Mmtex: Creating Mathematical Content for eLearning

Nina Dahlmann[*], Sabina Jeschke[*], Tilman Rassy[*], Ruedi Seiler[*]

[*]Berlin University of Technology, Faculty II – Mathematics & Natural Sciences, 10623 Berlin, Germany

In the past years the XML-based standard MathML has enhanced the professional presentation of mathematical formulas on websites significantly. Especially for modern eLTR environments (eLTR := eLearning, eTeaching, eResearch) this development was an important step since it allows creating reusable, adaptable mathematical content. Nevertheless, the editing process for formulas that are encoded in MathML still is a complex task which has to be supported by suitable authoring environments. These editing tools have to be based on common standards as e.g. LaTeX which are not only accepted by the worldwide mathematical community but furthermore have to be capable to convert the input into a file format strictly separating content and layout.

The Mmtex-converter discussed in this article is a text/command line based environment that has been designed as an editing tool for the interactive, multimedia-based eLearning platform Mumie, focusing on the learning and teaching process in the university education of mathematics. It aims at easily creating XML-documents that include formulas, pictures, applets and control elements for interaction by using a simple LaTeX dialect.

Keywords: MathML; LaTeX; mathematics; reusability; eLearning; semantic encoding; authoring environment

1. Creating mathematical contents – Demands

The basic design of the Mmtex-converter is closely connected to the pedagogical and technological concept of the eLearning environment Mumie [5], [1]. An editing environment that allows creating interactive, reusable contents has to be developed under the following aspects:

- **Development of eLearning contents with LaTeX**
  Among mathematicians LaTeX presents a well accepted standard for the creation of mathematical texts. It therefore provides a suitable basis for a tool designed to develop mathematical eLearning contents. LaTeX allows encoding and presenting mathematical expressions of any complexity and is, after an adequate time of practice, much easier to use than most drag & drop formula editors following a "what-you-see-is-what-you-get"[1] philosophy. So far, one of the problems of standard LaTeX, which it actually shares with most other editing environments for mathematical content, is the lack of fully semantical encoding: Semantical descriptions are necessary to offer functionalities as e.g. connecting formulas with computer algebra systems, searching within formulas or adapting the presentation to the different users. To create semantically enriched markup languages for the web (Content MathML, W3C standard) LaTeX serves as a basis but has to be adapted to the new target. Additionally, standard LaTeX does not implement all features necessary for a modern eLearning environment and therefore has to be extended accordingly (e.g. annotating metadata, or integration of interactive components, see below).

- **Integration of interactive elements**
  One of the most important characteristics of computer-based learning is the possibility to implement and connect different, especially interactive content elements. This allows to realise individual, explorative learning environments which are considered to be one of the most effective and sustainable ways of approaching complex topics. Mumie is developed to support these learning styles and therefore re-

[1] LaTeX dialects are currently developed to support a more strict enhanced description of mathematical content (sTex).

Corresponding author: Tilman Rassy, e-mail: rassy@math.tu-berlin.de , Phone: +4930 3147 9291, Fax +4930 3142 4413

lies on contents, where interactive elements, like applets, controllable animations or visualisations can be easily integrated [4].
A concept has to be developed that allows integrating interactive objects into eLearning contents. Since the original LaTeX is directed towards printable media-types and thus does not support formats like applets or animations, a LaTeX-based converter has to implement suitable extensions. These extensions have to take into account that interactivity also implies that contents can be connected with each other via links, a feature that is essential for an Internet-based eLearning-environment.

- **Modular design, development of granular reusable content**
  Regarding the rapid development in all technology-related fields, a modular software design ensures that new features can be implemented on an acceptable time scale without having to redesign the whole software. This also implies the development of standardised interfaces allowing third-party software to be integrated into the editing environment. A modular design also allows users of the converter to extend it according to their specific needs by adapting only parts of the software. The philosophy of modularity has also to be applied on the design of the content itself: The contents for Mumie consist of fine-granular knowledge pieces, as e.g. definitions, theorems or examples which can be easily connected to complete courses specifically adjusted to different learning groups or -targets. This concept demands "context-neutral" knowledge pieces to ensure their broad reusability: teachers use the content base like a library only having to choose the contents they need for their specific courses. In order to preserve this flexibility, it is necessary to strictly separate content and layout which can be realised optimally by using XML/XSL-technology in an appropriate manner for the encoding of the contents and its delivery respectively[2]. Therefore since LaTeX itself can not be displayed by commonly used browsers, an editing environment has to convert the LaTeX content into XML/XSL, using the MathML standard for presentation of mathematical symbols.

