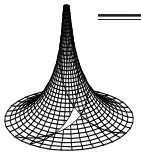

5/2 (2007), 317–335
tmcs@inf.unideb.hu
http://tmcs.math.klte.hu

Teaching
Mathematics and
Computer Science

Teaching XML

Péter Jeszenszky

Abstract. The author has been teaching XML at the Faculty of Informatics, University of Debrecen since the end of the nineties. This paper gives an overview of XML technology from an educators viewpoint that is based on the experience that the author has gained teaching XML over the years. A detailed description of the XML course is provided. Methodological issues are also discussed.

Key words and phrases: programming languages; programming techniques, software engineering; objectives of computer science teaching, computer literacy; lesson planning; teaching methods and classroom techniques, evaluation of instruction; data bases, information systems; telecommunication; educational uses; document and text processing, administrative data processing, application packages for personal computing; comprehensive works on instructional materials, educational technology and media research; technological tools.

ZDM Subject Classification: P40, P50, Q50, Q60, R50, R70, U10, U70.

1. Introduction

XML was born in the mid-nineties. Although an initial draft was published in late 1996, the first stable standard came out at the beginning of 1998. The goal of its designers was to create a general purpose format for structured documents on the World Wide Web that is also compatible with the rather complicated SGML [11] standard. It is an universal data exchange format and more as there exist countless applications of the technology today. XML is one of the hottest Internet technologies in the industry in our age.

There is no need to explain the necessity of teaching XML in computer science education. XML is a technology that all well-educated IT professionals must be familiar with.

The author has been teaching XML at the University of Debrecen almost since the birth of the technology, the end of the nineties. Since then XML has grown up and became inevitable. The authors goal was to feature the most current technologies on the lectures and to empower students to put XML into practise. Another important objective was to provide students with the ability to retrain themselves alone in the field. The latter is quite difficult since it requires several capabilities, e.g. the ability to read technical specifications.

This paper gives an overview of XML technology from an educators viewpoint that is based on the experience that the author has gained teaching XML over the years.

Section 2 summarizes the most important characteristics of XML from an educators perspective.

Section 3 presents a detailed description of the XML course that the author teaches here at the Faculty of Informatics, University of Debrecen (target audience, prerequisite requirements, topics covered, course requirements, detailed course outline).

Section 4 discusses teaching XML, including methods and difficulties.

Section 5 is devoted to methodological issues. It presents practical recommendations that should be taken into consideration in teaching XML in the classroom.

Section 6 provides an overview of the software tools that are used in the course.

In the following section two quite complex and realistic XML applications are presented that are served as case studies on the lectures to demonstrate the implementation of XML processing techniques and also the versatility of XML itself.

Finally, concluding remarks are presented.

2. Characterization of the field

XML was born as a markup language that is a simpler alternative to SGML to be used on the World Wide Web. In a wider sense XML now stands for a bunch of standards, that is often called the XML family. It is easy to get lost in the proliferating jungle of mysterious acronyms.

Familiarizing students with all the different applications of the technology would require several semesters. Separate courses may be dedicated to many of the XML standards. The course that is presented here is an introductory one that covers only the most important standards and applications.

Another important characteristic feature of the field is that it is evolving rapidly. Forthcoming versions of many XML standards are under development and they might change their predecessors in the near future. Some of the specifications are still experimental and under heavy development. The good news is that the most important standards can be considered to be stable, thus they are teachable in the classroom.

Moreover, several stable versions of the same standard may coexist. For example, the XPath 2.0 and the XSLT 2.0 standards have been advanced to W3C Recommendation status at the end of January, that is a significant step forward. However, XPath 1.0 and XSLT 1.0 will remain widely used and important for a while, as these are still more widely accepted and supported than their 2.0 counterparts. Accordingly, the course covers only the 1.0 versions of these standards currently.

3. About the course

3.1. General information

The title of the course is XML Technology and Applications. Its main objectives are the following:

- To give an overview of XML and related standards (the "XML family").
- To present many different applications to demonstrate the importance of the field.
- To provide an introduction to the most fundamental XML standards.
- To teach students to work with XML tools.
- To provide students with the ability to use XML processing techniques in Java applications.

