Adaptive Intelligent Hypermedia using XML

Maria Elena Bonfigli
Dip. Scienze dell'Informazione
Università degli Studi di Bologna
Via Mura A. Zamboni 7
40134 Bologna, Italy
Tel. +39.051.2094880
bonfigli@cs.unibo.it

Giorgio Casadei
Dip. Scienze dell'Informazione
Università degli Studi di Bologna
Via Mura A. Zamboni 7
40134 Bologna, Italy
Tel. +39.051.2094507
casadei@cs.unibo.it

Paola Salomoni
Dip. Scienze dell'Informazione
Università degli Studi di Bologna
Via Mura A. Zamboni 7
40134 Bologna, Italy
Tel. +39.051.2094880
salomoni@cs.unibo.it

ABSTRACT

In this paper we discuss the problems of developing Web-based Adaptive Hypermedia (AH) for engineering education using Extensible Markup Language (XML). AH systems are capable of altering the presentation of the content of the hypermedia on the basis of a dynamic understanding of the individual user [5]. The user profile can be collected in a *user model*, while the knowledge about the domain (the content of the hypermedia) can be represented in the form of a concept-based *domain model*. So we have defined two different markup languages using XML: the former for structuring the domain model and the latter for describing the student model. These languages can be easily extended and authored, with the result of obtaining a simple methodology for data structuring in the field of Web-based educational AH.

Keywords

adaptive hypermedia, educational hypermedia, intelligent tutoring system, user modeling.

1. INTRODUCTION

With the rapid advances in WWW interactive technologies, the use of Internet-based distance learning tools is rapidly growing. Most of these products are nothing more than a network of static hypermedia pages. In fact the domain knowledge implicit in traditional Web-based educational hypermedia is well defined and carefully structured and provides an only learning path optimal for a generic average student [5]. Otherwise, a Web application, which is designed with a particular class of users in mind may not suit other users. Moreover, "static" hypermedia assume that the users can make sensible decisions about when to use navigation tools, about when to proceed in the learning process, about when they need an explanation, etc. [3]. This could be a problem for those users who access the hypermedia through the Internet and that can’ t have a

teacher at their disposal.
Adaptivity is the feature of hypertext and hypermedia that allows one to adapt the contents to the user needs [1]. AHs systems modify the presentation of the domain knowledge according to the user profile. This mechanism permits to personalize the Hypermedia in terms of contents and of navigation tools for each user.
The focus of this paper is on the general architecture and the implementation issues of a Web-based educational AH. This system has been implemented by using the functionalities provided by XML in order to stress the separation of the information content from presentation. The basic idea is to define a general hypertext structure in order to create pages dynamically using a structured description of the domain knowledge and a model of the current user.

2. SYSTEM ARCHITECTURE

The AH system architecture (see Figure 1) consists of [9] three main components: a student model that represents the student knowledge; a *domain model* that contains the domain knowledge information structured in topics and relationships; a *dynamic management module* that is responsible for creating pages dynamically by using the information contained both in the domain and in the user modules.

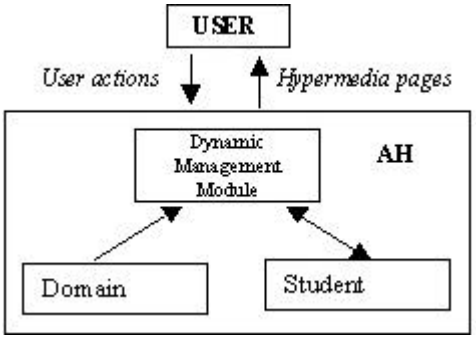

Figure 1: System architecture

XML is a meta-language that allows us to create specialized markup languages for specific purposes [8]. XML is a subset of the more general Standard Generalized Markup Language (SGML) and represent a "de facto" standard for developing specialized markup languages. For example, it is possible to create a tag <TOPIC> (and the closure tag </TOPIC>) that identifies a particular knowledge element.

For implementing our AH system we have implemented two languages:

- the Domain Structure Markup Language (DSML) in order to describe the knowledge general structure,
- the Student Model Markup Language (SMML) used by the system in order to describe the student model status.

In particular we have implemented two different Document Type Definitions (DTDs) that define the fixed structure of the two above mentioned markup languages. In the following two subsections we will give some examples of the new tags that we have

introduced. Finally in the last subsection we describe the Dynamic Management Module.

