

Collaborative Courseware Authoring and Publishing Based on WebDAV, XML, and XSLT

Changtao Qu
Knowledge Based Systems
Inst. of Computer Engineering
University of Hannover
Appelstr. 4, 30167, Hannover,
Germany
qu@kbs.uni-hannover.de

Johann Gamper
Faculty of Computer Science
Free University of Bozen
Mustergasse 4, 39100, Bozen
Italy
Johann.gamper@unibz.it

Wolfgang Nejdl
Knowledge Based Systems
Inst. of Computer Engineering
University of Hannover
Appelstr. 4, 30167, Hannover,
Germany
nejdl@kbs.uni-hannover.de

Abstract

Teaching and learning is naturally an interactive, recursive, and iterative process. Consequently, the courseware is almost always changing during the whole teaching and learning process, which makes courseware authoring and publishing more complicated. In addition, with today's use of Web-enabled global cooperation in education, courseware authoring and publishing is further becoming a collaborative process due to the necessary cooperation among geographically dispersed partner universities. In this paper, we present a courseware authoring and publishing system which has been practically applied in a joint course held both at the University of Hannover and the Free University of Bozen. We adopt a recent collaboration-friendly Internet protocol, WebDAV, to support collaborative courseware authoring, the markup-language XML to represent meta-data of course contents, and the stylesheet language XSLT to accomplish courseware presentation. With its simple syntax, XML can on the one hand simplify the courseware authoring and structuring process, on the other hand, as a neutral meta-language, it can also separate course contents from courseware presentation when used with XSLT.

1. Motivation

The Web is fundamentally changing the way education is done. Today, not only in so-called virtual universities but also in many traditional universities more and more courses are partially or even entirely provided directly on and through the Web. As a matter of fact, Web-enabled global cooperation in education as well as Web-enabled new pedagogical methods have provided us with more possibilities to reduce the cost of education and to improve the efficiency and quality of the teaching and learning process, although at the same time they provide also more challenges on Web-based courseware design.

Since the summer semester 1999, the CS1 course "Introduction to Java Programming" (Info-1 for short) has been given directly on the Web at three German universities taking advantage of software and hardware facilities constructed in the 3-year-project "Virtual Campus Hannover- Hildesheim - Osnabrueck" [3][4][9]. Beginning with the winter semester 2000/2001, Info-1 is further given at the Free University of Bozen, Italy. At the same time, a project-based teaching and learning

method is adopted. In this paper, we will describe specifically the joint course Hannover/Bozen and the latest version of our courseware authoring and distribution tool, which has been developed within the Virtual Campus Project and the cooperation with Bozen University.

Put simply, "project-based" teaching and learning in Info-1 includes two important parts. On the one hand, the course material is no longer structured sequentially according to some commonly used Java books, rather it is introduced with a comprehensive example project which implements the game "BINGO!". As the course proceeds, the "BINGO!" project is elaborated step by step and new Java constructs, which are required for the example, are introduced. The "BINGO!" project is based on the source code for this game in the SUN Java Online-Tutorial, but the original SUN Java Tutorial is only presented as a completed project example, which is not suitable for a beginners course to accompany the lectures. On the other hand, the students are required to implement a similar project over the whole semester with the help of mentors and 24-hour-available on-line Info-1 courseware. Actually, Info-1 courseware plays an important role in the project-based teaching and learning process. It includes about 20 course units and each course unit consists of corresponding Java concepts, appropriate learning material about elementary computer science concepts, further references, and the source code of the Bingo units which can be directly connected to the students' own projects and which are used as best practice examples for the student projects. In the winter semester 2000/2001, we used 6-channel-ISDN video conferencing between Hannover and Bozen to give synchronous lectures from Hannover to Bozen. Students from Bozen and Hannover also worked collaboratively on the same projects.

In general, we have to face three main challenges in the design of project-based Info-1 courseware, especially when the design has to be done in a collaborative manner among four partner universities.

First, we have to find an efficient mechanism in order to support collaborative courseware authoring among geographically dispersed authors.

The authoring of the Info-1 courseware was performed by several lecturers in Hannover and Bozen. In fact, the courseware structure of Info-1 needs almost always to be adjusted and revised over the whole semester. On the one hand, we continually have to

integrate into Info-1 the latest teaching materials in order to reflect the technology development. On the other hand, according to students' feedback, the lecturers frequently have to adjust the courseware structure and content, e.g. moving a Java concept to a latter course unit because it turned out to be more difficult than originally imagined, or introducing more detailed concepts in a course unit to help the students to achieve a better understanding.