The course consists of two hours computer lab per week (total number of weeks in a semester is 14). The course is aimed at computer science students.

3.2. Prerequisite requirements

Prerequisite requirement for the course is the following: students must have completed a Java programming course. Students are required to have a sound knowledge of the Java programming language. An ideal student is in his or her third year at least, is enthusiastic about learning modern technologies, likes to read technical literature and has the ability to learn independently.

3.3. Topics covered

The following topics are covered in the course:

Introduction The role and importance of XML. Overview of the XML family of standards.

XML fundamentals An introduction to the core XML standards. Basic concepts: characters, markup, well-formedness and validity, namespaces.

Working with XML documents Creating XML documents using XML editors.

W3C XML Schema An overview of W3C XML Schema [30]. Working with XML Schema documents.

XML format design How to design an appropriate XML format.

Processing XML documents in Java Using the SAX [25], the StAX [16], the DOM [2] and the JAXB [13] APIs.

Transforming XML documents XSL Transformations (XSLT) [28]. Working with XSLT stylesheets.

3.4. Course requirements

Students taking the course are required to hand in a homework assignment to get a grade at the end of the semester. That is either a computer program or an XML schema to a hypothetical XML application, together with an example instance and an XSLT stylesheet that transforms instances into nice looking XHTML web pages.

A typical Java programming assignment is the following: Write a Java application that reads an XHTML document using the StAX API, and rewrites URIs in attribute values according to the rules specified in a special XML document (actually, in an XML catalog)! The assignment requires some background research on XML catalog formats and also on the third-party class libraries that are available to handle XML catalogs.

A typical assignment of the second kind: Desing an XML format to desribe an athletics event calendar/the periodic table of elements! An appropriate XML schema is required that makes heavy use of the built-in simple datatypes of W3C XML Schema. Provide an instance and also an XSLT stylesheet that transforms it into a nice looking XHTML web page!

3.5. Course outline

Table 1 contains the detailed course outline. Although a semester consists of 14 weeks, there are only 13 weeks in the table. One extra lab session is kept in reserve, in case of running out of time during the discussion of a topic.

Table 1. Detailed course outline

	Topics discussed
Week 1	**Introduction** What is XML and what is it good for? Core and related standards. The "XML family". XML applications. **An introduction to the XML 1.0 specification** XML documents. Characters. Markup constructs. Well-formedness.
Week 2	**An introduction to the XML 1.0 specification (continued)** Document type definition (DTD). Validity. Entities. **Working with XML** Using XML editors to create, edit and validate XML documents.
Week 3	**XML design** How to design an appropriate DTD? **Working with XML (continued)** Using XML editors to create, edit and validate XML documents. **Namespaces** Namespace concepts. Expanded names and qualified names. Declaring namespaces. Scoping rules for namespace declarations.

Topics discussed (continued)	
Week 4	**An introduction to W3C XML Schema** Datatypes. Simple datatypes: atomic, list and union datatypes. Built-in simple datatypes. Schema documents. Schema components. Declaring elements and attributes. Defining simple types. **Working with W3C XML Schema** Using XML editors to create schema documents.
Week 5	**An introduction to W3C XML Schema (continued)** Complex types. Building content models. Defining complex types. Abstract elements. Substitution groups. Using namespaces. Associating instances and schemas. Defining identity-constraints. Re-using schemas (import, include).
Week 6	**XML Path Language 1.0 (XPath)** Data model. Data types. Expressions. Location paths. **An introduction to XSLT 1.0** Stylesheets. Template rules. Processing model. Applying templates. Built-in template rules. Creating the result. **Working with XSLT** Using XML editors to create and edit XSLT stylesheets and to transform XML documents.
Week 7	**An introduction to XSLT 1.0 (continued)** Named templates. Repetition. Conditional processing. Sorting. Variables and parameters. **Working with XSLT (continued)** Using XML editors to create and edit XSLT stylesheets and to transform XML documents.
Week 8	**Processing XML with Java: introduction** An overview of XML Java APIs. XML support in Java SE 6. **Processing XML with Java: the SAX API** Event-driven XML document processing. Event handler interfaces and callback methods. SAX filters.

