designed to be accessible to users of a wide variety of software. Here we explain the basic drivers.

3.1 Basic Drivers

Each of the basic drivers are used in conjunction with a layout analysis module, which identifies structures such as display mathematics, alignment and justification, columns and paragraphs.

LaTeX This produces a `.tex` file, which when compiled, has been developed to reproduce the original formatting and style closely. All of the fonts identified from the original PDF file are reused, and reproduced together with layout and spacing where possible.

MathML The MathML driver returns an `.xhtml` file, containing standard HTML, interspersed with presentation MathML where appropriate. Unlike the LaTeX driver, styling is not closely reproduced, with standard HTML fonts and formatting used instead of the originals, however elements such as headings and paragraphs are retained.

Festival The Festival driver produces plain text that can be given to text-to-speech engines directly. That is, all mathematical expressions occurring in a document are transliterated in natural language. For example, the expression $x^3 + \pi$ would be translated to "x to the power of three plus pi".

We have particularly focused on and experimented with Festival [6], and have added some optimisation for this particular engine.

3.2 Annotated PDF

The idea of annotated PDFs is to not only reproduce the original document using the LaTeX driver, but also allow the user to view and copy the markup for each mathematical formula when viewing the compiled PDF. By associating each formula with a `\pdfannot` command, which is processed by `pdflatex`, the user is presented with clickable notes, containing the respective markup. These notes can be opened, closed, moved and edited by the user, and of course their content can be viewed and copied. The annotation features are not universally supported by PDF browsers, although they are part of the PDF specification and are supported by Adobe Reader.

LaTeX Annotations A special version of the LaTeX driver is used to produced the LaTeX annotations. As the markup is designed to be viewed, copied and possibly edited, positioning and font commands are removed in order to improve the clarity and simplicity of the generated code.

MathML Annotations These are produced by the same MathML driver as described previously, containing valid snippets of MathML for each formula.

Double Annotations In addition to the singly annotated files we can also produce doubly annotated files, where each formula is associated with both the MathML and LaTeX.

3.3 Layered PDF

By taking advantage of the Optional Content features of PDF, we can create files that contain multiple layers, allowing a user to switch between, say the rendered file and the underlying LaTeX. To achieve this we make use of two additional LaTeX packages, `OCG` [9] and `textpos` [7]. The first one, developed in Brno, is used to produce the separate layers and the second one is used to position each layer correctly. Again these official features of PDF are not widely supported by PDF browsers other than Adobe Reader yet.

Text Layer This layer is produced by using the Festival driver. The layer has been designed to work with the read-out-loud facility in Acroread, allowing the screen reader to accurately read the whole document, including mathematics which is usually garbled in standard documents.

LaTeX Code Layer In a similar manner to the annotated PDF, this allows the user to see and copy the underlying LaTeX code. However, the layer shows the code for the whole page rather than just the formulae. This is the simple LaTeX code, without fonts and spacing commands.

Both Layers Again, we can also produce double layered PDF files containing both the LaTeX and text.

3.4 Accessibility Formats

The accessibility formats that we produce are designed to be compatible with all screen readers. This is achieved by producing standard, plain text files, with none of the special characters or formatting that are often incompatible with accessibility tools. Any non-linear mathematics using special symbols is replaced with alternative ASCII based text.

Text only Produces a text file that can be given to text-to-speech engines directly. The output is similar to that of the text layer we add into layered PDF files.

Text only as LaTeX Text only as LaTeX wraps the text described above with a standard LaTeX header and footer. No other commands are used and any mathematics is replaced by alternative text. When compiled into PDF, this is essentially the same as the text layer from the layered PDF.

Text only as HTML This produces a standard `.html` file to be used with speech enabled browsers. The text is wrapped in HTML with a standard header and footer, and line break is the only tag used. Mathematical equations are replaced by alternative text.

4 MaxTract Online Interface

The MaxTract demonstration consists of an HTML form to select and upload a PDF file for extraction, select an output format and enter an email address. It can be found at `http://www.cs.bham.ac.uk/research/groupings/reasoning/sdag/maxtract.php`.

A PDF file is compatible with MaxTract if it contains only fonts and encodings that are embedded and of type 1. This can be checked by viewing the fonts tab in the file properties within Acroread. Once uploaded, the file is processed if it is found to be compatible with MaxTract, and the user is emailed with a link to download the output. If the file cannot be processed, this will be confirmed via email.