2.1 Domain Knowledge Structure

To be dynamically processed, the domain knowledge has been structured into *topics* and each topic is associated to a page. Each topic is characterized by:

- a *title*,
- a set of main keywords, called *concepts*,
- a set of conditioned *relationships*,
- a set of *explanations*,
- a set of possible *examples*, *exercises*, etc.

Therefore a link from a page to another one represents a relationship between the two topics. We have considered six different types of relationships between topics [9]:

- *son/father* that is a bi-directional relationship between a topic and all its sub-topics; for example the topic "array" is a son of the topic "data structures" (and on the other hand the topic "data structures" is the *father* of the topic "array");
- *next* that is a relationship between a topic and the next topic to learn;
- *contrary* that is a bi-directional relationship between a topic and the "opposite" one. For example the topic "top-down programming" is the *contrary* of the topic "bottom-up programming" and vice-versa;
- *example* that is a relationship between a topic and an example that explains it;
- *widening* that is a relationship between a topic and a study in depth of it;
- *exception* that is a relationship between a topic and an exception or a particular case.

The first relationship (*son/father*) defines the hierarchical structure of the knowledge domain. Knowledge is represented as a tree in which each node identifies a topic and a root topic is defined. The *next* relationship represents the sequence of topics suggested by the teacher. All the other relationships create irregular connections between topics that transform the original tree-scheme in a graph. The student can navigate the graph following all the different types of relationships and defining a personal learning path [9].
Each relationship is associated to a cluster of *conditions* that defines which kind of requirements the student needs to follow the link. If the student model satisfies all the conditions the link is displayed. The link associated to the son/father relation always appears without requirements. All the other links can be conditioned, and in particular the teacher links can have different arriving pages in function of the student model status.
In this way each node is associated to a set of next links, each one valid under a different set of conditions. The condition scheme is organized as follows.
Each *condition* is divided in different *condition terms* composed with *and* and *or* operators. We have considered that students are classified in three stereotypes and we have associated to each argument that needs to be known a stereotype and a certain

minimum knowledge level. So that a condition term can be read as: "*if the student of a level X have got a knowledge level Y on the argument A, then the condition term is true*". By composing condition terms with and/or operators we can obtain complex didactical requirements.

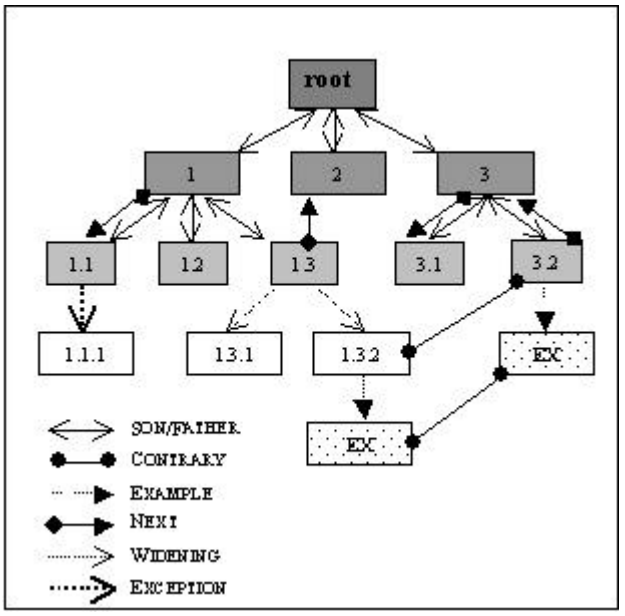

Figure 2: A simple example of domain knowledge structure

The DTD that describes DSML contains the definition the following tags:

- `<ADAT></ADAT>`. is the root of the adaptive hypertext. The `<ADAT>` tag can contain different `<TOPIC>` tags. By using `<ADAT>` attributes the author can also define the `TITLE`, the `AUTHOR` name and other proprieties of the hypermedia.
- `<TOPIC></TOPIC>` represents a topic and is identified by using the attributes `TITLE` and `ID`; each `<TOPIC>` can contain different relational tags (`<PARENT>` and `<SUBTOPIC>` described below), different conditioned elements (introduced by using the `<CONDITION>` tag) and different `<CONCEPT>` tags.
- `<CONDITION></CONDITION>` defines the requirement needed to follow a certain link or to give a certain explanation. Nested `<CONDITION>` tags can be used to realize *and* between different conditions; an attribute `OR` associated to a `<CONDITION>` tag is used to realize an *or* between this condition and the following one. Each set of nested conditions contains an `<EXPLANATION>` or a relational tag.
- `<EXPLANATION></EXPLANATION>` contains the real explanation of the topic. The `<EXPLANATION>` tag can contain any HTML tags. This choice lets one use different media to give the explanation to the student.
- `<CONCEPT></CONCEPT>` defines a keyword associated to the `<TOPIC>` tag that contains the `<CONCEPT>`.
- `<EXAMPLE></EXAMPLE>` defines a particular kind of topic that can be used as an example of a certain `<TOPIC>`; a `<EX-LINK>` relationship links this `<TOPIC>` to the `<EXAMPLE>`.

