xml2lms

Ein Werkzeug zur Transformation XML-basierter Lerninhalte

Jens Pönisch

TU Chemnitz

Zur Person

- Wissenschaftlicher Mitarbeiter Professur Wirtschaftsinformatik der TU Chemnitz
- Durchführung von Programmierpraktika
- Übungen Algorithmen und Programmierung, Datenstrukturen

Verbindung zu OLAT

- Persönliche Kontakte zum Bildungsportal Sachsen
- Zuarbeit zur Praktikumsarbeit Kosler/Hahn «OLAT-Analyse»
- Realisierung der PostgreSQL-Anpassung in Chemnitz
- Implementation des Uploads von ZIP-basierten Lerninhalten
- Mitimplementation von *Magma* für Streaming-Inhalte

Historie 1

- «Internet-Studium» als Internet-Fernstudiengang der TU-Chemnitz, ca. 1997
- 6 Bausteine, jeweils 3 Monate Studiendauer
- Lehrmaterial als HTML
- Einheitliches Layout
- Aber ca. 10 Autoren

Historie 2

- Entwurf eines XML-Dialekts durch Prof. Hübner
- Speziell für Lehrmaterialien
- Erfassen des Textes im normalen Editor
- Tippaufwand gering halten (<-> DocBookXML)
- Einarbeitungsaufwand gering halten, deshalb HTML-nah
- Unbekannte XHTML-Tags werden einfach durchgereicht
- Offline-Transformation mit Kommandozeilentool

Tool **lconml** bzw. **x2x**

Historie 3

- Teilnehmer wünschten Druckversion
- Neue Transformation nach LaTeX ergänzt
- Nebenprodukt: Lehrmaterial ZIN (Zertifikat Internet-Nutzung):
  - Lehrbuch
  - Online-Material
  - Generierung Prüfungsfragen (Papierform)

Historie 4

- Nach Einstellung Internet-Studium Entwicklung eingeschlafen
- Aber selbst viele Lehrmaterialien mit *x2x* erstellt
- Einführung OLAT - Material sollte weiter genutzt werden:
  - Variante 1: ZIP-Import für OLAT 3.1 für BPS Sachsen entwickelt
  - Variante 2: Umbau des Tools zur Transformation in das IMS-CP-Format

Entstehung von xml2lms

- Kompletter Neuentwurf
- Namensänderung zu *xml2lms* wegen Namenskonflikt
- Realisierung als Tcl-Starkit (Binary!) mit tDOM-Bibliothek:
  - in C implementiert, sehr schnell
  - DTD-Unterstützung
  - Elementsuche mit XPath
  - Namespace-Unterstützung

  XSL-T erschien zu umständlich.
- Derzeit nur Linux, Portierung nach Windows ca. 1 Tag Aufwand

Anforderungen

- Transformation nach: CP, HTML-Slides, LaTeX, LaTeX-Beamer
- QTI-Unterstützung
- Strengere Syntaxprüfung
- Anpassung an XML-Standards (Encoding, XInclude)
- Mathematiksatz über LaTeX (MathML zu komplex)
- Erweiterungen der Grammatik
- Geringer Transformationsaufwand der vorhandenen Dokumente

Kursaufbau

samples/struct.xml

```
<!DOCTYPE collection SYSTEM "xml2lms.dtd">
<collection id="kursID">
<title>Titel</title>
<author>Autor</author>
...
<motivation>Motivation</motivation>
<objective>Lernziele</objective>
<prequisites>Voraussetzungen</prerequisites>
<content>
...
</content>
<bibliography>...</bibliography>
<acronyms>...</acronyms>
</collection>
```

Gliederung

Das `content`-Element gliedert sich in

- `section`
  - `subsection`

Inhaltsverzeichnis:

- CP: automatisch
- LaTeX: `tableofcontents`-Element

Tabellen-, Abbildungs-, Folien- und Listingsverzeichnisse können ebenfalls erzeugt werden.

