A Modular Approach to E-Learning Content Creation and Maintenance

Phan Thanh Duc[1], Peter Haddawy[2]

[1]Mathematics and Informatics Department, Academy of Banking, Hanoi, Vietnam
ducpt@email.com

[2] CSIM, School of Advanced Technologies, Asian Institute of Technology, Thailand
haddawy@ait.ac.th

Abstract. Preparing multi-media rich e-learning content is a labour-intensive process, requiring great time investment from content experts and multi-media designers. This high labour cost is particularly acute in fields in which knowledge changes rapidly. Keeping material current requires periodic review and updating of the material. Thus there is a need for tools to facilitate the updating of asynchronous e-learning material. While much software exists for content creation, content maintenance has received relatively little attention. We present an Learning Content Management System (LCMS) that addresses two aspects of this problem: updating course structure and updating course content. Our approach organizes course material into modular units and externally specifies course structure. We introduce the notion of Updatable Content Unit (UCU). The content author can define such units at content creation time or any time there after. Our system provides functions to search for UCUs, edit them, and integrate them back into the surrounding material.

Keywords: System Architecture, Courseware Building Tool, Management of Learning Resources.

1. Introduction

Of the two basic modes for delivering course material online, synchronous and asynchronous, the latter has gained more popularity due to a number of factors. Among these are: the relatively fixed cost, independent of the number of students; the ability of students to tailor the learning pace and time schedule to their needs; and the ability to use the material for either distance education or to supplement traditional lectures. But this flexibility comes at a price. Preparing multi-media rich learning content is a labour-intensive process, requiring great time investment from content experts and multi-media designers. This high labour cost is particularly acute in fields in which knowledge is changing rapidly. Keeping material current requires periodic review and updating of the material. In the case of rapidly changing areas like e-commerce, this may need to be done on a semester basis. Thus there is a need for tools to facilitate the updating of asynchronous e-learning material. While much software exists for content creation, content maintenance has received relatively little attention. Although content creation tools can be used for content maintenance, the problems are fundamentally different and maintenance requires its own specific set of tools.

There are two aspects of the content updating process. The first relates to the content structure of the course. Let's take an Electronic-Commerce (EC) course for example. Normally, a course includes some Learning Content Objects (LCOs) – (e.g. course, chapter, lesson). Each of them has a set of resources (assets, physical files) and the sequence among them. Suppose now that some standards for electronic payment are changed, so that the old material must be replaced with updated material. The process of adding the new material is not a very complex task and can be solved with File Transfer Protocol (FTP) applications. However, the major problem is course navigation. Navigation refers to both intra- and inter-LCO navigation. Since objects typically link to other objects, changing from one LCO to another may affect other LCOs. Furthermore, if we introduce a new chapter or lesson, this material must be integrated into the existing navigation scheme.

The second aspect of maintenance involves identifying material in need of updating and then making the changes. For example, in the case of e-commerce, information on things like online sales, the number of online businesses in a given sector, the performance of various high-profile online businesses would ideally be updated every few months. We require support to identify this content and then to make changes to it in such a way that it still fits within the scope of the surrounding material. This problem also arises with localization of material. In web-based training it is not unusual to offer the same course to students in different countries. The teaching material can be more effective if the content is localized to address the particular environment of that country.

In this paper, we describe an implemented system for creating modular e-learning materials that can be easily updated and for supporting the updating process. The technique addresses both aspects of updating discussed above. The problem of updating the navigational structure is solved by explicitly representing the hierarchical and sequential organization of the material and by packaging it as learning content objects. The problem of updating of content is solved by using the concept of "data islands", which are small sections of material that may need to be updated in the future, which the content author can identify during or after the authoring process. These data islands are stored in a database in order to facilitate searching, adding, deleting, editing. Our solution makes extensive use of the Sharable Content Object Reference Model (SCORM) in order to facilitate exchange of material and interoperability with other e-learning software.

2. Related Work

Systems supporting creation and maintenance of e-learning content are known as Learning Content Management Systems (LCMS). Available LCMSs provide functions that allow e-Learning organizations to rapidly author, deploy and manage e-Learning content [3,4,8,9,10]. Although features of LCMSs differ from one system to another, the fundamental components of an LCMS consist of an authoring tool, an administrative application, and a learning object repository or central database. Although it is recognized that one role of an LCMS is to support maintenance of course content [2], little has been done in this area. WebCT [12] has good tools for managing files to update the course content and structure, however, the issue of indexing and updating frequently changing content in a course is not addressed.