During the courseware authoring process, we frequently need to edit course script files, manipulate the namespace of the courseware repository (copy, move, or rename course script files), and exchange ideas and opinions on course scripts. Moreover, all these authoring activities must occur in a collaborative manner and need an efficient mechanism to secure the authoring process, e.g., preventing "overwriting" on each other's work. According to our experience, it is difficult and cumbersome to accomplish the above mentioned courseware authoring activities with the use of existing tools designed to support collaborative work among geographically dispersed team members (e.g. FTP and e-mail notification), especially when these activities occur permanently during the whole semester. On the other hand, usual version management tools are usable, but require additional client programs and specific commands to work, which again distracts the lecturers from the basic task, i.e. authoring courseware. By finally adopting a collaboration-friendly Internet protocol, WebDAV (Web-based Distributed Authoring and Versioning)[2], we have found a way to facilitate the courseware authoring process.

Second, we have to find an efficient mechanism to represent the courseware structure (expressed as meta-data of course contents), and based on it, to further separate course contents from courseware presentation.

Manipulation on the meta-data of course contents is actually a "must" for presenting the courseware on the Web. With regard to Info-1, we have also to provide different presentation versions based on the same course contents since the partner universities generally have a different teaching schedule and different teaching purposes. By adopting XML (eXtensible Markup Language) as the standard meta-language to represent the courseware structure, we have found a good way to separate course contents from courseware presentation.

Finally, we have to design a courseware publishing engine which can automate the courseware publishing process and reflect the changes of courseware structure at any time.

As we've mentioned above, the courseware is almost always changing during the teaching and learning process. Consequently, a courseware publishing engine should be able to immediately reflect the changes in the courseware structure, and accordingly to update the courseware presentation on the Web at any time. Moreover, the publishing engine should be independent of the courseware authoring process in order to hide implementation details from courseware authors, who

usually have no or little knowledge in Web design. By adopting XSLT (eXtensible Stylesheet Language Transformations)[1] together with JSP (JavaServer Pages) as the core of the publishing engine, the courseware publishing process has been to a great extent automated. The lecturers can adjust, revise, and then re-publish courseware at any time by themselves.

In the following we will describe the implementation details of our courseware authoring and publishing system which addresses the challenges discussed above. The system exploits the advantages of WebDAV, XML, XSLT, and JSP, as well as showing their application prospects in the area of Web-based courseware design.

2. Collaborative Courseware Authoring Based on WebDAV and XML

In general, the courseware authoring and publishing system consists of a courseware authoring module and a courseware publishing engine. They are constructed on a WebDAV&JSP-enabled Web server. Figure 1 shows the infrastructure of the courseware authoring module.

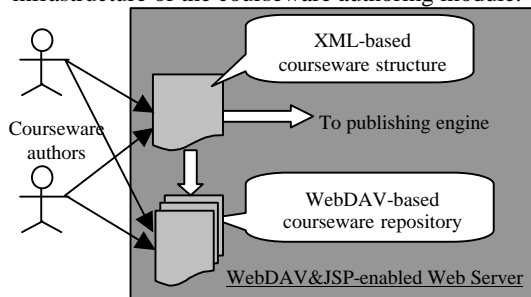


Fig. 1 Courseware authoring module

The courseware authoring module comprises a courseware repository which is used to store script files, and an XML file which is used to represent the courseware structure. The latter serves also as the standard data interface between the courseware authoring module and the publishing engine in order to cleanly separate course contents from courseware presentation. The authors can access the courseware repository via WebDAV protocol using WebDAV-enabled authoring tools, e.g., Microsoft FrontPage 2000, Office 2000, Internet Explorer 5.0 (Web Folder), Adobe GoLive 5.0, PhotoShop 6.0, Micromedia Dreamweaver 4.0, etc. Similarly, the XML file which stores the courseware structure can also be accessed via WebDAV using the above mentioned authoring tools. Another possibility to modify the courseware structure is downloading the file via HTTP or FTP for local editing with a visual XML editor, e.g., IBM Xena 1.2, Altova XML Spy 3.5, etc. (currently no WebDAV-enabled visual XML editor exists).

2.1. WebDAV-based Collaborative Courseware Authoring

The WebDAV protocol was worked out by the IETF (Internet Engineering Task Force) WebDAV working

group. It was partly finalized in the form of IETF RFC (Request for Comments) in February 1999 [2]. Designed originally to overcome some shortcomings of HTTP in supporting Web-based collaboration, WebDAV extends HTTP/1.1 and provides a coherent set of new methods, headers, and XML-based request and response entity body formats to directly support collaborative authoring on the Web.

