An XML-based Framework for the development of Adaptive Educational Software

Lefteris Moussiades
Technological Educational Institute of Kavala
Dept. of Industrial Informatics,
Div. of Computing Systems
GR- 65404 Kavala, Greece
lmous@teikav.edu.gr

Athena Vakali
Department of Informatics
Aristotle University
54124 Thessaloniki, Greece
avakali@csd.auth.gr

Anthi Iliopoulou
University of Sheffield
Department of Educational Studies
Sheffield S10 2JA, UK
edp02ai@sheffield.ac.uk

ABSTRACT

Large-scale adaptive educational software requires advanced technological infrastructures in order to support various cooperative educational organizations. This paper proposes a framework for overcoming the difficulties that arise due to the distribution and heterogeneity of current collaborative educational technologies. The proposed framework is based on XML it is of distributive and cooperative nature since it is characterized by the following:

- Each educational organization may plan its development to meet its own needs.
- It is possible for each organization to *concurrently* reuse the software that has already been developed by any other organization,
- Each organization can combine its own software production, with the production of other cooperative organizations, in various ways.

I. INTRODUCTION

One of the basic issues in the software development process is the maximization of reusability. This issue is of a great importance when trying to develop adaptive educational software that serves the needs of various cooperative organizations. For example, if we consider the various departments of a University, it is likely that many of them provide the same cognitive object in their curriculum, but they are supposed to present it in a different manner (because of their varied educational objectives and the different knowledge background and orientation of their students). In this context, separate software development may be rejected (as very expensive), whereas central development may be problematic (as very complex).
By the term *adaptive educational software* we refer to Web-based Adaptive and Intelligent Educational Systems (AIES) [1], which are tutoring systems that operates over the web, support on-line asynchronous education and encapsulate various forms of adaptivity to the learner's special needs. Such systems are characterized by high complexity (in their implementation phase), whereas they become even more complex when they are designed to support the operational needs of more than one educational organization. In such a case, the developer of the system has to consider all the different needs of the participant organizations, as well as all their common features. Moreover, the above requirement has to be met in the design phase so that it assures the corporation of various members of the educational organizations. In spite of the extensive use of XML for developing commercial sides, it appears that its application to authoring educational content has not been investigated sufficiently yet

[2]. Recent work on the area that combines XML with educational software includes: definition of specialized languages for the development of various educational content [3], XML's comparison to static HTML in the dynamic generation of the educational content [4], test tools developed in XML [5], etc. In our perspective XML should be utilized as the primary development tool for complex adaptive educational software, as it allows distributive in time and space development eliminating the potential code conflicts.

In this paper, we present XML's mechanisms in order to achieve :

- Code reusability
- Differentiation of the content
- Distributive development through the elimination of variable names conflicts.

The remainder of the paper is structured as follows: In section 2 after we discuss traditional software engineering approaches to the development of complex systems and state out their disadvantages, we point out the characteristics of a more elegant solution that is available through the use of XML and enables for distributive and cooperative software development. In section 3, we present a case study that concerns different university courses, which invoke in their curriculum the same cognitive subject but they have to include different content. In section 4, a presentation of the suggested solution is given, along with some basic concepts of XML, which are essential to the comprehension of the suggested solution.

2. EDUCATIONAL SOFTWARE DEVELOPMENT

2.1. Central and distinct approach

A popular approach to the development of complex educational software is the development of a parameterized system with central design Such a solution, has some significant disadvantages, such as:

- **High complexity level**. A central design, which tries to parameterize the set of possible differences, it is obvious that has high complexity level. This becomes even more complex, in case that other departments as well require educational software for this specific knowledge domain.
- **Luck of flexibility.** A central design should predict all the diverse differences at the design time. The changes that are made to large-scale software after its design and implementation phase are difficult and constitute a source of errors. Nevertheless, predicting all the different needs during the design phase is extremely difficult. Even in the ideal case where something like this is accomplished, the possible future needs will be a factor of inconsistency for the whole system.
- **Excluding the teacher**. According to the general principles of the pedagogical science, one of the most important factors of the learning experience is the tutor. In traditional education, the tutor participates in defining the learning goals, in creating the curriculum, in designing the didactic units and the learning content. The tutor also chooses the **visual aids**, the examples and the exercises, he conducts the assessment, he designs the time schedule of the learning process etc. Despite the rapid growth of technology in the last years, it is not likely that the tutor will be deprived of the above work. Nevertheless, the central design of

educational software excludes tutors from defining its functionality and the elements of the user interface that the tutor might be more familiar with or that he believes that will facilitate the realization of learning goals.