Recently, the idea of creating e-Learning courses by combining reusable e-Learning objects has attracted much attention. The SCORM Content Aggregation Model [1] "represents a pedagogically neutral means for designers and implementers of instruction to aggregate learning resources for the purpose of delivering a desired learning experience." But while this standard can be applied to create a modular multi-media e-learning material that facilitates updating, it does not directly address the problem of updating e-learning materials.

3. Modular E-learning Materials

3.1 Learning Content Model

The Learning Content Model is described as components used to build an LCO from learning resources that can be easily replaced. Also, it defines how these lower-level learning resources are aggregated to compose higher-level units of instruction.

- *Assets:* Assets are electronic representations of learning content.
- *Learning Content Objects (LCO):* This is a collection of one or more assets and is the lowest level of learning resource that will be used by the LCMS.
- *Content Aggregation:* This is a content structure that can be used to aggregate learning resources into a cohesive unit of instruction (e.g. course, chapter, lesson, etc.).

In the past, the tool to control the sequence of learning resource typically embedded, inside proprietary data formats, all of the navigation information that governs which parts of the course the learner will view next. In most cases, authoring tools or systems are defined and applied to designated courses and sometimes used only for a unique course. Using this mechanism, it is difficult to update content.

In our approach, learning resource sequencing is defined in the content structure and is external the learning resource. During run-time, the LCMS is responsible for launching the learning resources in a defined sequence. So the task of content creators is separated into two different parts: developing the learning content objects and specifying their sequence.

3.2 Meta-data

The Meta-data provides a common nomenclature enabling learning resources to be described in a common way. Meta-data can be collected in catalogs or directly packaged with the learning resource. Learning resources described with meta-data can be systematically searched for and retrieved. Meta-data can be classified by their positions within the course manifest file, and according to whether they provide context specific or context independent descriptions.

Context specific meta-data are data specific to a particular course packaged by a manifest file. During the process of creating and updating course structure, context specific meta-data about each LCO such as chapters, lessons are also generated in order to support the process of specifying learning resource sequencing.

Context independent meta-data are data describing individual LCOs or Assets that are independent of a particular course that uses these objects. This information can be used to search the LCOs or Asset for replacement or updating. These might include: LCO or Asset title, general description of an LCO or Asset, the original author of an LCO or asset, keywords associated with the LCO or asset, copyright of the LCO or asset, etc.

3.3 Content Packaging

Figure 1 shows the structure of the Content Packaging. The Content Packaging defines the structure and the intended behaviour of a collection of learning resources and also provides a standardized way to update digital learning resources by using the LCO update tool.

A Content Package contains two major components:

- A special document describing the content organization and resources of the package. This document uses eXtensible Markup Language (XML) to describe the content and is called the Manifest file (manifest.xml). This document is automatically generated based on context specific meta-data about LCOs in the course.
- The physical files (assets) referenced in the Manifest file.

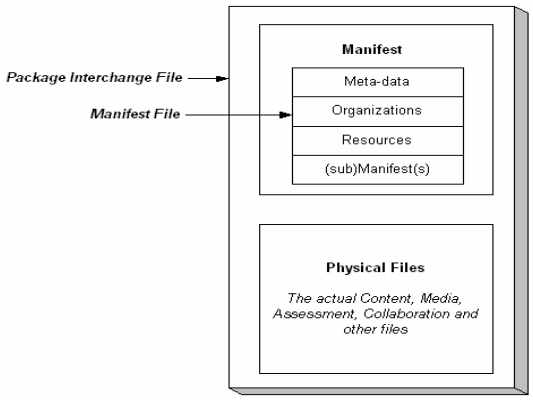


Fig. 1. Content Packaging Conceptual Diagram (Source: SCORM)

3.3.1 Package

A package represents a unit of replaceable content. This may be a part of a course that can be delivered independently as an entire course or as a collection of courses. A package is able to stand-alone; it contains all the information needed to use the packaged contents for learning when it has been unpacked.

3.3.2 Manifest

A Manifest is a description in XML of the resources comprising meaningful instruction. A Manifest can describe a part of a course that can stand by itself outside of a course, an entire course or a collection of courses. So the content developers have

to describe their content in the way they want it to be considered for aggregation or disaggregation.

A package always contains a single top-level Manifest that may contain one or more sub-manifests. The top-level Manifest always describes that package and any nested sub-manifests describe the content at the level in which the sub-manifest is scoped, such as a chapter, a learning object, etc.

3.3.3 The physical files

The physical files represent the actual files referenced in the resource components. These files may be local files that are actually contained within the content package, or they can be external files that are referenced by a Universal Resource Indicator (URI).