2. Mmtex – The concept

Mmtex is a command-line based converter designed as an editing tool for the interactive, multimedia-based eLearning platform Mumie. Currently, Mmtex can convert two different LaTeX dialects; one is very similar to standard LaTeX, with the limitation that a few commands and environments are not implemented. The second dialect is a very specialised one, developed for the contents of the Mumie-eLearning-platform, focusing on the creation of mathematical texts. With both dialects, non-mathematical parts are converted to a particular XML, whereas mathematical parts are converted to MathML. With the appropriate stylesheets, this XML then can be transformed into the desired output format (e.g. XHTML, with the formulas coded in MathML).

Mmtex is written in Perl, since this programming language is not only capable to conveniently handle complex regular expressions, which are needed especially for the parsing process, but is furthermore easily extensible. The Mmtex converter is designed modularly (see Fig. 1). No commands or environments except the '[]' command are hard-coded in the core modules. Commands, environments, and the resulting XML are defined in *document classes* and *Mmtex libraries*, which are implemented as Perl code files. Due to this design none of the core modules need to be updated if a new LaTeX dialect is implemented – they are completely independent of the LaTeX dialect. All information concerning the actual transformation is contained in the code files implementing the document classes and libraries. Thus, Mmtex can easily be customised and extended by writing new document classes and libraries.

The actual converting process consists of the following, recursively called procedures:

1. **Resolving into tokens:**
   While reading the document source, Mmtex parses the input and breaks it up into tokens. This is done by the cooperation of two modules: *Scanner* and *Parser*. The first one is more low-level and provides

[2]XML/XSL also allows to use the W3C standard MathML for the presentation of mathematics which is supported by different browsers as well as mathematical software as Maple or Mathematica where it acts as a data exchange format.

generic routines to analyse a character stream, while the latter one is more high-level and provides specific implementations for the generic constructs of the first.

2. **Processing tokens:**
   Tokens can have different *types*. For example, a backslash followed by alphabetic characters is a token of type "cmd", i.e., a command name. For each token type a so-called *token handler* is registered. Most of the token handlers are implemented in the *Parser*; however, special purpose tokens may be added by the document class or a library. As soon as Mmtex has parsed a token and identified its type, the corresponding token handler is called, which is responsible for the actual conversion. The handler for command tokens, for example, looks up the command in the so-called *command table*, gets the number of mandatory and optional arguments from there, asks the Scanner/Parser to parse the arguments, and calls a function that performs the conversion. The latter is specific to the command and also retrieved from the command table.

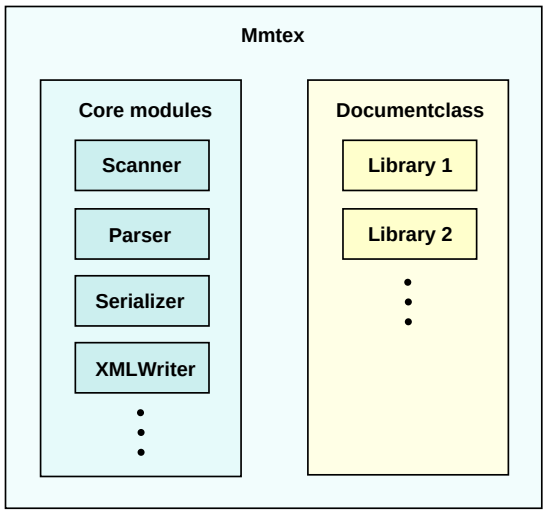


Fig. 1: Overview – the Mmtex components.

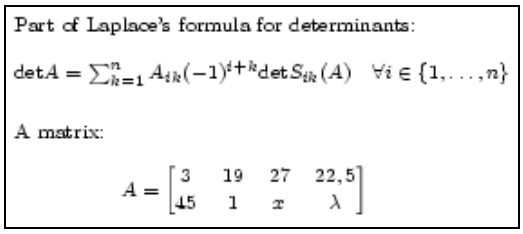
3. **Writing output:**
   The final XML output is written to a file, but none of the XML-generating functions accesses this file directly. Rather, they call generic functions of the *Serializer* module. These functions notify the Serializer of the beginning of an XML element, the end of an XML element, etc. (quite similar to SAX parsers). The Serializer, in turn, calls methods of the *XMLWriter* module to write the XML output.