Topics discussed (continued)	
Week 9	**Processing XML with Java: Streaming API for XML (StAX)** Reading and writing XML documents using the cursor API and the event iterator API. Simultaneous reading and writing. Filtering and manipulating the stream of events.
Week 10	**Processing XML with Java: the Document Object Model (DOM)** The DOM data model. DOM interfaces. Reading XML documents into a DOM document tree. Traversing DOM trees. DOM modification. DOM serialization.
Week 11	**Processing XML with Java: the JAXB API** Architecture. The binding process. Binding XML schemas. Java code generation using the schema compiler. Binding customization. Marshalling, unmarshalling and validation.
Week 12	**Processing XML with Java: case study (1)** XML processing in real-life Java applications. See subsection 7.1 for details.
Week 13	**Processing XML with Java: case study (2)** XML processing in real-life Java applications. See subsection 7.2 for details.

Note that, a few related topics are not listed explicitly in the course outline. For example, XHTML and CSS are not mentioned namely, there are no separate lectures that are devoted to them. Actually, the result of transformations is XHTML in the most cases when XSLT is discussed. Students usually have preliminary knowledge about HTML. On the other hand, they can learn XHTML basics from examples with ease. In the authors experience, most of the students are not familiar with CSS, but a few minutes length introduction is enough to start with it.

4. Teaching the course

4.1. Structure of the lectures

As mentioned earlier, the course consists of two-hour laboratory sessions per week only and there are no separate lectures. However, some of the topics covered (i.e. XML standards) require in-depth explanations. As a solution, a part of the laboratory sessions in the first half of the semester is held as a lecture.

For example, on the first week a traditional classroom lecture is presented to the students. In the first half of the lecture many different XML applications (e.g. XHTML, MathML, SVG, XForms, XML-based web services, VoiceXML) are introduced to demonstrate the versatility of the technology. The second half of the lecture is an introduction to the XML 1.0 standard. Students will work with XML documents on the second week.

Usually, the first half of a two-hour meeting shall be considered as a classroom lecture, but only in the first half of the semester, when the topic to be covered is an XML specification (e.g. W3C XML Schema, XPath, XSLT).

4.2. Teaching

Certain topics occur again and again at the lab sessions. For example, XML schemas are used in the second half of the semester also, that is about Java programming. (The JAXB API also relies upon W3C XML Schema.) That gives the opportunity to present more examples, to revise important concepts again and again, and also to introduce additional features. Similarly, students work with XML tools constantly, that allows the teacher to introduce their features gradually, "in small doses".

In order to motivate students they often get interesting homework assignments that require research work (for example, reading API documentation or other literature). The goal of these assignments is often to improve discussed classroom examples. The solutions are discussed at the end of the following meeting and they are also made available on the web page of the course.

4.2.1. Teaching XML standards

Unfortunately, some XML specifications are so huge that it is impossible to present them in details if only a few hours can be devoted to each of them. Therefore the course must concentrate on the most important features of the

standards. It is very important that students must understand key concepts, that can serve as a basis for independent learning.

For example, there are only two two-hour sessions to discuss the quite complex and sophisticated W3C XML Schema standard. To teach the specification key concepts and features are identified. As a result, a compact syllabus is available at the web page of the course [31] that has been developed by the author during the years. This syllabus presents the most important concepts and features through examples. (The W3C XML Schema examples together with example instances are also available in a ZIP file for download.)

4.2.2. Teaching XML processing

Teaching XML processing in the second half of the course is more straightforward. Although APIs are often made up of many classes and interfaces and many of them contain a huge number of methods, key concepts can be demonstrated using a small subset of carefully selected examples. Students must be provided with simple and reusable usage patterns to demonstrate the steps that are requried to solve a specific programming task using a specific part of an API. To solve problems they can use these examples as a starting point.

A typical example is a Java class that reads an XML document using the StAX API and copies the contents of the document to another file unaltered. The program consists of a few lines of code only and is not very useful, but shows how to create an XML reader and writer object, how to read the contents of the document iterating over a stream of event objects, and how to write an event stream to create another XML document. To do something useful (for example, to remove comments from the input document) just a few additional lines of code should be inserted into. Actually, the next example program in the course performs comment removal.

Of course, more advanced examples must be introduced also. Moreover, at the end of semester a few really advanced applications are presented as use cases (see section 7 for details).

