Improving Web-Based Training Using an XML Content Base

Simon Wiest
University of Tübingen
Wilhelm-Schickard-Institute for Computer Science, Sand 1
72076 Tübingen, Germany
wiest@informatik.uni-tuebingen.de

Andreas Zell
University of Tübingen
Wilhelm-Schickard-Institute for Computer Science, Sand 1
72076 Tübingen, Germany
zell@informatik.uni-tuebingen.de

Abstract: This paper proposes the application of a structured content base to Web-Based Training settings and introduces a workflow using an XML-based file format for educational data. Content is authored in a SGML/XML editor employing a modified DocBook DTD that caters for semantics needed in educational scenarios. Using a Java servlet, this content is formatted according to learners' preferences on-the-fly from XML files deployed on a web server. Additionally, all user requests are logged with their parameters into a database, supporting statistical usage evaluation.

Introduction

Interactive Computer-Based Training (CBT) has experienced a tremendous growth over the last few years. However, the rapidly growing demand for interactive CBT revealed that the traditional content creation workflow has serious shortcomings that make its large-scale application difficult: Reuse and exchange of content modules are hard to implement, because most learning environments use proprietary authoring tools and data formats. It soon turned out that also HTML, *the* common standard on the Internet, lacks much of functionality to fulfill many of the needs of online educators, e.g. no dedicated semantics for online exercises, no support for complex equation rendering, difficult reuse of HTML content in new contexts. This paper presents a workflow that tries to solve some of these problems using an non-proprietary content structure and a web server extension that enables HTML browsers to access the features made possible by XML content.

Why structuring educational content?

Structured educational material decouples style from content. Storing course content in elements that reflect its semantics (e.g. introduction, example, quiz, ...) allows the reuse of data in different situations and formats - adapted to learner preferences, target media or purposes. All groups concerned with the online learning process benefit from the use of structured documents:

Authors

- Authors can concentrate on content not (necessarily) on style and formatting
- Structural integrity can be checked/enforced by an authoring application (e.g. link checking)
- Indexes, summaries, glossaries etc. can be generated automatically
- Content modules can be reused in other contexts (e.g. related courses, in different skill levels)

Publishers / Internet Service Providers

- Content is well machine-processable (i.e. easily stored in databases to facilitate content administration and improvement of retrieval performance)
- Crossmedia publishing is possible from one, single source (e.g. online/offline versions, printed material). The concept of *viewgroups* presented in (Müller & Deponte 1999) addresses this idea.
- Versioning is facilitated.

Learners

- Mandatory structure facilitates navigating the course content.
- Advanced search and retrieval of content components.
- New features like multi-representation content become feasible (e.g. the same content in both visual and textual representation. (Ram et al. 1999) suggests a Presentation Markup Language.)

Researchers / Quality Assurance

- Enhanced possibilities for learner tracking and usage evaluation allow the collection and sharing of comparable data on applicability and effectiveness of the content and help to improve its quality.
- Flexible rendering allows experiments with screen designs, navigation concepts etc.

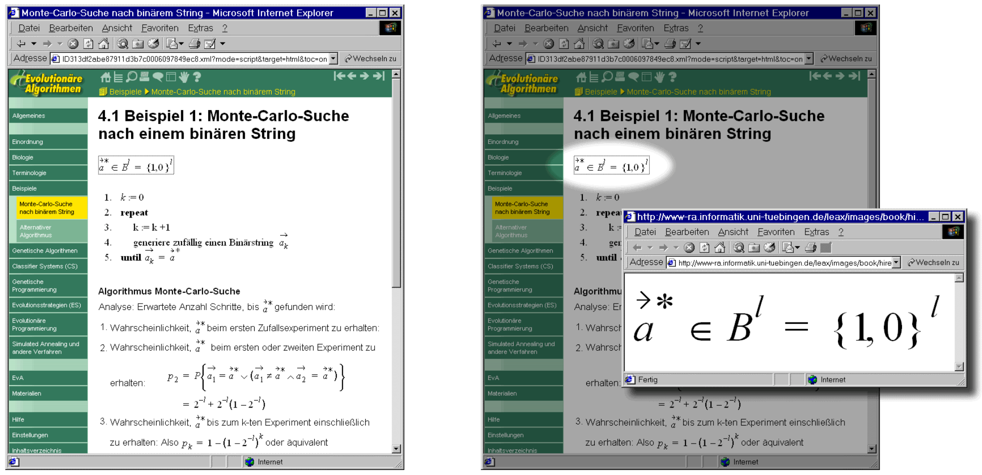
Current activities employing stuctured content in the educational field

