XML-BASIERTE DATENHALTUNG UND VERARBEITUNG FÜR DIE LAPIS-STUNDENPLANUNG

Dr. Thomas Meinike

Das im Folgenden kurz vorgestellte Projekt »planX4L« (Planung mit XML für LAPIS) wurde vom Autor im Rahmen von Aktivitäten zur Stundenplanung an der Hochschule Merseburg konzipiert und umgesetzt. Es dient zur besseren Verwaltung und Koordinierung der Stundenplan-Rohdaten und zur Produktion der vom Planungswerkzeug LAPIS importierbaren Datensätze zur weiteren Verwendung in der zentralen Planung.

Überblick

Die **Planungssoftware LAPIS** (genauer L.A.P.I.S, das Lehr-Angebot-Planungs- und Informations-System von Prof. Dr. H.-J. Rogge) [1] wird seit mehr als 20 Jahren an vielen Hochschulen und sonstigen Bildungseinrichtungen zur Stundenplanung verwendet. An der Hochschule Merseburg arbeiten die Einzelplaner der Fachbereiche bzw. Studiengänge ihre Daten der zentralen Planung zu. Das noch immer unverändert unter DOS laufende Programm besitzt den Charme und Bedienungskomfort einer Software der frühen 90er Jahre.

Erstellung und Verwaltung von mehr als 100 Lehrveranstaltungen eines typischen Semesters mit mehreren Studierendengruppen erweisen sich als eine aufwändige und fehleranfällige Prozedur. Um den eigenen Planungsprozess stärker zu optimieren und konsistenter zu machen, entstand die Idee der Datenhaltung in einem geeigneten XML-Format. Zusätzlich sorgt ein XSLT-Stylesheet für die Umwandlung der XML-Daten in von LAPIS importierbare ASCII-Dateien.

XML-Format

Die konzipierte XML-Struktur berücksichtigt alle wesentlichen Merkmale der von LAPIS benötigten Eingaben wie Namen und Nummern von Lehrveranstaltungen, Angaben zu Dozenten, Terminen und Räumen sowie Kopplungen mit anderen Veranstaltungen. Abbildung 1 gibt einen Einblick in die grundlegende Struktur und veranschaulicht einen prototypischen Veranstaltungs-Datensatz (Element fach).

```
<?xml version="1.0" encoding="UTF-8"?>
<stundenplanung xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="planX4L.xsd"
  info="wintersemester 2011/12" stand="2011-06-25">

  <matrikel stgnr="250" mname="MTRWK" semester="1" fb="2">
    <module>
      <modul kurz="EDM I" modnr="M1.1" name="Elektronische Dokumentation und Multimedia I"/>
      <!-- weitere modul-Elemente ... -->
    </module>

    <planung>
      <fach modref="M1.1" plannr="10" ang="j" stud="25">
        <name>Auszeichnungssprachen I (HTML)</name>
        <planname>AZ-Spr. I (HTML)</planname>
        <art plansws="4" effsws="2">VU</art>
        <dozent status="intern" fb="2">Meinike</dozent>
        <termin woche="u" raum="124/1/020" tag="Montag" zeit="09-11"/>
        <termin woche="u" raum="124/1/020" tag="Montag" zeit="
        <kopplung mit="1234567"/>
        <kommentar>Test</kommentar>
      </fach>
      <!-- weitere fach-Elemente ... -->
    </planung>
  </matrikel>

  <!-- weitere matrikel-Elemente ... -->

</stundenplanung>
```

07-09
09-11
11-13
13-15
15-17
17-19
19-21
??-??

Abb. 1: Einblick in die verwendete XML-Struktur

Zusätzliche Metainformationen bezüglich der Zuordnung von einzelnen Veranstaltungen zu den jeweiligen Modulen in den Bachelor- und Master-Curricula erleichtern später die Zuordnung von Prüfungen.

Datenmodell

Um konsistente Datensätze sicherzustellen, wurde ein XML-Schema entworfen (»planX4L.xsd«), welches detaillierte Vorgaben und Restriktionen beinhaltet, siehe Abbildung 2. Auch der Eingabekomfort für den Planer wird erhöht, da sich z. B. vorhandene Räume, Zeitblöcke und zur Verfügung stehende Dozenten aus maßgeschneiderten Listen auswählen lassen. Für die unmittelbare Arbeit am Markup wird natürlich eine generelle Affinität zum Umgang mit XML in einschlägigen Editoren vorausgesetzt.