4.3. Difficulties of teaching

In the early years of teaching XML the most difficult problem was the manipulation of XML documents (XML schema documents and instances, XSLT stylesheets) in practice, because of the cumbersome and often verbose syntax. Today that is not a problem at all, since excellent software tools exist to work

with XML documents. Thanks to context sensitive XML editors students do not have to be worried about the syntax any more and the teacher can stick to the essentials.

Today's central problem is that the most important XML standards are often too complex and not easy to understand. W3C XML Schema is a typical example of that. The use of W3C XML Schema complex types are far from being simple. Although GUI tools may assist with the creation of XML schema documents and instances, a fair understanding of the concepts is still necessary to use XML Schema. Furthermore, the detailed discussion of some of the topics should require several lectures, but only a few hours are available to cover each of them. As a solution, the author offers notes that present the key features of the core XML specifications.

4.4. Books and course materials

The author has learned XML from the most authentic sources that are available, i.e. from technical specifications. These documents are very pedantic since they are intended primarily for implementers. It is clear that they are inappropriate for teaching purposes.[1] It is the educators responsibility to provide the students with carefully selected learning resources, including books and handouts. Students usually prefer books that are written in Hungarian. Fortunately, several technical books on XML are available in Hungarian.

It is very important that several technical books on XML have been translated into Hungarian in the past few years[2]. There is also a broadly accepted Hungarian XML terminology. On the other hand, there are (sometimes experimental) application domains where there is still a need for adequate Hungarian terms.[3]

[1]A good example is the XML 1.0 specification itself, that is an extremely elaborate document. Although well-formedness and validity are basic concepts, the specification contains approximately 40 well-formedness and validity constraints. Some of them are easy to remember, however there are more sophisticated constraints that are hard to comprehend.

[2]It is worth noting that the XML 1.1 specification also has been translated into Hungarian by the students of our faculty. Its publication as an Authorized W3C Translation is being in progress.

[3]The W3C-translators [29] mailing list maintained by the Hungarian W3C Office is a public forum devoted to the translation of Web technology related technical terms into Hungarian. The last topic discussed on the forum was that how to translate the term "Web Service Policy" into Hungarian.

[1] and [9] are good reference books that cover all of the topics that are discussed or mentioned in the lectures, including XML fundamentals, XPath, XSLT, XML schemas, XHTML and CSS. Since [1] is available in Hungarian, it can be used as a textbook. [17], [19], [20] and [26] are good references also. Each of them is devoted to a certain topic, such as XSLT, CSS or XHTML, providing a more in-depth discussion. Although [8] was published in 2002, it is still one of the best Java and XML books that are available today. Moreover, the complete text of the book is freely available on-line. [7] provides an excellent collection of XML best practices, and is available in Hungarian also.

However, the author prefers to use his previously mentioned self-made course materials in the lectures. These together with the above mentioned books provide the students with a rich source of information, that is more than enough.

5. Methodological issues

5.1. Provide students with carefully selected electronic and printed course materials

This includes books, papers, clear examples as well as handouts and lecture slides on the standards. Course material that have been developed by the author to support XML teaching can be found at [31].

5.2. Be practical

Since the course covers a wide range of topics, each of which is huge and complex, the course must be very practical. This approach is often appreciated by students so much. Examples (sample XML documents and XML schemas, computer programs) must be expressive. Provide a huge number of such examples to demonstrate all aspects of the technical specifications and the APIs. Based on these examples and other course material students must be able to solve real problems.

5.3. Provide the students with interesting and challenging assignments

It is very important to provide the students with exciting problems and assignments. Tasks taken from real life are the best.

5.4. Provide a broad perspective

Students must comprehend the role and importance of XML. To achieve the goal provide a wide range of different applications.

5.5. Provide a solid foundation

Students must understand the underlying concepts well. (For example, well-formedness, W3C XML Schema datatypes, validity against a schema, or the evaluation of XPath location paths.)

6. Tools used

Its platform independence and vendor neutrality are the most important features of XML. Teaching XML is similarly not bound strictly to any specific hardware or operating system environment. However, working with XML in real life requires expertise in the use of several different software tools, such as programming language APIs, editors and XSLT processors.