XML (Bray et al. 1998) is, like its predecessor SGML (ISO 1986), a scheme to define new languages. It does not specify what kind of application data is stored in a file by itself but rather provides a framework to define purpose-depended languages that follow one common syntax. This is done in a document type definition (DTD) that describes the overall structure and the components of generic data file. There have been already different suggestions to apply SGML, XML or at least structured documents to educational purposes:
The Tutorial Markup Language (Brickley 1998) provides a language to describe several classes of online exercises like multiple-choice tests, drag & drop puzzles etc. IMS is currently defining a specification for standarized questions and test data formats (IMS 2001). Other structured file formats (Krauße 1999, Arneil et al. 1999) pursue the same goal but currently don’t use neither SGML nor XML.
The IEEE Learning Technology Standards Committee proposes a learning object meta-data standard (LOM) (IEEE 2000) to facilitate the retrieval, reuse and exchange of educational content. Learning objects range from small components like a single sentence, a figure or a Java applet up to complete on-line courses. LOM contains information on technical, didactical and legal issues of educational content and provides support for versioning. LOM does not standardize the actual internal data structure of learning content.
The recently released specification for content packaging by the IMS (IMS 2000) has also bindings to an XML file format. There are already commercial applications like Microsoft’s LRN project that are strongly influenced by these specifications (Microsoft 2001).

Our implementation

We offer several online courses on computer science topics to our students at our department. This material consists typically of some 200 linked HTML pages per course, enriched with additional interactive elements like Shockwave animations, Flash movies, and Java applets. The courses were created from legacy content in Adobe FrameMaker format as a starting point. Dealing with mathematical and statistical subjects the courses contain over 500 equations. A typical page is shown in (Fig. 1).

Although Adobe FrameMaker ships with HTML/XML export functionality, files generated by the HTML export function needed manual clean-up which took one person about 1-2 days to complete for each new release of a course. Additionally, an evaluation among student users turned out that the poor display quality of equations was a major annoyance to learners. From our previous experience we also could identify two main usage scenarios for our online content: the *tutorial mode*, in which learners tend to navigate through the course linearly and slowly paced and the *script mode,* in which users want to look up small amounts of information like definitions, proofs or concepts quickly. Because both modes are related representations of the same content, we wanted to keep maintaining course content simple and generate both versions from one single data source. Because interface design for WBT is still a young discipline we also realized that there might be a need for additional modes in the future. Ideally, a learning environment would therefore apply formatting and user preferences to the desired content not until the moment of an incoming learner's request.


Monte-Carlo-Suche nach binärem String - Microsoft Internet Explorer
Datei Bearbeiten Ansicht Favoriten Extras ?
Adresse ID313df2abe87911d3b7c0006097849ec8.xml?mode=script&target=html&toc=on
Wechseln zu
Evolutionäre Algorithmen
Beispiele ▶ Monte-Carlo-Suche nach binärem String
Allgemeines
Einordnung
Biologie
Terminologie
Beispiele
Monte-Carlo-Suche nach binärem String
Alternativer Algorithmus
Genetische Algorithmen
Classifier Systems (CS)
Genetische Programmierung
Evolutionsstrategien (ES)
Evolutionäre Programmierung
Simulated Annealing und andere Verfahren
EvA
Materialien
Hilfe
Einstellungen
Inhaltsverzeichnis
Internet
http://www-ra.informatik.uni-tuebingen.de/leax/images/book/hi...
http://www-ra.informatik.uni-tuebingen.de/leax/images/book/hire
Fertig

4.1 Beispiel 1: Monte-Carlo-Suche nach einem binären String

$\vec{a}^* \in B^l = \{1,0\}^l$

```
1. $k := 0$
2. repeat
3.     $k := k + 1$
4.     generiere zufällig einen Binärstring $\vec{a}_k$
5. until $\vec{a}_k = \vec{a}^*$
```