On the opposite side of central development, there exists distinct software development. This is of course an expensive approach, since a great part of the development phase is repeated, while it introduces a high difficulty level in communication between the systems of educational software of different institutions. This communication is in many cases desirable.

2.2 Distributive and cooperative development

As a solution to the above described disadvantages of central and distinct software development, we suggest distributive and cooperative development which is identified by

- The possibility, it provides to every organization to develop educational software according to its preferences and particularities, along with central control, which will reassure that communication between the educational software of different organizations, will be easy (distributive development).
- A function of full reusability of the material produced by each organization, so as to avoid pointless repetition in the phase of development and also eliminate the costs (cooperative development).
- The use of XML-namespaces mechanism to eliminate conflicts in variable names providing thus the possibility for distributive and cooperative development

In the proposal presented in this paper, the XML syntax is used since the required functionality may be explicitly described and an XML model that serves this functionality is analyzed.

The XML (eXtensible Markup Language) is the applied model of SGML (Standard Generalized Markup Language). Each document of XML by default complies with the rules of SGML [6]. Despite the fact that XML is a Markup Language, it doesn't define its own tags. On the contrary it provides the user with the possibility to define the elements and the attributes that he considers as being necessary [7].
XML is sustained by a series of supportive mechanisms. We will mention three of these mechanisms:

- XML schemas. The purpose of a schema is to define a class of XML documents [8]. Schemas by themselves are XML documents, which define the syntax that other XML documents should comply with.
- XML namespaces. A mechanism is provided which relates the names and attributes of the elements with a URI [9]. In this way a possibility is provided to define unique names for the elements and their attributes.
- XSLT (eXtensible Stylesheet Language Transformation). This is a language, which allows for the transformation of XML documents to XML documents of some other type [10]. One of the main usages of XSLT is to transform XML documents so as these to become appropriate for transmission through an output device. In this way it is possible to separate data from their presentation.

By using the XML schemas it is possible to define a series of Markup Languages that focus on the development of some part of the educational software. A similar approach is presented by Maria Elena Bonfligi [3], who developed a Student Markup Language and a Domain Structure Markup Language, in order to implement adaptive hypermedia. This kind of language is also known as vocabulary, since it effectively defines a set of XML elements (tags). With the help of namespaces, it is possible the elements of vocabularies to be reusable [7]. These vocabularies constitute the base of the cooperative development upon which a set of XML documents and style sheets is added. The set of the above elements constitutes the first active educational software. Every organization that participates in this, can do the required modifications, add or remove elements in all levels of schemas, documents and style sheets, without affecting the functionality of the software for the other organizations. At the same time an organization can use the work of any other. Even when new vocabularies are defined or new elements are added in the existing ones, the possibility of conflict between symbols or meanings is non-existent, thanks to the namespaces mechanism.

3. A CASE STUDY : UNIVERSITY COURSES

The proposed framework is presented through the use of an appropriate example in the context of University education. More precisely, suppose that our target is to develop educational software that has all the characteristics of the adaptive educational Software [1], it has as knowledge domain the C++ programming language and should cover the needs of an introductory course in programming, which is taught in the first semester of the School of Economics and applies to students that are not familiarized with computer programming. It should also cover the needs of a basic C++ course, which is taught in the 3rd semester of the School of Computer Engineering and applies to students that have already been taught the Pascal programming language.

It is obvious that these two goals have a lot in common, but they also have significant differences. Their common properties are related to the main characteristics of the contemporary educational software since in both cases we need:

- Knowledge representation
- Student modeling
- Adaptive functionality
- Automatic management of questions
- Automation of the exams
- Support of Cooperation

In addition, the knowledge domain is the same in both cases.

On the other hand, each case should be realized according to different terms. Thus, in the case of economists the basic concepts should be taught in more detail and examples with economical content should be presented. In the case of computer engineers, the basic concepts have already been taught so the emphasis should be on the details of the programming language. The examples that apply to computer engineers may include cases such as programming a device, which makes them inappropriate for the Economists. Except for the special content, the examples may have a different structure as well, since for each example in C++, which is demonstrated to the engineers, it would be helpful to provide the analogous example in Pascal.