Since the author is an open source fundamantalist and also a Linux enthusiast, he always prefers open source solutions to proprietary software. There are many advantages of using free software in education. Among others open source software solutions often compete with their proprietary competitors in quality while they represent a lower-cost alternative. It is also very important that students can use the same software at school and at home on their computers. See [4] for more information on these issues.

There are excellent free software products to work with XML, due to the lack of space they are not mentioned here by name.

The Java programming language is an ideal environment to implement XML applications, since many of the XML processing APIs are part of the Java platform currently. Java SE 6 [14] that is the current major release of the Java SE platform provides a comprehensive set of XML APIs. Moreover, Java SE 6 is now open source, licensed under the GNU GPL license.

The following XML APIs are shipped with the platform:

- Java API for XML Processing (JAXP) [12]
- Document Object Model (DOM)
- Simple API for XML (SAX)

- Streaming API for XML (StAX)
- Java Architecture for XML Binding (JAXB)

These APIs support the core XML standards and enable developers to build XML-aware applications. All of the above mentioned APIs are covered in the course.

Unsurprisingly, bulk of the existing XML related software products are also implemented in Java.

Since Java SE 6 now includes the most important and most widely used XML APIs providing a comprehensive XML support, only a few additional software products should be introduced in the classroom.

Although it is not open source (only a 30-day trial version is available), <oXygen/> [23] is an excellent XML editor with many features. It is written in Java and runs on multiple platforms. It may be one of the best XML editors that are available today.

The editor provides support for W3C XML Schema that makes it easy to teach W3C XML Schema. It has context sensitive schema editing capabilities that helps students work with XML Schema documents. Without such a feature the creation and editing of XML Schema documents should be an extremely painful job. <oXygen/> also functions as an XSLT editor and debugger, and can be used very effectively in teaching XSLT. Another great advantage of <oXygen/> is that it is also available as an Eclipse [5] plugin. Thus, its entire functionality is accessible in the popular IDE. (Eclipse is the recommended Java IDE in the XML course.)

Unfortunately, the author has no knowledge of any open source alternative that is as good as <oXygen/>. Conversely, there are many other commercial alternatives. Although it might seem that <oXygen/> has been chosen arbitrarily, that is not a problem at all, since these tools are quite similar in both features and usage. For example, see the excellent EditX XML editor [6] that provides almost the same features and functionality as <oXygen/> does. An <oXygen/> user will become acquainted with EditX with ease.

7. Sample projects

In this section two sample projects are presented that have been developed by the author to teach XML processing in the classroom. It must be emphasized that both are realistic and quite complex, and students like them very much.

7.1. Object Relational Mapping

The goal of the project was to implement a simple Object Relational Mapping (ORM) Java library. ORM is a popular technique in the OO world to help with persistence related programming tasks.

An object model is mapped to a relational data model in order to provide transparent object storage and retrieval functionality. Object states are stored in relational database tables but SQL is hidden deeply in the implementation, that provides high level services to automatically store, query, update and delete objects. Moreover, the programmer need not to use SQL at all since SQL statements are generated and executed automatically.

ORM tools are often used in application server environments to implement the business logic tier, and they can make development more easier and also less time consuming.

Hibernate [10] is a typical representative of the family of ORM tools, that may be the most widely used and at the same time the best ORM solution today in the Java world. Hibernate is open source, distributed under the LGPL license, and shipped also with the JBoss application server.

Like the other ORM tools, Hibernate uses XML to describe the mapping of an object model to a relational data model. Java classes are mapped to relational tables and their fields to table columns. XML documents with their hierarchic structure are well suited to describe such mappings. In a mapping document a Java class is represented by a single XML element, and each child element of such an element maps a field to a table column.

It is a very hard and challenging task to implement a general purpose ORM tool. However, a skilled programmer can develop a small working prototype quite easily. Our implementation consists of just a few classes (12 classes, including 5 exception classes) and the total number of source code lines does not exceed 1000. Of course it is just a toy compared to Hibernate but demonstrates well the ideas and techniques that are also used in the professional solutions.