For the XML designed for the Mumie-platform, Mmtex provides XSL-stylesheets that can be used to transform the XML-output of the converter into a static XHTML web page where the formulas are presented as MathML. Within the Mumie-project, these XSL-stylesheets are used to provide a preview functionality, allowing the author of mathematical contents to easily check the output of the newly created or modified documents.

Thus, a page which contains the following code

```
\documentclass[sloppy]{japs.subelement.remark}
%here additional metadata
\begin{content}
  \title{Example}
  Part of Laplace's formula for determinants:
  $\text{det}A =
  \sum_{k=1}^n A_{ik} (-1)^{i+k}
  \text{det} S_{ik}(A) \quad
  \forall i \in \{1, \ldots, n\} \quad$
  A matrix:
   \[A=\begin{pmatrix}
     3  & 19 & 27 & 22,5   \\
     45 & 1  &  x & \lambda
   \end{pmatrix}\]
\end{content}
```

results in the following XHTML page:


Part of Laplace's formula for determinants:

$\det A = \sum_{k=1}^{n} A_{ik}(-1)^{i+k}\det S_{ik}(A) \quad \forall i \in \{1,\ldots,n\}$

A matrix:

$$A = \begin{bmatrix} 3 & 19 & 27 & 22,5 \\ 45 & 1 & x & \lambda \end{bmatrix}$$

Fig. 2: XHTML page as rendered by Firefox/Mozilla (screenshot).

The Mumie-specific LaTeX dialect (Mumie-TeX) implemented into Mmtex also offers code-environments for the integration of additional metadata, as e.g. authors, names, and descriptions of the documents. Additionally, Mumie-TeX provides commands which can be used to insert interactive com-

ponents and visualisations. An applet, for example, that is stored either in a database or on the file system, can thus be linked easily within a document.

Besides the actual converting process of elements, developing contents for an eLearning environment also includes tasks like managing a rapidly increasing amount of documents and administering different versions of elements. Cooperation between different authors has to be supported in order to develop high-quality contents for different media types [2]. Furthermore, it is necessary to offer an overview of existing contents and allow authors to use and define templates for e.g. recurring or frequently used LaTeX documents.

For the creation of the contents of the Mumie-project, the Mmtex-converter has been embedded into an (so far also command-line) environment called Mmcdk (Multi Media Content Development Kit). Mmcdk was developed as a Java-based shell environment[3] which is designed to ease creating, editing and managing the contents.

Mmcdk therefore implements the following functionalities and features:

- Access to Mmtex:
  The Mmtex-converter can be accessed easily from within the Mmcdk. Additionally, the Mmcdk allows to convert either all existing contents in one step or alternatively to specify arbitrary subsets of the content files for conversion, also generating the according previews.
- Navigation & search functions:
  Since the content of the Mumie-platform is structured fine-granularly, the number of the "knowledge pieces" is high and increases extremely rapid. To guarantee that an author can easily find elements related to specific topics, the content of the Mumie-platform is stored in a specific matter ontology. This allows authors (usually experienced mathematicians) to intuitively navigate within the elements (for further information see [3]). Mmcdk's navigation mechanisms support this structure and additionally offer options for searching the existing content for keyphrases.
- Presentation & Overview:
  Mmcdk offers the possibility to get an "explorer-like" overview of all existing elements in a browser, including a preview of the resulting XHTML-page of each document, the attached meta-information about the authors, its status and its LaTeX source-code.
- Templates:
  For the fine-granular "knowledge pieces" of the Mumie-platform, templates were designed where the basic MumieTeX environments for definitions, theorems or remarks are already set, allowing the author to concentrate on the creation of the "mathematical" content. Within the Mumie-platform, these templates also lead to a uniform usage of the different content elements, thus assuring a homogenous presentation.
- Integrating Mathletfactory:
  Although Mmcdk has not been developed to substitute a development environment for applets, it integrates the so-called "Mathletfactory", a framework for the creation of applets with mathematical content (see [7]). These applets can be edited and compiled within Mmcdk, linked with documents and previewed in a browser.