In short, WebDAV provides four basic capabilities [2][5]:

Properties: The ability to create, remove, and query meta-information about documents, e.g., authors of document, comments on the document, etc.

Collections: The ability to create sets of related documents and to retrieve a hierarchical listing of their members.

Locking: The ability to control access to resources to avoid “lost updates” in a distributed, multi-user-authoring environment.

Namespace Manipulation: The ability to copy, move, or delete Web resources and collections of resources.

Taking advantage of WebDAV, the authors can “in-place” (directly on the remote server) accomplish most of the tasks required for collaborative courseware authoring, e.g., editing script files in the courseware repository, manipulating repository’s namespace, utilizing locking mechanism to prevent “overwriting”, or manipulating properties of course script files in order to exchange ideas and opinions among lecturers [5].

A typical authoring scenario in Info-1 is the following: a lecturer in Bozen wishes to edit course scripts. To accomplish this task, the lecturer chooses an authoring tool (e.g. Microsoft Web Folder) and logs into the WebDAV-enabled server using the password mechanism provided by the server. He is then presented with the directory structure of the courseware repository. Now the lecturer can view and edit course script files, manipulate the repository’s namespace, view the change history of a script file, or add comments to a specific script file utilizing properties manipulation. In addition, WebDAV’s locking mechanism ensures that only one lecturer at a time is able to edit a particular file, which can prevent “overwriting” on each other’s work.

Since all WebDAV-enabled authoring tools are aware of WebDAV’s methods (e.g., PROPFIND, PROPPATCH, LOCK, UNLOCK, etc.), they can “in-place” handle all “format-compliant” course script files without the need of explicit download and upload. Also the coordination work among lecturers is greatly simplified by WebDAV’s capability of property manipulation, which can to a great extent replace frequently used e-mail notification.

2.2. XML-based Courseware Structure Representation

We choose XML to represent the courseware structure mainly for two reasons. First, as a neutral meta-

language, XML is able to separate course contents from courseware presentation so as to provide “multi-views” based on the same course contents. Second, with its simple syntax, XML is very comprehensible and easy to use. Moreover, editing XML-files is greatly simplified by available visual XML editors, which can also validate XML files based on their DTD (Document Type Definition). In fact, the validity of courseware structure files is crucial to make the publishing engine work smoothly.

Figure 2 shows the DTD which specifies the courseware structure.

```

<!ELEMENT Courseware (Title, Author+,Description?, CourseUnit+)>
<!ATTLIST Courseware
xmlns:courseware CDATA #FIXED "http://www.kbs.uni-hannover.de/Courseware" >

<!ELEMENT Title (#PCDATA)>
<!ATTLIST Title
pic CDATA #IMPLIED >

<!ELEMENT Author (#PCDATA)>
<!ELEMENT Description (#PCDATA)>

<!ELEMENT CourseUnit (Overall,Location?, CourseElement*+)>
<!ATTLIST CourseUnit
name CDATA #REQUIRED
url CDATA #REQUIRED>

<!ELEMENT Overall (#PCDATA)>
<!ELEMENT Location EMPTY>
<!ATTLIST Location
uni (all|Hannover|Dresden|Hildesheim|Bozen) #IMPLIED>

<!ELEMENT CourseElement (#PCDATA)>
<!ATTLIST CourseElement
name CDATA #REQUIRED
url CDATA #REQUIRED >

```

Fig. 2. DTD of courseware structure

A DTD describes the general structure of courseware. Any courseware validated by the DTD can be directly rendered by the publishing engine without the need of any re-configuration.

3. Courseware Publishing Based on XSLT and JSP

In figure 3 we illustrate the infrastructure of the courseware publishing engine.