4. THE PROPOSED SOFTWARE DEVELOPMENT MODEL

We present the proposed model under the example of the Economics department and the department of Computer Engineers. We consider that the Economics department wants to structure the knowledge base of C++, according to the syntax that is shown in Figure 1**.**

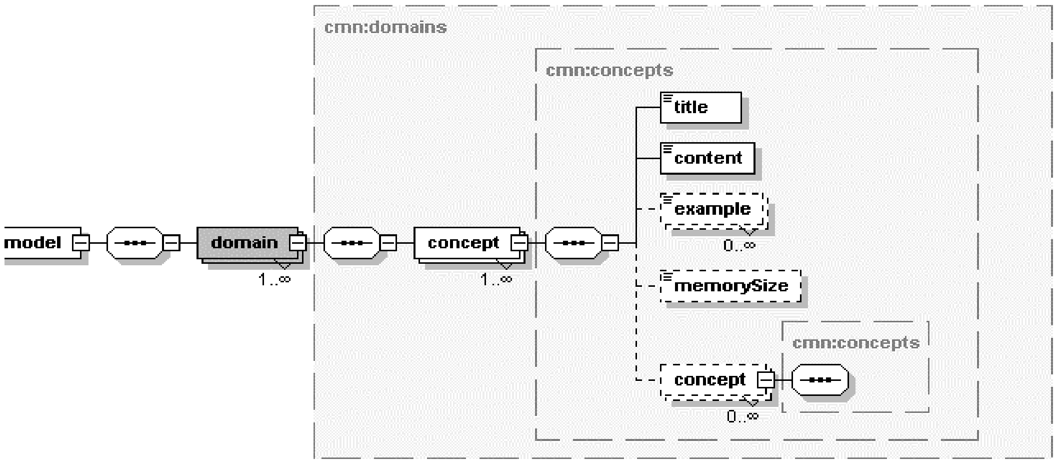

Figure 1 *XML schema defining the Economics department courseware*

Figure 1 represents an XML-schema for the proposed educational software model. The field "domain" corresponds to different cognitive areas that someone should be able to encode. In our example, we suppose the constant value C++. This syntax defines its basic element (concept) recursively. In this way it supports the production of documents, which organize their concepts hierarchically in trees with no depth limit. In addition there is no limit as to the number of concepts for each level. Thus we have the option to produce unlimited number of documents, which follow this syntax and represent the knowledge base of C++. Such a document is represented in figure 2. Each leave of the tree under the node C++(domain) represents a concept.

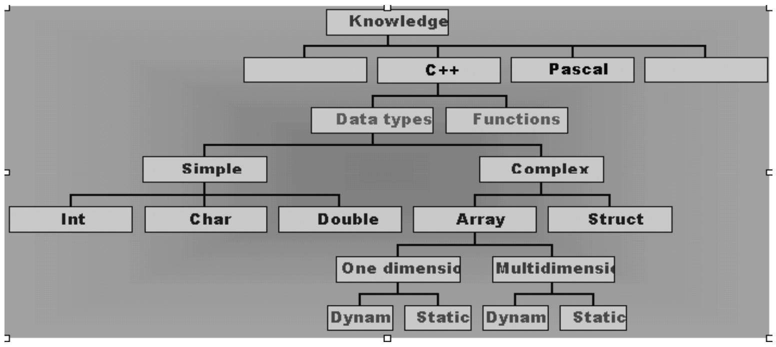

Figure 2 *XML document representation as defined according to the schema of figure 1*

