

# Creating and Publishing Courseware Independent of the Authoring System

Martin Rotard and Lutz Finsterle

Visualization and Interactive Systems Institute  
University of Stuttgart  
Universitätsstraße 38  
70569 Stuttgart  
Germany  
rotard@vis.uni-stuttgart.de

Institute of Communication Networks  
and Computer Engineering  
University of Stuttgart  
Pfaffenwaldring 47  
70569 Stuttgart  
Germany  
finsterle@ikr.uni-stuttgart.de

**Abstract:** The effort for creating content for e-learning applications is high. For authors it is more comfortable to keep their well known authoring system they are familiar with. In this environment they could also reuse existing content. By using modern web standards for presentation and an XML-based courseware format we meet future requirements and are platform independent. We propose a tool chain based on OpenOffice.org, which allows to generate standard conform courseware and how it can be adapted and delivered in an user centered way.

## 1 Motivation

While e-learning is an ever faster evolving part in today's teaching, real integration of commonly used authoring tools is not very well supported. Reasons for this are that these tools have specific storage formats and also are not designed to support „document semantics“ but are layout driven. With our tool chain we are able to support metadata standards [LOM, SCORM, Dublin Core] by still leaving the authors favorite tool available for sake of their productivity. To avoid typing hundreds of pages from scratch in a new authoring tool it is necessary to find a method to reuse the existing content. To keep the authoring system they are used to is comfortable for the authors. To support common authoring tools in creating e-learning content we developed a tool chain for [OpenOffice.org] (Figure 1), because it provides a lot of functionality which made our development easy. But our solution is not restricted to OpenOffice.org. We used it because it supports many import filters, modern standards like XML-, MathML- and Scalable Vector Graphics output, macro scripting and external scripting by the UNO interface for Java [MathML, SVG, UNO]. With the same tool chain, we can support other authoring systems like Microsoft Word, Adobe FrameMaker, etc. in future.

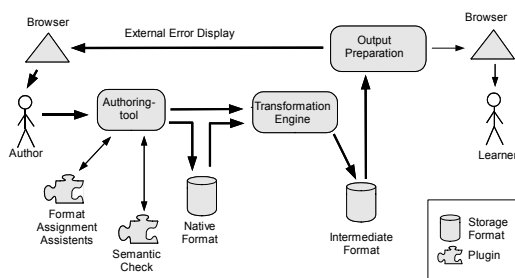


Figure 1: Tool chain

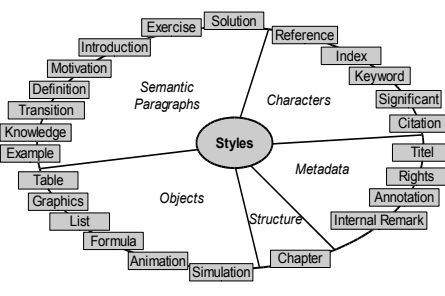


Figure 2: Paragraph and text styles

Related work is done in this area by a group of the Open University of the Netherlands that defined the Educational Modelling Language [EML], the FIGARO Project, which uses OpenOffice.org as authoring Tool [Oeltjen 2003] and the Humboldt-University Berlin that transforms OpenOffice.org documents to XDiML (DissertationMarkup-Language) [Henneberger et al. 2003].

## 2 Creation of Courseware in Authoring Tools and User Centered Support

In the project Information Technology Online [ITO], which is initiated by the German Federal Ministry of Education and Research, we do not focus only on the authoring tools issues, but also want to support the

reuse and exchange of those materials. Therefore the primary way would be to agree on some common format for those materials that are to be exchanged. Hence we defined an XML based intermediate format described by an XML-Schema definition to be independent of the specific storage format of the authoring system (see Section 3.1). To specify the semantic category of a paragraph or text phrase we defined paragraph and text styles (Figure 2) in the authoring system. In the transformation engine these categorized paragraphs and text phrases are mapped to XML tags of the intermediate format. In our process the paragraph and text styles in the authoring system do not express layout specification, but the semantic categorization. An example for the usage of the paragraph and text styles is a definition paragraph for a specific keyword. The author has to apply the “definition style” to this paragraph and the “keyword style” to the keyword.