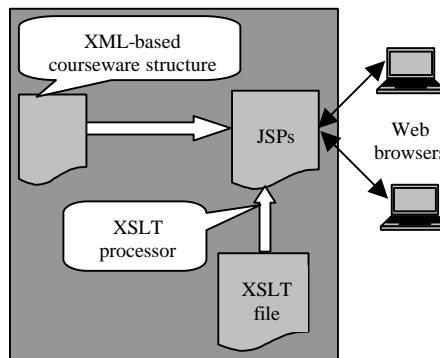


Fig. 3. Infrastructure of courseware publishing engine

The kernel of the courseware publishing engine is XSLT and JSP. XSLT is used to encapsulate presentation templates and to generate “multi-views” (different views) of courseware based on the same course contents. It can also render the courseware structure in various ways so as to dynamically present courseware, e.g., present only a part of the course units, or present specific contents of a course unit, etc. The courseware presentation is controlled by JSP, which is responsible for calling an XSLT processor to handle XSLT files and to generate the courseware presentation on the fly. In fact, designed originally as a presentation layer technology, JSP can be directly used to generate courseware presentation utilizing Java XML data-binding [6] and embedded HTML codes. We adopt XSLT to encapsulate presentation templates and use JSP only for controlling presentation logic, which can reduce the necessity to embed large amounts of in-line Java code (to control presentation logic) and HTML code (to achieve presentation) in JSPs. In terms of the Model-View-Controller (MVC) patterns, the XML file acts as the model, the JSP serves as the controller, and the XSLT file is the view. By changing only the XSLT file, different views can be easily realized. Actually, such sort of design can make the courseware publishing engine more robust, reusable, and maintainable.

Because JSP is 100% Java compliant, the generated courseware presentation can be directly accessed across various operating systems and Web browsers.

4. System Implementation

Figure 4 shows the server architecture of the courseware authoring and publishing system.

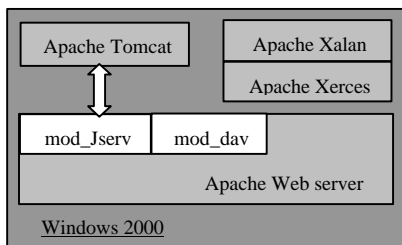


Fig. 4. Server architecture of the courseware authoring and publishing system

The whole system is built on Windows 2000. The standard Web server is Apache Web server 1.3.14. It is extended with several additional modules for WebDAV and JSP support. One module is mod_dav 1.0.2 [8], which is written in C and provides the four basic capabilities of WebDAV for the Apache Web server. The JSP support is provided by Apache Tomcat 3.2.1, which is installed as an add-on to the Apache Web server. Another module, mod_Jserv 1.1 [7], is responsible for transferring all JSP requests on Apache Web server to Tomcat. In addition, the Apache XML

parser Xerces 1.2.3 for Java is used to handle XML syntax, and the Apache XSLT processor Xalan 1.2.2 for Java is used to render courseware structure.

5. Conclusion

Our courseware authoring and publishing system is characterized by a WebDAV-based courseware authoring module which supports collaborative courseware authoring among geographically dispersed lecturers, and an XSLT&JSP-based publishing engine which is independent of the authoring module facilitated by the standard data interface XML. The clean separation of course contents from courseware presentation ensures the flexibility, reusability, and maintainability of the system.

6. References

- [1] Clark, J., “XSL Transformations (XSLT) Version 1.0”, Available at: <http://www.w3.org/TR/xslt>, Nov. 1999.
- [2] Goland, Y. Y., E. J. Whitehead, A. Faizi, S. Carter, and D. Jensen, “HTTP Extensions for Distributed Authoring-WEBDAV”, RFC 2518, Feb. 1999.
- [3] Henze, N., and W. Nejdl, “Extendible Adaptive Hypermedia Courseware: Integrating Different Courses and Web Material”, In *Proc. of International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, Trient, Italy, Aug. 2000.
- [4] Henze, N., W. Nejdl, and M. Wolpers, “Modeling Constructivist Teaching Functionality and Structure in the KBS Hyperbook System”, In *Proc. of Computer Supported Collaborative Learning Conference*, Stanford, USA, Dec. 1999.
- [5] Qu, Ch., T. Engel, and Ch. Meinel, “Implementation of a WebDAV-based Collaborative Distance Learning Environment”, in *Proc. of ACM SIGUCCS (Special Interest Group on University and College Computing Services) Fall 2000 User Services Conference*, Virginia, USA, Oct. 2000.
- [6] Reinhold, M., “An XML Data-Binding facility for the Java Platform”, Available at: <http://java.sun.com/xml/docs/bind.pdf>
- [7] Rochat, J. L., “Load-Balancing-Fault tolerance with Apache JServ 1.1”, Available at: <http://java.apache.org/jserv/howto.load-balancing.html>
- [8] Stein, G., “mod_dav: a DAV module for Apache”, Available at: http://www.webdav.org/mod_dav/
- [9] Wagner, E., “Creating a Virtual University in a traditional environment”, In *Proc. of the 7th European Distance Education Network*, Bologna, Italy, June 1998.