Currently the Mmcdk-prototype operates on contents stored on a file system, but the original design of Mmtex is aimed at accessing contents of a database. The development process is progressing rapidly and this feature will be implemented into Mmcdk within the next months. Based on a suitable database design, the Mmcdk can offer features like an improved privilege management and an adequate version control system, both necessary for e.g. creating contents cooperatively.

3. Conclusion & Perspectives

[3]It is also planned to develop a graphical interface for Mmtex in the near future.

Comparing the demands developed in section 1 with the features that are already implemented in Mmtex, the only significant lack of Mmtex in its current version is the missing support for a semantical encoding of the content. Since the components of the software are capsuled into different modules, the desired functionalities can be added as soon as an appropriate standard for the semantic encoding of mathematical expressions and contents is developed. Presently, there are two important approaches for the semantical encoding of text-based mathematical content:

MathML Mathematical Markup Language [8] has been developed by the W3C in cooperation with industry partners mainly from the field of computer algebra systems. Designed as an XML application, MathML can be used to encode both the presentation of mathematical notation for high-quality visual display ("MathML Presentation Markup"), and mathematical content, for applications where the semantics plays more of a key role such as scientific software or voice synthesis ("MathML Content Markup"). The coverage of MathML language is insufficient for scientific learning, teaching and research scenarios (oriented towards the so-called "K–12"-standard) but can be extended by so-called content dictionaries which are still under development.

OpenMath [6] is designed as an – again XML-based – semantical meta language of mathematics, focusing on the communication between different system and software components. The presentation of mathematical objects plays a subordinate role (integration of Presentation MathML solves this problem). The development process has mainly been driven by the academic community, in particular by the DFKI Saarbrücken (German Research Centre for Artificial Intelligence) and its partners. The semantics of mathematical objects is given by content dictionaries. So-called "phrase books" act as interfaces between different software applications and the OpenMath language: their task is the translation of OpenMath into an appropriate software representation. Compared to MathML, OpenMath possesses the greater potential but is less well known so far.

As soon as the development of one of the here discussed standards (or perhaps a completely new approach) has reached a level where it can be implemented into the creation of mathematical contents, Mmtex will be adapted accordingly.

Note: Regarding the rapid development of technology, a sustainable, flexible eLearning-platform has to implement modular concepts of software integration and portal technologies. The Mumie-platform implements open, standardised interfaces to enable the integration of additional functionalities, provided by third-party software. In order to allow adapting Mumie to different proposes and to ensure the platform's further development, Mumie and all its components (in particular Mmtex, Mmcdk) are Open Source software, licensed under GPL.

Acknowledgements: The support by the BMBF (Bundesministerium für Bildung und Forschung) for the Mumie-project is greatly acknowledged.

References

[1] N. Dahlmann, S. Jeschke, R. Seiler, and U. Sinha. MOSES meets MUMIE: Multimedia-based Education in Mathematics . In Freddy Malpica, Andrés Tremante, and Nicoletta Sala, editors, International Conference on Education and Information Systems: Technologies and Applications, Orlando, Fla., 2003.

[2] T. Hampel, R. Keil-Slawik. sTeam: Structuring Information in a Team - Distributed Knowledge Management in Cooperative Learning Environments. ACM Journal of Educational Resources in Computing, 1(**2**), 2002.

[3] S. Jeschke. Mathematics in Virtual Knowledge Spaces, CIT-Structures and IT-Technologies in Teaching and Research. PhD thesis, Berlin University of Technology, Berlin, April 2004.

[4] S. Jeschke, M. Kohlhase, and R. Seiler. eLearning-, eTeaching- & eResearch-Technologien - Chancen und Potentiale für die Mathematik. DMV-Nachrichten, Juli 2004.

[5] Mumie community. Mumie. http://www.mumie.net .

[6] OpenMath Society. OpenMath. http://www.openmath.org .

[7] T. Paehler. Design, Implementation and Application of a Reusable Component Framework for Interactive Mathematical eLearning Sites. PhD thesis, RWTH Aachen, Aachen, March 2005.

[8] World Wide Web Consortium (W3C). Mathematical Markup Language (MathML). http://www.w3.org/Math , March 2001.