It is obvious from Figure 1, that each concept consists of a title, a field that refers to its contents and one or more optional fields of examples. Each of these examples is a simple text field. Suppose now, that the department of Computer Engineering is interested in the software produced by the department of Economics, but the examples of the latter do not correspond to the needs of the former for two reasons. Firstly, for each example a Pascal analogous should be provided, since Pascal is a prerequisite lesson for C++ in the department of Engineering. Secondly a possibility should be provided to note the difficulty level for each example, so as to evaluate these examples in relation to the student model and achieve in this way the proper adaptive presentation. In this case the department of Engineering can use the vocabulary of the Economics department and develop its own as well, which consists in this particular case by only one tag that is the example element. The definition of this tag in XML-schema follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.teikav.edu.gr/bp"
        xmlns:bp="http://www.teikav.edu.gr/bp"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
 <xs:complexType name="exCPlus">
  <xs:simpleContent>
   <xs:extension base="xs:string">
    <xs:attribute name="difficultyLevel" type="xs:byte"/>
   </xs:extension>
  </xs:simpleContent>
 </xs:complexType>
  <xs:complexType name="examples">
  <xs:sequence>
   <xs:element name="CPlus" type="bp:exCPlus"/>
   <xs:element name="PascalAnalogous" type="xs:string"
```

According to the above XML schema code an appropriate example element is defined in the bp namespace. This vocabulary can be imported [11] in the basic syntax of Figure 1, resulting in the schema of Figure 3.

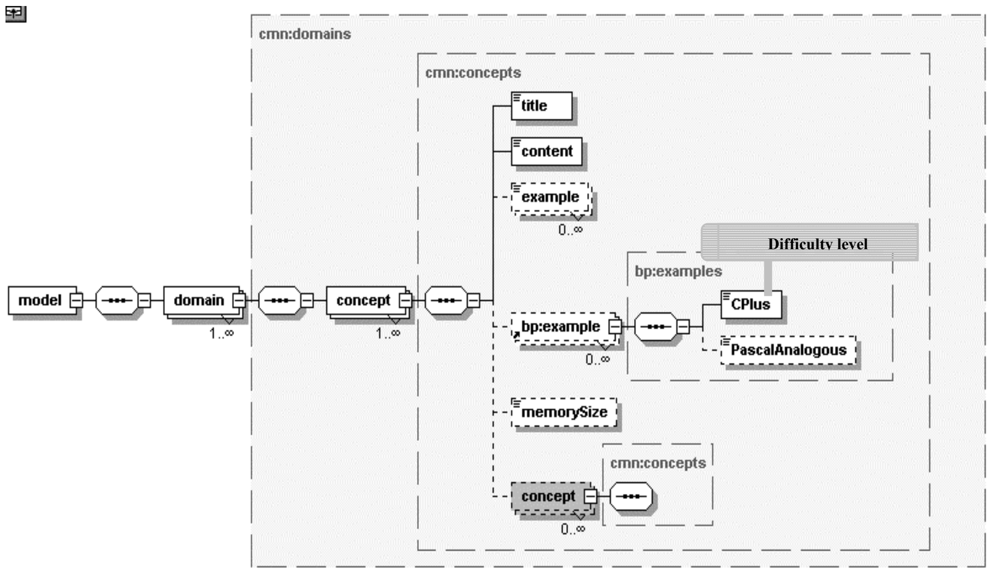

Figure 3 *Multiple XML schemas supporting both types of examples*

This is a multiple XML-schema that contains both types of examples and differentiates between them by the different namespaces that they belong to. The XML schema code is presented below:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.teikav.edu.gr/cmn"
 xmlns:cmn="http://www.teikav.edu.gr/cmn"
 xmlns:bp="http://www.teikav.edu.gr/bp"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified">
 <xs:import namespace="http://www.teikav.edu.gr/bp" schemaLocation="bp.xsd"/>
<xs:complexType name="concepts">
  <xs:sequence>
   <xs:element name="title" type="xs:string"/>
   <xs:element name="content" type="xs:string"/>
   <xs:element name="example" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
   <xs:element ref="bp:example" minOccurs="0" maxOccurs="unbounded"/>
   <xs:element name="memorySize" type="xs:byte" minOccurs="0"/>
   <xs:element name="concept" type="cmn:concepts" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
 </xs:complexType> <!-- concepts -->
<xs:complexType name="domains">
  <xs:sequence>
   <xs:element name="concept" type="cmn:concepts" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
 </xs:complexType>
<xs:element name="model">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="domain" type="cmn:domains" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
 </xs:element> <!--Global element-->
</xs:schema>
```

This new syntax can support both of the considered cases. XML documents based on this multiple schema can reference either of the defined examples. Since the

Economists example is defined in the default namespace, it needs no namespace identifier. Thus, existing documents may remain unaltered. The Computer Engineers example may be added to existing documents identified by the bp namespace or new documents may be created.