- <EXERCISE></EXERCISE> defines an exercise. The <EXERCISE> tag contains a <TEXT> a <SOLUTION> and an <HELP> tag.

The following code fragment represents a simple example of the DSML tags.

```
<TOPIC>
        Topic Title
        <CONCEPT> Keyword</CONCEPT>
        <CONDITION USER_LEVEL="3">
                    <EXPLANATION>
                          ...
                          HTML Explanation description
                          ...
                    </EXPLANATION>
        </CONDITION>
        ...
</TOPIC>
```

Three different type of tags are used:

- <CONCEPT>: that includes the keyword related to the topic.
- <CONDITION>: this tagindicates a didactical requirement. In this case there is a very simple condition based on the student' s knowledge level. Here an expert level (USER_LEVEL=3) is required. The student model, illustrated in the next section, manages values of the USER_LEVEL variable.
- <EXPLANATION>: this tag includes the real explanation of the topic. Text between this tag and its closure can be written in HTML.

Other tags are defined in order to describe relationships between the topics and in particular:

- <PARENT> and <SUBTOPIC> are used to represent the son/father relationship. We have used two tags to describe the relation, one for each direction.
- <NEXT> is used to define the next relationship and represents the path suggested by the teacher; the previous topic is dynamically computed.
- <WIDENING> defines a widening relationship.
- <EX-LINK> represents the relationship between a <TOPIC> and an <EXAMPLE> related to it.
- <CONTRARY> defines a contrary relationship.
- <EXCEPTION> represents an exception relationship.

2.2 Student Model

The student model is a structure maintained by the system with the aim of being able to respond to the individual student's learning style and deliver customized instruction. There are different ways to approach the student model problem.
The most commonly used student modeling technique in AH systems is the overlay model [5]. In this model, the student knowledge is considered an overlay of the domain knowledge and is frequently represented as a set of concept/value pairs. For each domain model concept the student model stores some values that estimate the student knowledge level of this concept. This model assumes that the student knowledge is a

subset of the knowledge domain described by the teacher and cannot capture different knowledge conception. In spite of this, the overlay model is commonly used in the adaptive hypertext field because the content of the adaptive hypertext itself can be considered as a description of the entire domain knowledge. This involves that the student can be represented simply overlaying the hypertext structure.
Other adaptive hypermedia systems use simple stereotype model [5]. Student knowledge is represented as a set of concept/value pairs (as in overlay model), but the values are not completely independent. The student can be assigned to one or more class (called stereotypes) and each class is identified by a fixed set of concept/value pairs.
On the other hand the most part of the Intelligent Tutoring System [2] are developed using a representation of the student's knowledge based on student bugs and misconception. The buggy model represent the student's knowledge in terms of deviations from an expert's knowledge. The system classifies each student error and then uses this information to predict the student' s behavior in future situations. The buggy model is really useful in learning-by-doing systems in which the system needs to monitor a student activity (as for example, exercises solving procedure).
To implement the student model we have used a mix of the overlay model and of the stereotype model.
The overlay model has been implemented associating to each topic a value that weights the level of the student knowledge of that topic. This evaluation is given measuring the time spent reading the topic and the number of visits. Each time the student requests a page, the associated value is updated.
We have also classified possible students into three different stereotypes: expert user, medium-level user and beginner user. This evaluation refers to all the arguments of the domain model and is a general assessment of the student ability. We have used a scheme based on three stereotypes referring to different system described in well-known literature [7, 5]. Anyway our system can recognize each set of integer values as a stereotype scheme.
The student model is realized using two DTDs: the former defines a structure that contains the user's logins and the related passwords, the latter defines the structure for the SMML in order to describe the personal student model file. The DTD that describes SMML contains the definition of the following tags:

- `<MODEL></MODEL>` defines all the visited topics described by using different `<VISIT>` tags and a `<LEVEL>` tag.
- `<VISIT>` is used to represent a visited topic; the argument is specified using the `TOPIC` attribute.
- `<LEVEL> </LEVEL>` and `<LEVELT> </LEVELT>` define respectively the general level of the student knowledge using the `DEGREE` attribute and the level of knowledge specific of the topic.
- `<LAST>` defines the last visited topic.
- `<NUMVISIT> </NUMVISIT>` defines the number of times that the student has visited this topic page

A simple example of student model is the following:

```
<MODEL>
    <LAST topic="1">
    <LEVEL degree="0">
    <VISIT topic="1">
        <LEVELT>0</LEVELT>
        <NUMVISIT>1</NUMVISIT>
    </VISIT>
</MODEL>
```

The example shows the student model of a beginner level user (`LEVEL DEGREE=0`). In the <`MODEL`> structure there is only one `VISIT` tag meaning that the student has just run the hypermedia and has visited only the first page identified by the `topic="1"`.The `LAST` visited topic is obviously the number "1". The example shows the student model of a beginner level user (`LEVEL degree="0")`. The nested tags `<LEVELT>` and `<NUMVISIT>` identify respectively the level of knowledge of the topic and that the student visits this topic page for the first time.
It is worth to note that the student models are automatically created and managed by the system so that the SMML is used only by the system to store relevant information.

2.3 Dynamic Management Module

The Dynamic Management Module (DMM) creates the pages of adaptive hypermedia on the basis of the Knowledge Domain Model and the Student Model.
The general architecture of DMM consists of four main components: the User Action Management Module(UAMM), the Adaptivity Management Module (AMM), the Domain Model Management Module(DMMM) and the User Model Management Module (UMMM) [9].
In particular:

- The UAMM interprets the student's behaviors sending information to the other modules (e.g. possible changes to *Student model* status to the UMMM; student's request of a page to the AMM).
- The DMMM retrieves topics from the *knowledge domain model* on the basis of the student's requests and interactions.
- The UMMM updates the student model sending the related information to the other modules.
- The AMM adapts the presentation of the domain knowledge to the user; creating hypermedia pages dynamically. In particular each time a condition is found, the AMM asks the UMMM the Student Module status in order to solve the condition and create the new page on the basis of the topics already known by the student (provided by the DMMM) and of his knowledge level.

3. IMPLEMENTATION ISSUES AND FUTURE WORKS

We have described the developing of two new markup languages that allow us to describe complex knowledge structures. It's worth to note that the system is designed in

order to realize adaptive educational hypermedia, but it can be used for every kind of adaptive hypermedia thanks to the flexibility of the used structures.
Our AH system has been implemented using a client-server scheme based on HTTP. The server application that automatically produces personalized XML pages has been described in the previous sections. The client is a common used browser, Microsoft© Internet Explorer© that can evaluate and display XML pages. Before starting the navigation in the AH system, the user has to identify himself with a username and a password. The server uses this information to store the appropriate student model (or to create a new student model for a new user). Then, each time the student asks for a new page, the browser asks for it to the server side application that compute the content of the page dynamically on the basis of the student model. A Java applet, which communicates to the server application all the events generated by the browser, updates the student model status.
We are currently testing an AH system on programming, created with XML dynamic support, that is developed for a University undergraduate program course. This test has underlined different aspects of the system that need to be improved

