XML-based Markup Language for Description of Syllabus Courses

© Anastasia A. Zamishlyaeva
Chelyabinsk State University
stasya@csu.ac.ru

PhD advisor: Mikhail L. Zymbler

Abstract

In this paper we present a new markup language, named SyllabML, to describe syllabus courses according to standards of the Russian Ministry of Science and Education. SyllabML is XML-based mobile language and allows describe whole structure of syllabus in adequate way. The paper also gives an implementation schema of SyllabML translator.

1. Introduction

The Ministry of Science and Education of Russian Federation requires that every course have a standard course syllabus. *Syllabus* is an independent document. It includes multiple sections that have an information about the authors of syllabus, allocation of courses in semesters, students, for whom this syllabus is made, etc. This syllabus is developed by each department based on a standard format, and describes the goals, credits, examination, recommended literature, topics for each course, etc.

At first template of syllabus was developed in format of MS Word and MS Excel file at Chelyabinsk State University (ChelSU). Later on *Academic E-Document System* was put into operation in ChelSU. This System deals with Educational Standards, Curricula and Syllabus. It is based on the relational Database. Every lecturer creates syllabus using *Syllabus Preparation Module* (*SPM*) via web-interface.

There are two versions of SPM: an Internet and a local one. *Internet version* is used by lecturer to public his/her syllabus and make available for all concerned. For this lecturer needs to be connected to the Internet because all date is inserted in University Warehouse. A lecturer uses *local version* to create syllabus for him-/herself. He/she doesn't need to have an Internet connection. All information is placed on local database.

Dealing with this system, a lecturer is faced with the following basic problems:

1. *Necessity of synchronization*. Local version of Database and an Internet one need to be synchronized in both directions by request of a lecturer-owner of syllabus.
2. *Poor Internet connection*. There are such problems as low speed and quality of the Internet connection. It needs a lot of time to transmit data. Some lecturers don't have an ability to connect to the Internet at all as they don't have needed equipment.
3. *Difficulty of installing* local version. If a lecturer wants to develop a syllabus at home without using the Internet it is necessary for him/her to install system at home, that contains several essential parts (such as Apache, Perl, DBMS) that demand some special resources.

This article describes an approach of solving these problems on the base of specialized description language based on XML technology.

2. Syllabus Markup Language

2.1 Analysis of Requirements to SyllabML

The main idea of solution of listed problems is to use some formal description language of syllabus and translate from this language for inserting data to Database.

While having an analyses of requirements, we found out that this description language must possess the following basic criteria:

1. *Adequacy*. Course syllabus needs to be presented in formal way. There must be facilities for correspond description of syllabus (its structure, objects, etc).
2. *Mobility*. Lecturer can develop syllabus in any computer on any platform and OS he/she likes and then import it in data base. Independence from connection to the Internet.
3. *Minimality*. The amount of special software to install for this system needs to be decreased.
4. *Simplicity*. Language must be simple in its use both for Russian-spoken lecturers and translator developers.

Basing these criteria we have made analyses of some languages and description tools whether they can be used for description of syllabus for local version of SPM. Table 1 summarizes this analysis.

Markup languages, such as *HTML* [1], posses all this properties. But HTML is just a language of presentation of text and can't describe the structure and objects of syllabus.

Proceedings of the Spring Young Researcher's Colloquium On Database and Information Systems SYRCoDIS, St.-Petersburg, Russia, 2004

Set of *SQL*-queries can insert data of syllabus in database and adequacy describe the structure of syllabus. But SQL is too complex and difficult for ordinary lecturers. We can't demand from every lecturer to know SQL.

Table 1.Comparison of languages and tools for description of syllabus

Criteria	Languages			Tools	
	HTML	SQL	SyML	Excel forms	Web forms
Adequacy	-	+	-	+	+
Mobility	+	+	+	-	-
Minimality	+	+	+	±	-
Simplicity	+	-	+	+	+

SyML (Syllabus Markup Language) [2] is an XML application designed to make syllabus preparation easier, and to make syllabuses themselves more consistent and more amenable to post-processing into a variety of forms. For instance, it is easy to make a Web page out of your syllabus using SyML and the appropriate processing tools; and it would not be too difficult, given a large number of SyML-formatted syllabuses, to conduct database operations on them. It was developed at Seton Hall University. Educational standards in Russia and in the USA are various and have different parameters, that's why SyML can't reflect total structure of syllabus made by Russian lecturers.

Excel forms don't posses criterion of mobility.

Using local version of SPM via *web forms* needs installing a lot of applications (such as Apache web server, Perl interpreter, Oracle DBMS). Therefore this tool doesn't possess criterion of minimality.

For all these reasons language for description of syllabus must be a new one.

There are several ways of development of new description language. Description of this ways is generalized in Table 2.