It is worth to note that although both departments have used the symbol "example", to name the corresponding element, there is no conflict between them, provided that there is proper namespace identification in the XML documents. One belongs to the default namespace and the other belongs to the bp namespace. Of course each department should develop its own stylesheet, in order to design its own presentation and to pick up its own example. Furthermore the Computer Engineer department should have to implement adaptivity according to the difficulty Level that characterizes each "bp: example". In order to accomplish this, information of system's student model should be passed as parameter to the stylesheet, using the xsl:param tag [12]. This information is compared with the corresponding information, which is cited in the XML documents by the use of proper xsl variables [13]. With the aid of xsl conditionals, the proper presentation data are selected. A sample xslt code that contains the key features of the described xslt functionality is presented below:

```
<?xml version="1.0" encoding="UTF-8"?>
 <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                         xmlns:n1="http://www.teikav.edu.gr/cmn"
                         xmlns:bp="http://www.teikav.edu.gr/bp"
                         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <xsl:param name="studentLevel">10</xsl:param>
   ....
   ....
   ....
  <xsl:for-each select="bp:example">
   <xsl:for-each select="bp:CPlus">
    <xsl:variable name="difficultyL"><xsl:value-of
select="@difficultyLevel"/></xsl:variable>
    <xsl:if test="$studentLevel>=$difficultyL">
     <p>
      <span style="font-weight:bold">e.g</span>&#160;<xsl:apply-templates/>
     </p>
    </xsl:if>
   </xsl:for-each>
   <xsl:for-each select="bp:PascalAnalogous">
    <p>
     <span style="font-weight:bold">Pascal Analogous</span>&#160;<xsl:apply-
templates/>
    </p>
```

The above portion of xslt code presents how student Level parameterizes the styleseet, how difficulty Level is loaded from the XML-document into xslt variables and how xsl conditionals adapt the presentation.

5. CONCLUSIONS

The typical software engineering problem of distributive and cooperative development has met a quite challenging solution in the context of educational software development within the framework of XML. Adaptivity that is considered a functionality of great importance to educational software, may also be implemented in the same frame of XML. Thus, XML should be considered as a first priority choice language for the development of complex adaptive educational software.

References

1 **Brusilovsky, P.**: Adaptive and Intelligent Technologies for Web-based Education: C.Rollinger and C.Peylo (eds.) *Kunstliche Intelligenz*, Special Issue on Intelligent Systems and Teleteaching, 1999.

2 **Claus Pahl: An Investigation of XML-technologies for Infrastructures for Web-based Virtual Courses :** *Poster Proceedings of the Tenth International World Wide Web Conference*, WWW 10, Hong Kong, China, May 1-5, 2001

3 **Bonfigli, M.E., Casadei, G. & Salomoni, P.**: Adaptive Intelligent Hypermedia using XML: *Proceedings of SAC 2000 - ACM Symposium on Applied Computing,* Carroll, J., Damiani, E., Haddad, H., Oppenheim, D. Eds., Villa Olmo, Como, Italy, Vol. 2, pp.922-926, March 2000.

4 **Buentia F, Benlloch J.V, Gill J.A, Agusti M:** XEDU, a XML-based framework for developing didactic resources: *12th EAEEIE conference*, Nancy, 2001

5 **A. Navarro, J.L. Sierra, B. Fernández-Manjón, A.Fernández-Valmayor**, XML-based Integration of Hypermedia Design Techniques and Component-Based Software Development for Building Educational Applications, *Computers and Education in the 21st Century.* Kluwer Academic Publishers 2000.

6 Extensible Markup Language (XML) 1.0 (Second Edition): *W3C Recommendation,* 6 October 2000.

7 **Benoit Marchal**: XML by example: Second Edition, Que, September 2001.

8 XML Schema Part 0: Primer: *W3C Recommendation*, 2 May 2001.

9 Namespaces in XML: *World Wide Web Consortium,* 14 January 1999.

10 XSL Transformations (XSLT) Version 2.0: *W3C Working Draft,* 15 November 2002.

11 **Brett McLaughlin:** Referencing multiple schemas from XML, and from other XML schemas*: http://www-106.ibm.com/developerworks/library/x-tipschnm.html*, September 2002**.**

12 **Nicholas Chase:** Use parameters and conditionals in your style sheets, *http://www-106.ibm.com/developerworks/library/x-tipxsltrun/index.html,* August 2002.

13 **Barry R. Beggs, Joan Boone**: Using XML and XSL for code generation*, http://www-106.ibm.com/developerworks/library/ibm-codegen/index.html#author2,* May 2001.