Algorithmus Monte-Carlo-Suche

Analyse: Erwartete Anzahl Schritte, bis $\vec{a}^*$ gefunden wird:

1. Wahrscheinlichkeit, $\vec{a}^*$ beim ersten Zufallsexperiment zu erhalten:
2. Wahrscheinlichkeit, $\vec{a}^*$ beim ersten oder zweiten Experiment zu erhalten: $p_2 = P\left\{\vec{a}_1 = \vec{a}^* \vee (\vec{a}_1 \neq \vec{a}^* \wedge \vec{a}_2 = \vec{a}^*)\right\} = 2^{-l} + 2^{-l}(1 - 2^{-l})$
3. Wahrscheinlichkeit, $\vec{a}^*$ bis zum k-ten Experiment einschließlich zu erhalten: Also $p_k = 1 - (1 - 2^{-l})^k$ oder äquivalent

Figure 1: Left: Typical page from online course "Evolutionary Algorithms". Note the high number of equations and symbols in the main content area. Small symbols, especially indices, are hard to read on-screen and do not print very well. Right: Using our improved system, learners can click on any equation to open a high resolution version in a separate windows. An additional print view offers also a high quality hardcopy option.

Our new publishing workflow consists of six steps:

1. Choosing/creating an appropriate document type definition (DTD).
2. Structuring a complete course according to the DTD in an authoring environment.
3. Converting the SGML content into XML and preparing it to be served on the WWW.
4. Creation of HTML page templates and navigational controls.
5. Developing a servlet that converts XML to HTML on-the-fly.
6. Tracking content usage into a relational database.

Document Type Definition (DTD)

We decided to enrich an XML-compliant, light-weight variant of the DocBook DTD (OASIS 1998) used in technical publishing with additional elements and attributes that cater for educational needs, different media types and hyperlink capabilities. We classified the elements inherited from DocBook into the given aggregation levels according to the LOM specification (IEEE 200). All level 0 objects were assigned a 32digit hexadecimal identifier attribute - unique in time and space.

Authoring environment

FrameMaker+SGML was chosen a as XML editor because it combines the ease of use of a modern word processor (especially equation handling) with the structural functionality of an XML editor. To track down learning objects during the following processing steps, each level 0 object was assigned a globally unique identifier. Once the identifiers are assigned, they will be present in FrameMaker+SGML files as well as in exported SGML documents. It is important to note that even though the content is kept in the FrameMaker+SGML file format, this format represents nothing more than a wrapper around a DTD compliant structure.

Converting and preparing XML content

The next step is to export the FrameMaker content into an SGML file using built-in export functionality. Image representations of embedded equations are also generated in this step.
A common problem creating scientific content for the WWW is to display mathematical equations. While some browsers already feature limited MathML rendering capabilities it is common practice to embed equations as bitmap images, e.g. GIF or PNG files. This one-way conversion complicates later-on modifications for the author and results in poor image quality of the content (especially when hardcopied. We decided to keep equations in this format within FrameMaker but to convert them in GIFs during SGML export. By exporting the course content twice with different resolution (72 dpi, 300 dpi), we get two versions of the same equation. This allows features like zooming into equations that are hard to read (Fig. 1) or high quality hardcopy options.
In a final step the *course description*, a hierarchical table-of-contents structure is extracted from the SGML document to serve as website structure. Course description, course content (stored in a set of XML files) and equation images are now copied onto the web server file system, ready to be served.
The complete conversion procedure is depicted in (Fig. 2). Export and equation rendering of a 200 page course on an Athlon 900 MHz Windows PC takes roughly 6 minutes for the 72 dpi version (30 minutes for 300 dpi) and does not require any human interaction.