After the transformation into the intermediate format the paragraph in the output for the learner could have a special style, which is independent to the layout of the paragraph in the authoring tool. E.g. a colored border could surround the definition paragraph and the keyword can be emphasized in the output for the learner. In the case of the definition paragraph this structural categorizing of the author makes it possible to create an index and glossary automatically. Furthermore we defined some styles in the authoring system to specify metadata information. This makes a semiautomatic metadata input possible, e.g. the copyright for a illustration can be specified in the paragraph directly following by the illustration. Additionally it is possible to enter some alternative text for blind or visually impaired people in an “annotation” styled paragraph following by the illustration.

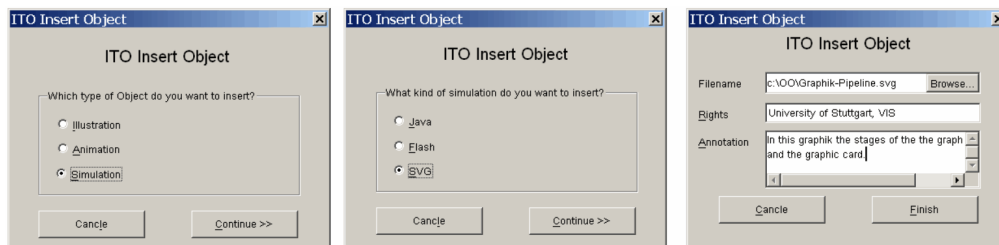


Figure 3: Wizard to insert objects in the authoring system

We have made good experiences in training our authors using a FAQ document (frequently asked questions) which has many examples in it. Additionally we created an ITO menu and some wizards to make the handling with the styles in the authoring system even more comfortable (Figure 3). This also hides the complexity of using paragraph styles from the user, thus making it easier to work with the system. To give the author feedback about the structural and semantic errors in the document (e.g. a “solution” without a “exercise” in the paragraph before) we created a special external output. In this output information, warnings and errors are displayed aside of the text. Our plan for a future version is to display these errors directly inside the authoring system, again without disturbing the layout of the document itself.

### 3 Transformation and Intermediate Format

The transformation engine described in this section is designed to support XML based input and output formats. The currently existing version focuses on reading “OpenOffice.org Writer” files and transforming them to an intermediate courseware format defined within the ITO-Project. This format, which is described in more detail in Section 3.1, has been carefully designed to be flexible, adaptable, and standard-conforming but yet not too complicated.

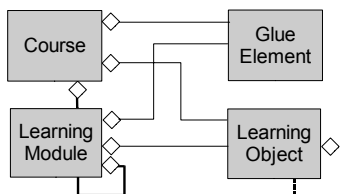


Figure 4: Structure of a course

```
<?xml version="1.0" encoding="UTF-8"?>
<document xmlns="http://ITO/Zwischenformat/ (...)"
<motivation>In this lecture you will learn (...
  <char:text type="keyword">HCI</char:text>
</motivation>
<example>Here is an example of (...</example>
<chapter><meta:title>Intro to HCI</meta:title>
  <knowledge>The user center design process (...
(...)
```

Figure 5: Intermediate format sample

When the rules described in Section 2 are obeyed, the OpenOffice.org files do contain all necessary information to generate the output automatically. We are able to extract and differentiate between content, layout and metadata information and store them either in one document again or directly divided into smaller bits like chapters (called learning modules) and paragraphs (called learning objects) and their metadata and structure. The design of this transforming engine is shown in Section 3.2

### 3.1 The Intermediate Format

The intermediate format is designed to fulfill the requirements for modularized courseware that came up during the design phase of LOM/SCORM and other standards [Koper 2001]. The format therefore does define the necessary structure and data types to store the content and its supporting information, but it does not define a format for metadata on its own. Here DC [DublinCore] it used, but can be extended to LOM, SCORM, etc. in future versions. The main focus is to support differently sized learning grains which are identifiable by use of their learning domain as well as their semantic meaning. This is useful for semiautomatic metadata generation within the transformation process. The structure and hierarchy of the format you find outlined in UML in Figure 4. The format itself is defined by a set of modularly designed XML-Schema files [Finsterle et al. 2003] (see Figure 5).