Table 2. Comparison of languages for development of description language

Criteria	Language in the base		
	XML	SGML	new
Adequacy	+	+	+
Mobility	+	+	+
Minimality	+	+	+
Simplicity	+	±	-

This language can be start from scratch, but it is too hard problem for developer.

It can be developed on the base of *SGML* [3]. SGML is the Standard Generalized Markup Language, the international standard for defining descriptions of the structure of different types of electronic document. SGML is very large, powerful, and complex. It has been in heavy industrial and commercial use for over a decade, and there is a significant body of expertise and software to go with it.

XML (Extensible Markup Language) [4] is a lightweight cut-down version of SGML which keeps enough of its functionality to make it useful but removes all the optional features which make SGML too complex to program for in a Web environment. That's why the use of XML is more justified then the use of SGML, as XML omits all the options, and most of the more complex and less-used parts of SGML in return for the benefits of being easier to write applications for, easier to understand, and more suited to delivery and interoperability over the Web.

So new language is an XML application. It is called Syllabus Markup Language (SyllabML). SyllabML ideologically is similar to such markup languages [5] as MathML, TEI. *Mathematical Markup Language (MathML)* [6] is intended to facilitate the use and re-use of mathematical and cientific content on the Web, and for other applications such as computer algebra systems, print typesetting, and voice synthesis. *TEI (Text Encoding Initiative)* [7] is an international and interdisciplinary standard that helps libraries, museums, publishers, and individual scholars represent all kinds of literary and linguistic texts for online research and teaching, using an encoding scheme that is maximally expressive and minimally obsolescent. So SyllabML is used in its own branch.

2.2 Informal Description of SyllabML

Structure of SyllabML is similar to the standard form of syllabus in ChelSU. It makes easier use of this language by lecturers, who has got accustomed with this format.

The main objects of SyllabML are tags, attributes, text elements and comments.

Example of SyllabML code in English is illustrated on the Figure 1.

Each SyllabML document, according to the specification of XML, contains one or more elements, the boundaries of which are either delimited by start-tags and end-tags.

The beginning of every SyllabML element is marked by a start-tag, e.g.

```
<literature type="suggested" InLibrary="yes">
```

In this case **literature** is *name* of start-tag. The name in the start- and end-tags gives the element's type. The pairs `type="suggested"` and `InLibrary="yes"` are attribute specifications of the element. Each attribute specification has a name (type, InLibrary in this example) and a value ("suggested", "yes"). Attribute value is the text between the ' or " delimiters.

The end of every element that begins with a start-tag must be marked by an end-tag containing a name that echoes the element's type as given in the start-tag, e.g. `</literature>` in Figure 1.

The text between the start-tag and end-tag is called the element's content. Content of elements can be an element, text or comment. For instance, content of the element `literature` consists of elements `author`, `title` and `param`. Next, the `direction` element has text content `Computer Science`. And `<!--start of title_list-->` is comment.

```
<syllabus lang_ver="1.0">
<!--start of title_list-->
<title_list>
   <faculty
type="author">Mathematical</faculty>
   <dept>Sortware development</dept>
   <faculty
type="student">Mathematical</faculty>
   <standard>2</standard>
   <direction>Computer Science</direction>
   <!--<trade></trade>-->
   <author>M.L. Zimbler</author>
   <hours>
      <lecture form="internal">70</lecture>
      <lecture form="corresp">8</lecture>
      ...
   </hours>
   <semester>
      <exam form="corresp">2</exam>
      <credit form="internal">2</credit>
   </semester>
   ...
</title_list>
<!--end of title_list-->
...
<!--start of literatire-->
<literature      type="suggested"      inLi-
brary="yes">
       <author>E.A. Zuev</author>
       <title>Turbo Pascal 6.0</title>
       <param>M.: Uniteh, 1992.</param>
</literature>
...
<lecture title="Introduction" hours="2">
Introduction to Turbo Pascal, main objects.
</lecture>
...
<!--end of literatire-->

…
</syllabus>
```

Figure 1.An example of SyllabML code

Comments may appear anywhere in a document outside other markup; in addition, they may appear within the document type declaration at places allowed by the grammar. They are not part of the document's character data. For compatibility, the string "--" (double-hyphen) must not occur within comments. An example of a comment from Figure 1: `<!--start of title_list-->`.

For lecturers there is no need to earn all the tags, attributes of tags by heart. They can use special templates for this edit them and get their own syllabus.

2.3 SyllabML Translator Development

On the one hand we need our SyllabML document to be presented in format of SQL (for databases synchronization) and HTML (visualization of syllabus for lecturer in local version). On the other hand, we need to get XML-document from SQL for synchronization local database from University Warehouse.

XML and Java technology are natural partners in helping developers exchange data and programs across the Internet. That's because XML has emerged as the standard for exchanging data in heterogeneous systems, and Java technology provides a platform for building portable applications [8].