Abschnitte

Sections und Subsections setzen sich aus Blockelementen zusammen:

- Absätze (`p div`)
- Aufzählungen (`ul ol dl`)
- Tabellen (`table ctable`)
- Anmerkungen (`remark`)
- Abbildungen (`fig`)
- Mathematischen Formeln (`eq`, AMS-LaTeX-Syntax)
- Lehrelementen (`demo task answer qti`)

Geplant: mehrspaltige Textabschnitte (`multicol`)

Abbildungen

```
<fig href="url" id="id">Titel</fig>
```

- Zentrierte Ausgabe, Abbildungsverzeichnis ist möglich (`listoffigures`)
- Dateiendung entfällt, das passende Format wird abhängig von der Ausgabe gewählt:
  - GIF, PNG, JPEG für Content Package
  - PDF, MPS, JPEG, PNG für LaTeX, Beamer

Listings

```
<listing id="id">
<title>Titel</title>
<example [class="Klasse"]>
...
</example>
</listing>
```

- Alternativ: Einlesen aus einer Datei (Attribute `href`, `lines` und `encoding`)
- Listingverzeichnis möglich (`listoflistings`)
- `example` auch ohne `listing`
- Klassen `syntax result`
- Metasymbole mit `ms`

Folien

```
<slide [id="ID"] [split="1"]>
<title>Titel</title>
Blockelemente
</slide>
```

- Werden in Script (CP und LaTeX) eingebettet
- Folienverzeichnis kann erstellt werden
- Formate `slide` und `beamer` übernehmen nur diese Elemente
- Für `beamer` sind Overlays und Aufteilen auf mehrere Folien möglich

Mathematik 1

- MathML zu aufwendig für manuelles Schreiben
- Deshalb wie in DocBookXML LaTeX-Unterstützung
- Automatische Konvertierung in Graphiken mittels `LaTeX` (Paket `preview`) und `dvipng` bzw. `dvips` und *GhostScript*
- Unterstützung von *AMSMath*

Mathematik 2

Abgesetzte Formeln:

```
<eq [id="ID"] [class="Klasse"]>
LaTeX-Formel
</eq>
```

- Bei Angabe einer ID kann die Formel referenziert werden, dann auch Vergabe einer Formelnummer.
- Klasse wählt ein- oder mehrzeilige Formel aus
- Formel muss u.U. mit `CDATA` geschützt werden

Inline-Formeln mit Element `m`.

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} \pm \ldots$$
$$\frac{\pi}{4} = 4 \cdot \arctan\frac{1}{5} - \arctan\frac{1}{239} \tag{1}$$

Referenzen

- `section`, `subsection`, `ctable`, `eq`, `fig`, `listing`, `remark` können mit einer ID versehen werden.
- Auf diese Elemente kann im an anderer Stelle verwiesen werden, Verweis ist Hyperlink (auch in LaTeX)
- Parallel kann auf die Bibliographie verwiesen werden, Kennzeichnung wie in LaTeX, z.B. [xml2lms] oder [1]
- Acronyme können in ein Abkürzungsverzeichnis aufgenommen und dort referenziert werden (<`acn id=‘‘acronym‘‘/`>).

Tabellen

- Problem: unterschiedliches Tabellenmodell in HTML und LateX
- Lösung durch `colgroup`-Spezifikation (Ausrichtung, Spaltenbreite)
- Durch Angabe des `id`-Attributs referenzierbar
- Tabellenverzeichnis möglich
- Ausgabe der Tabelle üblicherweise zentriert

QTI

- Original-Grammatik zu komplex, deshalb Transformation
- Derzeit nur Single-Choice und Multiple-Choice-Fragen
- Antworten werden einheitlich mit einem Punkt bewertet
- Hinweise zu falschen Antworten werden unterstützt
- Ausgabe derzeit nur als QTI-Paket, für LaTeX geplant
- Bisher wenig genutzt, Verbesserungsbedarf!

Benötigte Tools

Neben `xml2lms` werden weitere Tools benötigt:

- `xmllint` zum Prüfen der Dokumente,
- `zip` zum Packen der CP- und QTI-Archive,
- `latex`, `dvips` und GhostScript zum Erzeugen mathematischer Formeln

Ablauf Dokumenterstellung

1. Erstellen der Hauptdatei `index.xml`
2. Erzeugen der DTD-Datei:

```
1 xml2lms -m dtd
```

3. Prüfen des Dokuments:

```
1 xmllint --xinclude --noout --postvalid index.xml
```

4. Konvertieren ins gewünschte Ausgabeformat:

```
1 xml2lms -f format
```

5. Bei CP und QTI Upload der ZIP-Dateien ins OLAT

Literatur und Online-Ressourcen I

xml2lms: `http://xml2lms.in-chemnitz.de/`

Tclkit: `http://www.equi4.com/tclkit/`