This toy implementation has the following assumptions and limitations:

- Each class must follow the Java Bean design pattern (fields are accessible via getter and setter methods).
- Fields must have a "built-in" data type (the basic JDBC types are supported).
- Each class is mapped to a single relational table.

- Composite keys are not supported, each class must have a simple primary key.
- Key generation is not supported, primary keys must be assigned manually.

It also lacks many of the features of the professional ORM products, such as associations, object cache and query language.

The key techniques that are used in the implementation are XML parsing, JDBC [15] and reflection. Configuration data (e.g. database password) and class mappings are also stored in XML documents.

The operation of the ORM toy is summarized below.

The ORM toy is initialized parsing the XML documents. The StAX API has been chosen to extract the information and to build an in-memory object model of the mappings. Since our XML documents have a simple and regular structure, it is a very easy task even for a beginner.

Based on the mappings parameterizable SQL statements are generated automatically to each class, that will store, load, delete and update objects in the database.

When an object must be stored the fields that have been given in the mapping are accessed using reflection, the field values are substituted into the corresponding precompiled SQL INSERT statement and then executed.

Objects can be loaded by their primary key values. Passing the class and the primary key value to the load function an appropriate SELECT SQL statement is executed that must return a single row. Then an empty object is instantiated, whose fields are set using reflection according to the column values. (The update and delete operations are implemented similarly.)

The author thinks that the implementation is very elegant that uses quite standard and simple techniques to implement an advanced functionality. It is also a nice example of the power of XML.

7.2. Interpreting a program encoded in XML

The goal of the project was to design a simple programming language that uses XML syntax and to develop an interpreter that can execute programs encoded in XML.

The idea is not new, there are several programming languages that use XML documents to represent programs. Examples are o:XML [22] and MetaL [18].

o:XML is a fully-featured object oriented programming language. o:XML programs can be executed by an interpreter or can be transformed into another language (into Java, C++ or another target language).

Programs written in MetaL (Meta Programming Language) can be considered as higher level specifications of programs. These XML programs can be used to generate source code to potentially any target language. (Currently PHP, Java and Perl are supported.)

This approach has many advantages:

- A programming language of this kind can be considered as a meta-programming language, since XML programs can be transformed into other languages (e.g. Java, C++). The transformation can be done simply by XSLT.
- Programming language syntax can be defined by an XML schema. Legal programs are XML documents that are valid against the schema. Schema validation can be performed by an XML parser. Moreover, the parser can also build an in-memory representation of the program that can be treated as a parse tree.
- Using the namespace mechanism metadata (e.g. documentation) can be added to the source code without the need of changing the grammar that allows a flexible language extension.

Although they are not yet wide-spread, these programming languages could be applied very effectively in different application areas, such as web services, XML publishing, and server-side scripting.

The syntax of our toy programming language is defined by a DTD. (There is also an XML schema that specifies the grammar of the language.) The language supports the following language constructs and features:

- expressions
- assignment
- while loop (the break and continue statements are also supported)
- if-elseif-else
- I/O (output only).

The language lacks many features of the real programming languages. Among others, there are only global variables and it does not allow the definition of functions and procedures.

Instead of defining an XML expression syntax that might be cumbersome to use the interpreter uses a third-party JavaScript library to perform expression evaluation [24].[4]

Legal programs are parsed into a DOM tree that acts as a parse tree and execution is performed by traversing the tree using the standard DOM methods.

The students themselves have been extended the language and the interpreter with advanced features such as procedures, functions, variable scopes and exception handling.

The primary goal of the author was to demonstrate how to use the DOM application programming interface effectively, and not to implement a general purpose fully-functional programming language. In his opinion, the program is a nice example of that how to implement a quite complex behavior (i.e. program execution) using standard programming techniques. It is worth to note that the total number of lines of the source code is approximately 400.

8. Conclusions

The paper provides a detailed overview of an university-level introductory XML course. The course has been very popular among students since its first debut at our university. The author thinks that it is not surprising since students can meet the most up-to-date technologies in the classroom. He has got countless feedbacks from graduate students now working in the industry. They all say that they make good use of the knowledge that they have gained on the course that is very inspiring for the author.

As mentioned in section 2, separate courses may be dedicated to many of the XML standards and applications. It would be highly desirable to announce courses on them, since they are the cutting edge of technology and would attract a wide audience for sure. These goals have been at least partially fulfilled in our faculty, since it now offers a course on Semantic Web.