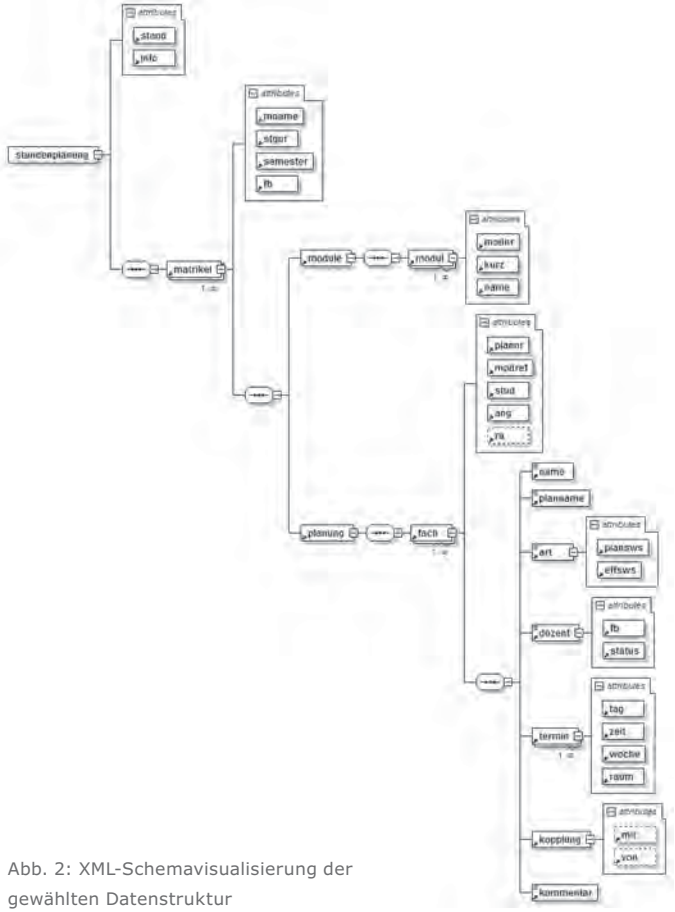

Abb. 2: XML-Schemavisualisierung der gewählten Datenstruktur

Datenkonvertierung für LAPIS

LAPIS arbeitet standardmäßig mit einem binären Datenformat, bietet jedoch auch eine ASCII-Formatoption. Dieses Format ist zeilenorientiert aufgebaut, wobei in jeder Zeile ein Datensatz (= Veranstaltung oder Fach) abgelegt ist. Die einzelnen Datenfelder sind durch »|«-Zeichen voneinander getrennt. Es handelt sich also um ein CSV-Format. Um dieses Format selbst zu erzeugen ist, mangels Dokumentation, einige Detektivarbeit erforderlich. Ein Zieldatensatz mit den in Abbildung 1 ersichtlichen Informationen hat diesen Aufbau (eine Zeile):

2250110|VÜ AZ-Spr. I (HTML)|MTRWK1|4|1234567|F2 Meinike|Mo2|124/1/020|Mo3|124/1/020|\Test|25|VÜ||1|1|1|H|72

Konkrete Details sollen an dieser Stelle nicht weiter diskutiert werden, da diese bei Bedarf durch das Studium des entwickelten XSLT-Stylesheets (»planX4L.xsl«) erschlossen werden können [2, 3].

Transformation

Mit der genannten XSLT 2.0-Transformationsvorlage kann unter Nutzung eines geeigneten XSLT-Prozessors wie Saxon-HE 9.3 [4] oder AltovaXML 2011 [5] das gewünschte ASCII-Zielformat aus dem angelegten XML-Plandokument erzeugt werden. Nach dem Einlesen in LAPIS können weitere Operationen und Verfeinerungen sowie notwendige Kontrollen von Überschneidungen oder Druckfunktionen angewendet werden.

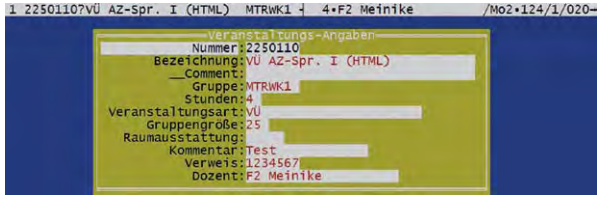

Abb. 3: Beispieldatensatz in der LAPIS-Ansicht nach ASCII-Import

Ergebnisse

Abbildung 3 zeigt den oben erläuterten Beispieldatensatz (Element fach) in der LAPIS-Programmansicht. Die Transformation der XML-Daten mit anschließendem Import der generierten ASCII-Datei funktioniert wie beabsichtigt.

Dennoch ist das vorgestellte Prinzip kein Ersatz für LAPIS selbst und in der konkreten Implementierung nur auf die eigenen Bedürfnisse zugeschnitten. Insofern handelt es sich um eine Insellösung, vergleichbar mit ähnlich gelagerten Versuchen von Kollegen unter Verwendung von Tabellenkalkulationen oder Datenbanksystemen für die Datenhaltung.

Unabhängig von der nur eingeschränkten Möglichkeit der Wiederverwendung durch andere Planer, demonstriert das Projekt speziell für die eigentliche Planungszielgruppe der Studierenden im Bereich Technische Redaktion die praktische Anwendung von XML-Technologien und einige Ansätze zur Automatisierung von derartigen Routineaufgaben.

Anmerkung: Die zum Zeitpunkt der Abgabe dieses Artikels aktuelle Planungs-XML-Struktur des Autors für das Wintersemester 2011/12 umfasst ~1300 Zeilen und wird durch die Transformation in ~100 Zeilen ASCII-Code aufbereitet.

Referenzen

[1] L.A.P.I.S: http://www.mcl.fh-osnabrueck.de/www-neu/lapis/infos/index.htm

[2] XSLT-Code: http://www.iks.hs-merseburg.de/~meinike/projekte/planX4L/planX4L_xsl.html

[3] W3C-Spezifikation XSLT 2.0: http://www.w3.org/TR/xslt20/

[4] XSLT-Prozessor Saxon: http://saxon.sourceforge.net/#F9.3HE

[5] XSLT-Prozessor AltovaXML: http://www.altova.com/altovaxml.html

Kontakt zum Autor: thomas.meinike@hs-merseburg.de
Web: http://www.iks.hs-merseburg.de/~meinike/