4. Solution

4.1. Updating course structure

To manage the whole content of the course, the course structure must be described in a meta-data file – Manifest.xml. Any changes of the course structure shall lead to changes of this file. In order to do that, the LCMS needs to have abilities to generate and update the Manifest file.

The course creator will describe (for a new course) or edit (for an existing course) course structure in a web application. First of all, he/she logs into the LCMS, then he/she will be able to create a new course or update existing courses within his/her authority.

Each course is described by a set of static and dynamic information. The static information describes all the stable data about the course such as course name, course identifier, etc. This data will be kept in static variables based on SCORM standard to ensure the compatibility of the system to the external resources. The dynamic information describes dynamic items of the course such as information about resources of each LCO: chapter, lesson, etc. The course structure is represented according to the following scheme: 1 course – n chapters, 1 chapter – m lessons, 1 lesson – k assets.

This information is not static; it may be changed time after time and we cannot determine the number of variables at the beginning. Therefore, a technique is required to manage the set of static information and dynamic information in the LCMS. The technique which is used in our system is called “dynamic code”. It automatically generates blocks of code based on the description of the course creator by using server-side code. However, for efficiency, the combination of the client-side and server-side should be used. As soon as getting all necessary values of the dynamic data, the system will transfer those values to the server to process. This mechanism is also used to easily upload multiple files (assets) from the course creator’s computer (client) to the server. The technique to transfer multiple files to self-determine concrete location on the server makes it easier for the course creator to upload files. To avoid the problem of line transferring between client and server when transmitting many files, the process of uploading will be performed with each LCO. Each LCO is

uploaded to a suitable directory on the server. Accordingly, all asset files of the LCO are uploaded at the same time.

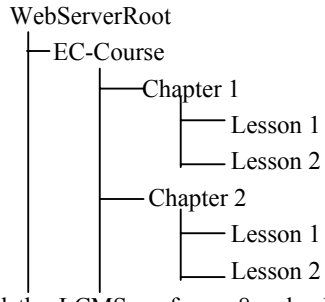
Example:

An E-Commerce course comprises 2 chapters.

Chapter 1 has 3 lessons : Lesson 1 has 5 assets (files), Lesson 2 has 7 assets

Chapter 2 has 2 lessons : Lesson 1 has 5 assets, Lesson 2 has 3 assets

The Multi-Files-Upload function of the LCMS will create a directory tree as follows:



And the LCMS performs 8 upload sessions to upload the declared assets for each directory. Figure 2 shows the online course declaration screen of the LCMS.

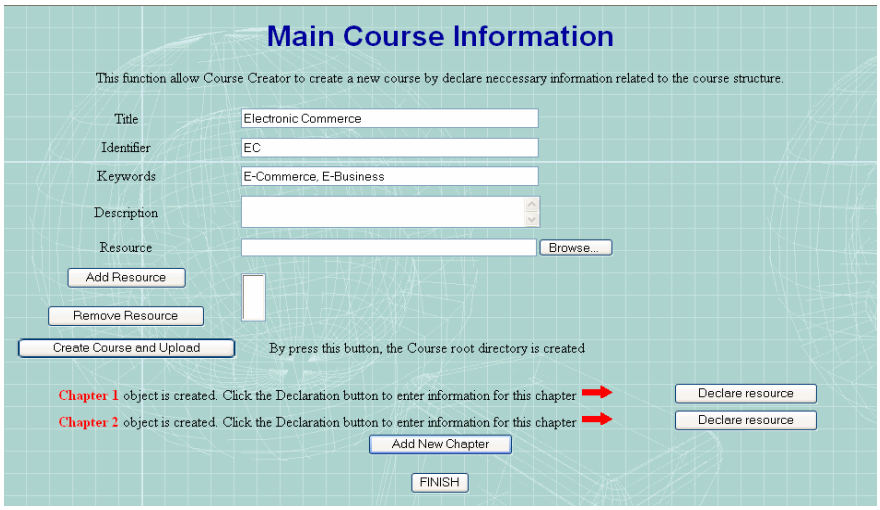


Fig. 2. Course Declaration Screen

After uploading all assets of the LCO in the course, the system will automatically generate a Manifest file. To perform this function, the crucial problem is how to manage all the static and dynamic variables already established during the course structure declaration process. As these variables exist during the time the course creator works with the system, they should be declared as application or session variables.