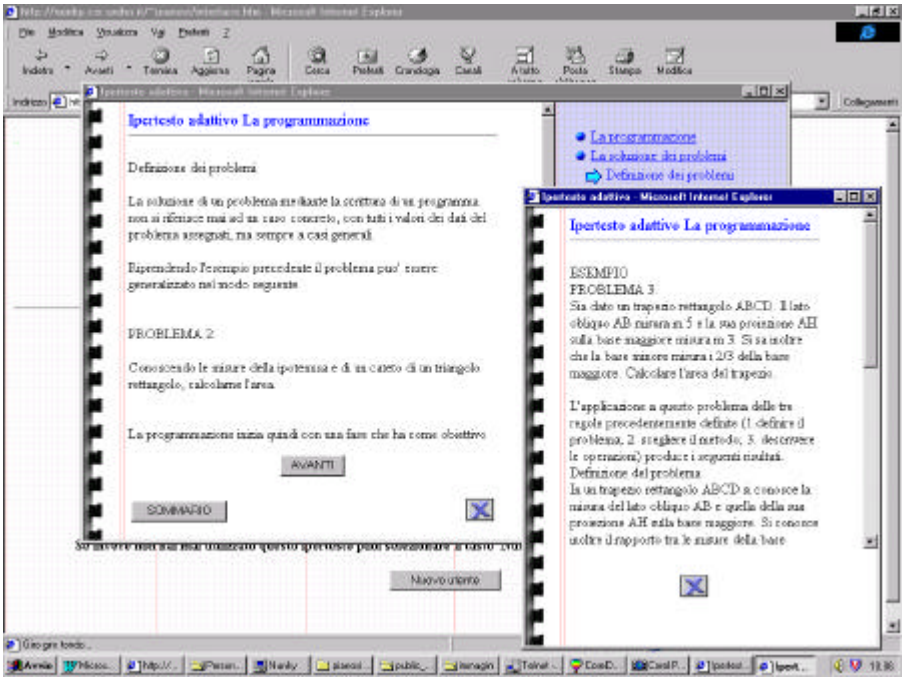

Figure 3: A snapshot of the Web-based educational AH on programming

In particular we are working in order to improve the student model and the user interface.
We are working on the student model to insert:

- an entry test in order to evaluate the initial level of knowledge of the student. This feature can improve the value of the DEGREE attribute to the LEVEL tag that implements the stereotype student model.
- A set of tests related to each main topic in order to modify the LEVEL tag associated to each topic. This feature can improve the overlay student model.

Evaluation tests can be closed answer tests, exercises that need to be solved, every different type of evaluation procedure that can give back an integer evaluation. The solving procedure associated to the exercises of the evaluation test can be monitored using every type of user model. In particular we are working on a Web-based educational AH system on Computer Architecture in which a set of exercises are based on the buggy model.
Moreover we are working on new user interface features to improve the navigation tools in the AH system [4]. In particular we are implementing an orientation map, that gives to the student a global idea of the domain knowledge organizing topics as *already known* and *not yet*.
Further works will consider the implementation of an authoring tool that will allow teachers to structure the domain knowledge without introducing DSML tags directly. This tool should be a WYSYWIG editor for developing AH courseware.

BIBLIOGRAPHY

1. Adaptive Hypertext & Hypermedia Home Page, http://wwwis.win.tue.nl/ah/
2. Anderson J.R., Corbett A.T., Koedinger K., & Pelletier R., (1995). Cognitive tutors: Lessons learned. The Journal of Learning Sciences, 4,167-207.
3. Brusilovsky P., 1998,"Adaptive Educational Systems on the World-Wide-Web: A Review of Available Technologies". In: Proceedings of Workshop "WWW-Based Tutoring" at 4th International Conference on Intelligent Tutoring Systems (ITS'98), San Antonio, TX, August 1998.
4. de La Passardiere B., Dufresne A., 1992, "Adaptive navigational tools for educational Hypermedia", I. Tomek (Ed), Computer Assisted Learning (pp. 555-567), Spinger-Verlag.
5. Eklund J, Brusilovsky P., Schwarz E., 1997, "Adaptive Textbooks on the WWW, in: Proceedings of AUSWEB97 - The Third Australian Conference on the World Wide Web, 186–192, Queensland, Australia.
6. Stern M., Woolf B.P., Kurose J.F., 1997,"Intelligence on the Web?", Artificial Intelligence in Education, IOS Press, 490-497.
7. Wenger E., 1987, Artificial Intelligence and Tutoring Systems, Morgan Kaufmann Publishers, Los Altos, CA.
8. W3 Recommendation, "Extensible Markup Language (XML)",http://www.w3.org/XML/, 1999.
9. E. Bonfigli M.E., Casadei G., Salomoni P., ``Adaptive Intelligent Hypermedia in Engineering Education", Proc. of 2000 ICSEE/Western MultiConference on Computer Simulation (ICSEE/WMC'2000), accepted for publication, San Diego (USA), January 2000.

BIOGRAPHIES

Maria Elena Bonfigli is Ph.D. student in "History and Computing" at University of Bologna, Italy. Her research interests include: Distributed Multimedia Systems, teaching/learning Environments, 3D Web Interface Design and Virtual Reality applied to Cultural Heritage.

Giorgio Casadei is full Professor of Information processing Systems and is currently the Department Chairman of the Computer Science Department, University of Bologna – Italy. He is also Member of the Scientific Council of Interdepartmental Center for Educational Research – Bologna University. Research area is the integration of information technology in education.

Paola Salomoni is currently an Assistant Professor of Computer Science at the Department of Computer Science of the University of Bologna. Her research interests include: Distributed Multimedia Systems, teaching/learning Environments and Integration of services in Computer Networks and Systems.