An example of each type of output is also available to be viewed or downloaded from the MaxTract web site. As stated in section 3, some of these formats make use of advanced features of PDF which are not supported by all readers, however they are compatible with Adobe Reader which should be used to view our output.

5 Conclusions

We have deployed the first implementation of our toolkit to provide accessibility to documents in EuDML with an emphasis on giving full access to the mathematical expressions. Although, the translation of documents in EuDML is already available (as part of WP7) the accessible output can not yet be integrated into EuDML, as the current release can not yet serve content. This feature will, however, be included in the v1.3 release. As consequence we have provided an extra demonstrator on a separate web page, which does not only present some example documents but give the additional advantage for users to upload their own documents.

The toolkit is primarily based on the MaxTract extraction tool which can work with digitally born documents. While this means that it is not applicable to all content in EuDML, it will nevertheless provide accessibility to a significant subset of the collection and demonstrate the support a dedicated digital mathematical library can provide. It is hoped that with more advances into full mathematical formula recognition from retro-digitised documents, these can be made amenable to the same techniques in the future.

We have put together a number of different formats that can provide accessibility to mathematical documents. However, we do not expect that all will need to be available in the final version of EuDML. We will now start evaluating the different formats in order to make a final selection. For the evaluation we have already ensured interest from a number of stakeholders: In particular, we aim to be testing with blind mathematicians, researchers in institutes for integrated studies and accessibility support officers that work with print impaired learners in institutions for higher and further education. We are hoping to have a first round of evaluation to narrow down the options for the v1.3 release of EuDML and then test the user interface for the accessibility tools in a second round of evaluation.

References

[1] Adobe. *PDF Reference fifth edition Adobe Portable Document Format Version 1.6*. Adobe Systems, 2004.

[2] Dominique Archambault and Victor Moço. Canonical MathML to Simplify Conversion of MathML to Braille Mathematical Notations. In Klaus Miesenberger, Joachim Klaus, Wolfgang Zagler, and Arthur Karshmer, editors, *Computers Helping People with Special Needs*, volume 4061 of *Lecture Notes in Computer Science*, pages 1191–1198. Springer Berlin / Heidelberg, 2006. `http://dx.doi.org/10.1007/11788713_172`.

[3] Dominique Archambault, Bernhard Stöger, Mario Batušić, Claudia Fahrengruber, and Klaus Miesenberger. A software model to support collaborative mathematical work between Braille and sighted users. In *Proceedings of the ASSETS 2007 Conference (9th International ACM SIGACCESS Conference on Computers and Accessibility)*, pages 115–122. ACM, October 2007. `http://portal.acm.org/ft_gateway.cfm?id=1296864&type=pdf`.

[4] Josef B. Baker, Alan P. Sexton, and Volker Sorge. A linear grammar approach to mathematical formula recognition from PDF. In *Proceedings of the Conferences in Intelligent Computer Mathematics, CICM 2009*, volume 5625 of *LNAI*, pages 201–216. Springer, 2009.

[5] Josef B. Baker, Alan P. Sexton, and Volker Sorge. Towards reverse engineering of PDF documents. In Petr Sojka and Thierry Bouche, editors, *Towards a Digital Mathematics Library, DML 2011*, pages 65–75, Bertinoro, Italy, July 2011. Masaryk University Press. `http://hdl.handle.net/10338.dmlcz/702603`.

[6] Alan W. Black and Paul A. Taylor. The Festival Speech Synthesis System: System documentation. Technical Report HCRC/TR-83, Human Communciation Research Centre, University of Edinburgh, Scotland, UK, 1997. Available at `http://www.cstr.ed.ac.uk/projects/festival.html`.

[7] Norman Gray. Textpos, 2010. `http://nxg.me.uk/dist/textpos/`.

[8] Martin Jarmar. Conversion of Mathematical Documents into Braille. Master's thesis, Faculty of Informatics, January 2012. `https://is.muni.cz/th/172981/fi_m/?lang=en`.

[9] Robert Mařík. Ocgtools, 2012. `http://ctan.org/pkg/ocgtools`.

[10] Volker Sorge, Mark Lee, Petr Sojka, and Alan P. Sexton. State of the Art of Accessibility Tools, February 2011. Deliverable D10.1 of EU CIP-ICT-PSP project 250503 EuDML: The European Digital Mathematics Library, `https://www.eudml.eu/sites/default/files/D10.1_0.pdf`.