### 3.2 Transformation Engine Design

The transformation process is designed like a filter chain. The pluggable filters are either XSLT-based or a set of Java classes working on an representation of the XML document structure. To be able to read and write arbitrary data structures, directory structures, or compressed archives, the first and last modules in this chain implement an open resource access component, which enables us at the moment to transparently work on files, directory structures, URL's and packed archives. Additionally to the XML transformation architecture, we found it to be vital to support the transcoding of different media types, too. These can also be plugged into the filter chain, most commonly at the output, for scanning the produced document and process the embedded or linked resource accordingly. The transcoding modules are needed to support for example Browsers on the user side that are not able to display SVG or MathML. Still, the main work has to be done in the transformer library, which is called from the transformation manager. Here the major structure transformations are performed and error handling and semantic checks are done. Additionally to the transformation of the OpenOffice.org writer documents we have realized a transformation of Microsoft PowerPoint and OpenOffice.org Impress slide sets to SVG format [Finsterle et al. 2003].

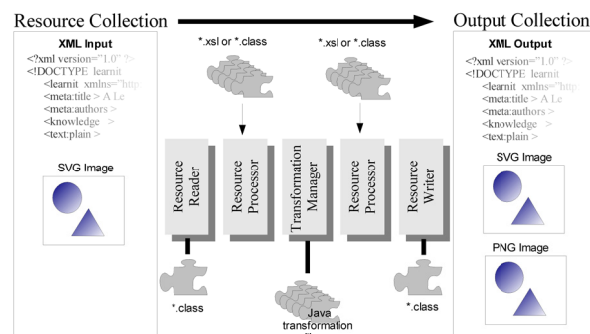


Figure 6: Transformation Engine Design

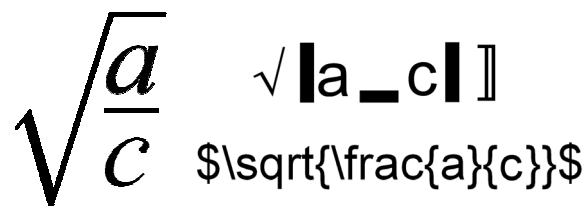


Figure 7: A Mathematical expression rendered in our presentation engine in MathML, SMFB, and LaTeX

Having this XML output, there are certain open issues when it comes to the delivery to the user. Here we have to take into account, that not all users are equal and therefore need to have their specialized versions. Section 4 gives you a brief introduction to the applied techniques.

## 4 User Oriented Delivery

In the project ITO we developed a presentation engine for a virtual learning environment. Therefore we used Apache Cocoon and JetSpeed for the Tomcat web server [Cocoon, JetSpeed, Tomcat]. An XSLT stylesheet is used to transform the XML content of the intermediate format into XHTML in our presentation engine. Additionally we integrated another XSLT stylesheet which presents the XML content in a special version for blind and visually disabled people. In the XHTML output the paragraphs are labeled with the paragraph type instead of using special style e.g. a definition paragraph is labeled with "Definition:" instead of a surrounding box.

In the intermediate format mathematical expressions are represented in MathML. This provides the possibility to support a special output for blind and visually impaired people. For the processing of documents Cocoon uses a pipeline concept. This makes it possible to process more than one XSLT stylesheet in sequence. As a result the users can configure their preferred mathematical output notation in the virtual learning environment.

For the transformation of mathematical expressions we integrated two XSLT stylesheets in this engine: A stylesheet for the transformation of mathematical expressions from MathML into SMFB (Stuttgart Mathematical Notation For the Blind) [Rotard et al. 2003, Schweikhardt 1980], a compact mathematical notation for the blind and a stylesheet to transform from MathML into LaTeX [Yaroshevich], since LaTeX becomes more and more important as an alternative mathematical notation for the blind. You can find an example of a square root of a fraction in Figure 7.

## 5 Conclusions and Outlook

In this paper we presented a concept for creating courseware by use of traditional authoring systems, as well as for the delivery of those modules and courses according to user preferences. We have achieved this by two major aspects: First we have proposed the usage of paragraph and text styles in the authoring system, which do not express layout specification, but the semantic categorization; second we have contributed the transformation engine and the user-oriented outcome. Hereby we presented a flexible engine which can be connected to different learning platforms. It is planned for the near future to connect it to the metacoon virtual learning environment [Medienquadrat] as well as the MTU database [mmdb-tu] designed by the University of Essen. We are also looking forward to transform upcoming documents from the Microsoft Word 2003 XML format to our intermediate courseware format.