SyllabML translator can be developed on using standard XML parsers. There are three popular XML parsing techniques for Java:

- *Document Object Model* (DOM) [9], a mature standard from W3C
- *Simple API for XML* (SAX) [10], the first widely adopted API for XML in Java and a de facto standard
- *Streaming API for XML* (StAX) [12], a promising new parsing model introduced in JSR-173 (Java Specification requests) [13]

Each of these techniques has benefits and drawbacks. It is shown in Table 3.

DOM is a tree-based parsing technique that builds up an entire parse tree in memory. It allows complete, dynamic access to a whole XML document.

SAX is an event-driven push model for processing XML. It is not a W3C standard, but it is a very well-recognized API that most SAX parsers implement in a compliant way. Rather than building a tree representation of an entire document as DOM does, a SAX parser fires off a series of events as it reads through the document. These events are pushed to event handlers, which provide access to the contents of the document.

StAX is an exciting new parsing technique that, like SAX, uses an event-driven model. However, instead of using SAX's push model, StAX uses a pull model for event processing. Instead of using a callback mechanism, a StAX parser returns events as requested by the application. StAX also provides user-friendly APIs for read-in and write-out.

DOM parser technique is more right for using for development SyllabML, as we don't need an event-driven model. The second reason for using DOM is that DOM is best for applications that need to modify XML documents or for XSLT (Extensible Stylesheet Language Transformations) for making HTML with syllabus.

Java Architecture for XML Binding (JAXB) [8] provides an API and tool that allow automatic two-way mapping between SyllabML and Java objects [10]0.

The main steps of SyllabML translating is presented on Figure 2.

On entrance of SyllabML translator we have a SyllabML document. With a given SyllabML schema definition (*Schema* is an XML-based representation of the structure of document.), the JAXB compiler can generate a set of Java classes that allow to build applications that can read, manipulate and recreate SyllabML documents without writing any logic to process its elements.

XML Scanner and parser are components of JAXB API.

SyllabML code generator is Java application.

SQL scripts and HTML are going out from SyllabML translator.

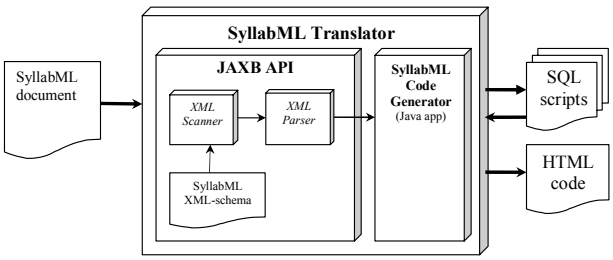

Figure 2. SyllabML Translator Implementation Schema

Also with help of SyllabML translator we can translate SQL- scripts into SyllabML document. It is necessary to use in synchronization local database with University Warehouse.

3. Conclusion

In this paper we have took up a problem of creating special language for description of syllabus. Syllabus Markup Language an XML application is presented as this special language.

This language confirms criterion of adequacy and simplicity. SyllabML document represented the whole structure of syllabus made by Russian spoken lecturers. SyllabML is mobile language. SyllabML document can be done in ordinary text editor. As SyllabML document can be written in ordinary text file, the problem of necessity of installing huge amount of software (DBML, interpreter, Apache) is no longer relevant. So SyllabML possess guideline of minimality.

In this paper SyllabML translator is offered. SyllabML translator uses Java Architecture for XML Binding (JAXB) as scanner and parser and our own Java application as SyllabML code generator.

Different parsing techniques have been view: DOM, SAX and StAX. In SyllabML translator a DOM parsing technology is used.

As result, SyllabML translator produses SQL-script and HTML code that corresponds SyllabML document.

SyllabML language was developed in Chelyabinsk State University within the frames of the project of Academic E-Document System.

References

[1] HyperText Markup Language http://www.w3.org/MarkUp/
[2] SyML – Syllabus Markup Language. http://knossos.shu.edu/dblack/syml/
[3] Extensible Markup Language (XML) and Standard Standard Generalized Markup Language. http://www.w3c.org/TR/WD-DOM
[4] Elliotte Rusty Herold, W. Scott Means. XML in a Nutshell. O'Reilly&Associates, 2001.
[5] XML Applications and Initiatives. http://xml.coverpages.org/xmlApplications.html
[6] Mathematical Markup Language (MathML) Version 2.0 http://www.w3.org/TR/2001/REC-MathML2-20010221/
[7] The Text Encoding Initiative (TEI) Consortium http://www.tei-c.org/
[8] Java Architecture for XML Binding http://java.sun.com/xml/index.jsp
[9] W3C Document Object Model http://www.w3.org/DOM/
[10] SAX - Simple API for XML http://www.saxproject.org/
[11] An Introduction to StAX http://www.xml.com/pub/a/2003/09/17/stax.html
[12] The Java Community Process(SM) Program - JSRs: Java Specification Requests http://jcp.org/en/jsr/all