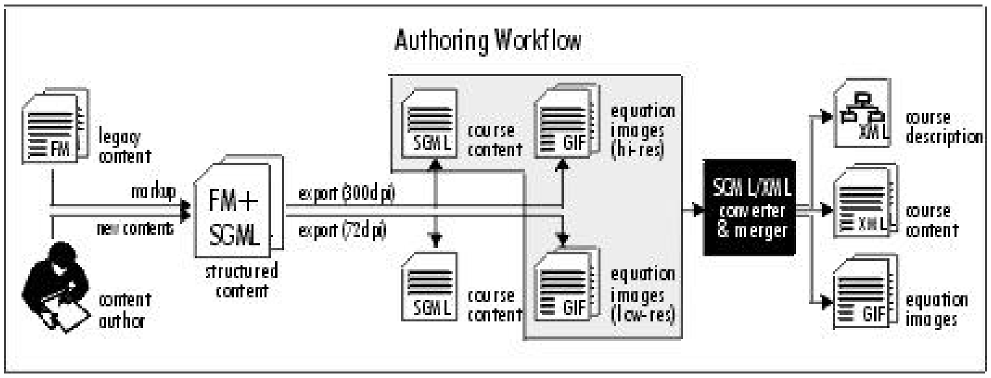


Figure 2: Authoring workflow. Content creation takes place inside the SGML editor (like FrameMaker+SGML). In two passes, SGML versions and two variants of equation graphics are exported. Finally these file get converted to XML and prepared for being served by our servlet-based system.

Page templates and navigational controls

Page templates were designed in an HTML editor, containing placeholders for dynamic content. These placeholders get replaced by applying several XSLT transformations using the open source XSLT formatter Xalan to the requested course content file. XSL style sheets exist for several target media (on-line, print) and usage modes (tutorial and script mode).
The globally unique learning object identifiers are preserved during conversion, allowing web browsers to make use of this information. We have implemented a smart, fine-grained annotation system that works at the subcomponent level within HTML pages and allows learners to tag individual page elements with personal comments. These comments can also be classified by the learner (question, answer, general remark, bug report) and given a restricted visibility to other users.

Serving the content

Because most browsers currently provide only limited XML support, the course content is server-side converted from XML to HTML. Additionally, this allows adaptation of the content to a specific user's learning context (e.g. formatting preferences, target media, usage mode) at this point.
To bridge the gap between web server and XML content, we developed a *S*ervlet *I*nterface for *O*nline *U*ser-adapted *X*ML (SIOUX). Java Servlets are plug-in extensions written in Java that add functionality to a HTTP daemon. To serve a request, SIOUX reads in the requested XML file, applies a XSL transformation to the target medium and mode, inserts the content into a page template, resolves server-side includes and other placeholders, returns the HTML page to the learner's client and logs user id, URL and learning context into a relational database.

Our setup builds on an Apache HTTP daemon and the MySQL RDBMS running on an RS/6000 workstation. Servlet support is provided by the JServ servlet engine, connecting to the SIOUX servlet on a Windows PC (Windows 2000, Athlon 900 MHz, 256 MB, Sun Java Runtime Environment 1.3). The response times vary between 0.2-0.6 seconds per page request depending on complexity of the XSLT style sheets. This is acceptable compared to the 1-5 minutes it takes a learner to work through the page content. (Fig. 4) shows how a request is handled by Apache/SIOUX and which resources are used.