*The German Federal Ministry of Education and Research BMB+F funded this work.*

## References

- [Cocoon] The Apache XML Project: *Cocoon*, <http://xml.apache.org/cocoon/>, 25.07.2003
- [Dublin Core] Dublin Core Metadata Initiative: *Dublin Core*, <http://dublincore.org/>, 25.07.2003
- [EML] Educational Modelling Language: *Learning Networks*, <http://eml.ou.nl>, 25.07.2003
- [Finsterle et al. 2002] Finsterle, L.; Rotard, M.: *Mit konventionellen Autorensystemen zum E-Learning Portal*. In: K.P. Jantke, W.S. Wittig, J. Herrmann (ed.), *Tagungsband 10. Leipziger Informatik Tage*, S. 111-121, Akademische Verlagsgesellschaft Aka GmbH, 2002
- [Finsterle et al. 2003] Finsterle, L. ;Rotard, M.: *Automated Format Transformation for Courseware*, OpenOffice.org Conference 2003, <http://marketing.openoffice.org/conference/thursday.html>, 25.07.2003
- [Henneberger et al. 2003] Henneberger, S.; Schulz, M.: *Save as XDiML, Writing and Converting digital Theses and Dissertations using OpenOffice.org*, OpenOffice.org Conference OOoCon 2003, <http://marketing.openoffice.org/conference/>, 25.07.2003
- [ITO] Information Technology Online (ITO): *Information Technology Online Homepage*, <http://www.ias.unistuttgart.de/ito/index.htm>, 25.07.2003
- [JetSpeed] The Jakarta Project: *JetSpeed*, <http://jakarta.apache.org/jetspeed/>, 25.07.2003
- [LOM] IEEE Learning Technology Standards Committee (LTSC): *IEEE P1484.12 Learning Object Metadata Working Group*, <http://ltsc.ieee.org/wg12/>, 31.05.2003
- [Koper 2001] Koper, R.: *Modeling units of study from a pedagogical perspective, the pedagogical meta-model behind EML*, Educational Technology Expertise Centre, Open University of the Netherlands, June, 2001, <http://eml.ou.nl/introduction/articles.htm>, 25.07.2003
- [MathML] W3C: *MathML- Math Home*, <http://www.w3.org/Math/>, 25.07.2003
- [Medienquadrat] Medienquadrat: *Das Projekt medienquadrat*, <http://www.uni-weimar.de/~m2/>, 25.07.2003
- [mmdb-tu] Multimedia Database Technology Education, <http://www.mmdb-tu.de/>, 25.07.2003
- [Oeltjen 2003] Oeltjen W.: *OpenOffice.org as an Authoring Tool in the FIGARO Project*, OpenOffice.org Conference OOoCon 2003, <http://marketing.openoffice.org/conference/>, 25.07.2003
- [OpenOffice.org] OpenOffice Organisation: *OpenOffice.org Homepage*, <http://www.openoffice.org>, 25.07.2003
- [Rotard et al. 2003] Rotard, M.; Bosse, K.; Schweikhardt, W.; Ertl, T.: *Access to Mathematical Expressions in MathML for the Blind*. In C. Stephanidis, editor, *Universal Access in HCI*, volume 4. Lawrence Erlbaum Associates, 2003
- [Schweikhardt 1980] Schweikhardt, W.: *A Computer Based Education System for the Blind*, in: Lavington, S. H. (ed.), *Information Processing 80*, pp. 951-954, North Holland Publishing Company, 1980
- [SCORM] Advanced Distributed Learning: *Sharable Content Object Reference Model (SCORM)*, <http://www.adlnet.org/>, 25.07.2003
- [SVG] W3C: *Scalable Vector Graphics*, <http://www.w3.org/Graphics/SVG/>, 25.07.2003
- [Tomcat] The Jakarta Project: *Tomcat*, <http://jakarta.apache.org/tomcat/>, 25.07.2003
- [UNO] OpenOffice.org: *UNO Development Kit Project*, <http://udk.openoffice.org/>, 25.07.2003
- [Yaroshevich] Yaroshevich, V.: *XSLT MathML Library*, <http://www.raleigh.ru/MathML/mmltex/>, 25.07.2003