XML education is changing at our faculty, as BSc-level education has been just introduced. Computer science students who started their studies in this semester have to attend a course titled XML, HTML obligatory now. First-year students do not have programming skills, they attend the course in parallel with

[4]Now Java SE 6 provides scripting support, it includes the Rhino JavaScript engine for the JavaScript programming language.

an introductory computer science course. Therefore, the new course roughly corresponds to the first half of the "old" XML course, but some topics (e.g. CSS) are discussed in more details. Since first-year students will have solid XML foundations after the completion of the new course, the "old" course may function as an advanced level XML course. One half of the lectures may be devoted to Java programming entirely as before, but other important and interesting topics, such as XQuery and native XML databases may be covered also.

References

[1] Neil Bradley, *The XML Companion*, 3rd edn., Addison-Wesley, 2001.

[2] Document Object Model (DOM), `http://www.w3.org/DOM/`.

[3] Extensible Markup Language (XML) 1.0, `http://www.w3.org/TR/REC-xml/`.

[4] Free Software in Education, `http://www.gnu.org/education/`

[5] Eclipse `http://www.eclipse.org/`

[6] EditX http://www.editix.com/

[7] Elliotte Rusty Harold, *Effective XML: 50 Specific Ways to Improve Your XML*, Addison-Wesley, 2003.

[8] Elliotte Rusty Harold, *Processing XML with Java: A Guide to SAX, DOM, JDOM, JAXP, and TrAX*, Addison-Wesley, 2002. `http://www.cafeconleche.org/books/xmljava/`

[9] Elliotte Rusty Harold, W. Scott Means, *XML in a Nutshell*, 3rd edn., O'Reilly, 2004.

[10] Hibernate `http://www.hibernate.org/`

[11] ISO 8879:1986 Information processing – Text and office systems – Standard Generalized Markup Language (SGML)

[12] Java API for XML Processing (JAXP) `http://java.sun.com/webservices/jaxp/`

[13] Java Architecture for XML Binding (JAXB) `http://java.sun.com/webservices/jaxb/`

[14] Java SE 6 `http://java.sun.com/javase/6/`

[15] JDBC `http://java.sun.com/javase/technologies/database/`

[16] JSR 173: Streaming API for XML (StAX) `http://jcp.org/en/jsr/detail?id=173`

[17] Salvatore Mangano, *XSLT Cookbook, Second Edition*, 2nd edn., O'Reilly, 2005.

[18] MetaL: An XML based Meta-Programming language `http://www.meta-language.net/`

[19] Eric Meyer, *CSS: The Definitive Guide*, 3rd edn., O'Reilly, 2006.

[20] Chuck Musciano, Bill Kennedy, *HTML & XHTML: The Definitive Guide*, 6th edn., O'Reilly, 2006.

[21] Namespaces in XML http://www.w3.org/TR/REC-xml-names/

[22] o:XML http://www.o-xml.org/

[23] <oXygen/> XML Editor & XSLT Debugger http://www.oxygenxml.com/

[24] Rhino: JavaScript for Java http://www.mozilla.org/rhino/

[25] Simple API for XML (SAX) http://www.saxproject.org/

[26] Jeni Tennison, *Beginning XSLT*, Apress, 2004.

[27] XML Path Language (XPath) Version 1.0 http://www.w3.org/TR/xpath

[28] XSL Transformations (XSLT) Version 1.0 http://www.w3.org/TR/xslt

[29] W3C-translators levelezőlista
http://acel.dsd.sztaki.hu/mailman/listinfo/w3c-translators

[30] W3C XML Schema http://www.w3.org/XML/Schema

[31] XML course page by Peter Jeszenszky http://www.inf.unideb.hu/~jeszy/xml/

PÉTER JESZENSZKY
UNIVERSITY OF DEBRECEN
FACULTY OF INFORMATICS
DEPARTMENT OF APPLIED MATHEMATICS AND PROBABILITY THEORY
H–4032 DEBRECEN
EGYETEM TÉR 1.
HUNGARY

E-mail: jeszy@inf.unideb.hu

(Received January, 2007)