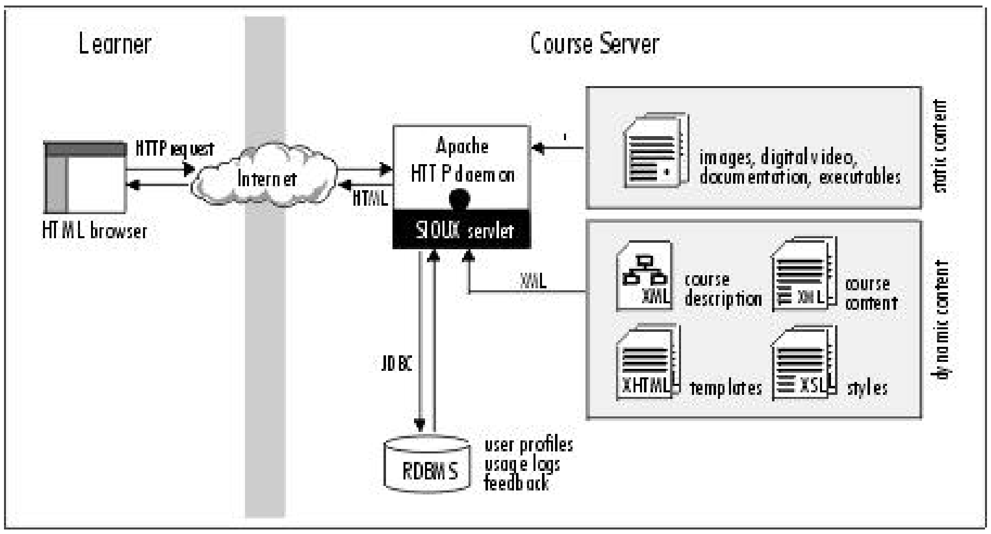


Figure 3: Resources used within the SIOUX architecture. The SIOUX servlet generates dynamic content from XML structure, content, style and template files. Static content is handled by the Apache server without SIOUX intervention.

Tracking content usage

The system contains a learner database that allows restricting access to course content, personalized tracking of content usage and persistent storage of user preferences. After a request has been served, the user's ID, the requested URL and all session parameters (e.g. current mode, target media, individual interface settings, date and time of access, referring URL, agent, ...) are logged into a relational database.

Conclusion

The paper illustrated how structured content supports the online learning process in creating, deploying and using course material. An example has been given, how an existing DTD can be modified to include the dedicated semantics needed in didactical processes while avoiding getting trapped in a proprietary format. Using one single, consistent source which is varied on-thy-fly also avoids complicated and error prone maintenance of several parallel versions. Rendering is acceptable fast for online learning purposes. Several modes (tutorial, script) match the main usage scenarios closer than a single all-purpose version. Preserving learning object identifiers in the resulting HTML pages makes way for applications like fine-grained annotation mechanisms on sub-page detail level.
The flexibility in displaying content in different flavors provides also a test bed for evaluation of different user interfaces or navigation concepts. Equally important is the ability to measure the usage of these experimental variations. While it is obvious that web interface design plays an important role in online education process, it is arguable which is the *best* way. Due to extended tracking possibilities we can offer an approach that helps to collect hard facts. In the likely case we'll learn that there is no single ideal interface for *all* learners, the presented workflow is already powerful enough to serve highly personalized educational content.

References

Arneil S. et al. (2000). *Hot Potatoes*, University of Victoria Language Centre, http://web.uvic.ca/hrd/halfbaked/

Bray T., Paoli J. & Sperberg-McQuee C.M., eds. (1998). *Extensible Markup Language (XML) 1.0*, W3C Recommendation 10-Feb-1998, http://www.w3.org/TR/1998/REC-xml-19980210.html

Brickley D. (1998). *Tutorial Markup Language (TML)*. Institute for Learning and Research Technology, University of Bristol, http://www.ilrt.bris.ac.uk/netquest/about/lang/

IEEE (2000). *Draft Standard for Learning Object Metadata*, IEEE Standards Department, Learning Technology Standards Committee, Piscataway, 2000. (IEEE P1484.12/D4.0).

IMS (2000). *IMS Content Packaging Specification*, IMS Global Learning Consortium, Inc., http://www.imsproject.org/content/packaging/index.html

IMS (2001). *IMS Question & Test Interoperability Specification*, IMS Global Learning Consortium, Inc., http://www.imsproject.org/question/index.html

ISO (1986). *ISO 8879: Information processing – Text and office systems – Standard Generalized Markup Language (SGML)*, International Organization for Standardization (ISO), Genf, 1986.

Krauße R. (1999). *Exercise Format*, TU Dresden, http://linus.psych.tu-dresden.de/Stupla/ef/

Müller H. & Deponte J. (1999). Distributed Multimedial Presentation and Conferencing, *Die Virtuelle Wissensfabrik*, Gesellschaft für Mathematik und Datenverarbeitung, St. Augustin, 1999.

Microsoft (2001). *LRN 2.0 Toolkit*, http://www.microsoft.com/eLearn

OASIS (1998). *The DocBook DTD*, Organization for the Advancement of Structured Information Standards, http://www.oasis-open/docbook, 1998.

Ram A. et al. (1999). PML: Adding Flexibility to Multimedia Presentations, *IEEE Multimedia*, 1 (2), 40-51.