A Dynamic Menu Tree program is used to read meta-data file (Manifest file) and then generate a tree menu for the course. Each node in the tree menu represents an LCO. Learners can click to each node to move to every part of the course. To move within

an LCO, the sequence of the assets in that LCO needs to be declared in advance by the course creator and a module of client-side code will read this information and move within that LCU.

The process of updating contents related to the course structure is solved by reading the Manifest file and allowing the course creator to edit necessary information.

4.2. Updating course content

The whole content of the WBT course includes multimedia materials which are embedded in web pages. From web environment point of view, these contents are static information and very difficult to edit. In fact, the frequently changing contents in a WBT course are usually text paragraphs. These paragraphs store contents which possibly change in a short time. In order to manage this kind of information, we propose the concept of Updatable Content Unit (UCU). UCU is a piece of content that is stored in a database. Each UCU can be treated as a record in the database; they are embedded in static web pages and can be converted to static data and vice versa. It has some specific fields to store necessary information to find frequently changing contents in the whole course. For example, an index item to identify the UCU and another item to locate the position (chapter, lesson, etc.) of the UCU in the course. This information will be used to find the UCU in the content update process.

The novelty here is the ability to convert from text to UCU and vice versa. As a result, the LCMS must have necessary functions to convert a static data section to a UCU, to update a UCU, delete or convert a UCU to static data.

We can approach this data from two points of view. From the learner's stance, the appearance of the static data and dynamic data is quite the same. Learners can not distinguish dynamic content from static content. From the course content creator's view, dynamic content can be treated in the following two ways.

First, the course content creator views all course contents in a similar way as the learner by using Dynamic Tree Menu. However, the dynamic contents are represented differently from static contents. They may appear in text boxes with specific colour, etc. The course content creator may edit these contents by selecting sections that need to be edited, calling the edit function to update new content, and saving the modified data.

To convert static content to dynamic content, the course creator selects a text paragraph and calls the convert function to convert the selected paragraph to a UCU. In fact, this function will create a new UCU in the UCU database, copy the selected content to the above UCU, create a connection to the web page and insert the new UCU to current location of the selected paragraph. In practice, each UCU has an index and this index number is used to determine which UCU to locate and in which position of the web page.

In the second way, the content creator can view all existing UCUs in the database and edit necessary UCUs. In this way, the content creator can easily manage all UCUs but can not add or remove any UCU.

5. Conclusion

The issue of maintaining asynchronous e-learning material, while important from the standpoint of cost over the lifetime of the material, has not previously been sufficiently addressed. We have presented an LCMS that addresses two aspects of this problem: updating course structure and updating course content. Our approach builds upon the SCORM standard by organizing course material into modular units and externally specifying course structure. We introduce the notion of Updatable Content Unit. The content author can define such units at content creation time or any time there after. Our system provides functions to search for UCUs, edit them, and integrate them back into the surrounding material.

References

1. Sharable Content Object Reference Model (SCORM) Version 1.2, 2002. (www.adlnet.org/index.cfm?fuseaction=scormabt)
2. Johan Ismail. The design of an e-learning system: Beyond the hype. *The Internet and Higher Education*, Volume 4, Issues 3-4, 2001, Pages 329-336.
3. Click2Learn (http://home.click2learn.com)
4. e-Learning Consulting Co. Ltd (http://www.e-learningconsulting.com)
5. Arnd Steinmetz, Martin Kienzle. "The e-Seminar Lecture Recording and Distribution System". Proceedings of SPIE Vol. 4312 (MultiMedia Computing and Networking 2001
6. Herng-Yow Chen, Gin-Yi Chen, Jen-Shin Hong, *"Design of a web-Based Synchronized Multimedia Lecture system for Distance Education,"* IEEE International Conference on Multimedia Computing and Systems Volume II-Volume 2. 1999
7. Christoph Meinel, Harald Sack, Volker Schillings November 2002. “Course management in the twinkle of an eye - LCMS: a professional course management system”. Proceedings of the 30th annual ACM SIGUCCS conference on User services.
8. Ryann K. Ellis “LCMS Roundup” *Learning Circuits*. August 2001. (http://www.learningcircuits.org/2001/aug2001/ttools.html)
9. Jill Funderburg Donello. “Theory & Practice - Learning Content Management Systems” Leadingway Corporation (http://www.leadingway.com)
10. “Making the case for content” - Institute of IT Training and the Training Foundation. (http://www.elearningprofessional.com/index.asp)
11. “The Evolution of the Learning Content Management System”. ASTD’s Online Magazine about e-Learning. April 2002. (http:// www.learningcircuits.org/2002/apr2002/robbins.html)
12. WebCT (http://www.webct.com)