# An Online Template-Based Authoring System for E-Learning

Simon Hui and James Liu

Industrial Centre, The Hong Kong Polytechnic University, Hong Kong, China
{iccshui, icdjliu}@inet.polyu.edu.hk

**Abstract.** This paper presents an online authoring system that allows users to create interactive course content and course structure on the Internet in a fast and easy way without any programming knowledge. The underlying CMS (Content Management System) provides the flexibility of XML-based application interface to publish content for Flash courseware or other online learning systems such as WebCT. An extensive library of Flash Templates is provided for course presentation and interactivity. With the templates and the CMS, this online authoring system allows minimum course development efforts from tutors and delivers interesting learning experience to students. In this paper, a primary school courseware is developed to demonstrate the effectiveness of the system.

**Keywords:** publishing tool for e-learning, online authoring system, template-based courseware, content management system, Zope application.

## 1 Introduction

For non-programmer teaching professionals, there is a need for rapid content authoring without the need to program and to use sophisticated commercial authoring tools. Standard Web-authoring tools [1] such as FrontPage, Dreamweaver, Photoshop and Flash, each covers some of all the aspects of course development such as course presentation, course navigation, multimedia and interactivity. When interactivity is concern, JavaScript or Action Script [2] programming will be necessary.

As a result, a course developer has to grasp deep understanding of suitable tools that are expensive in terms of purchase cost as well as training time, and the production of consistent courseware is complex and difficult to control.

Course interface design also causes headaches. Web authoring tools have limited methods to separate presentation and content. A tutor without design experience or with limited time can only achieve non-appealing visual effects and unfriendly user interface.

In an effort to free a tutor from the above problems, we have developed a one-stop authoring system for e-learning with a built-in Flash template library that includes many quiz features such as Drag and Drop, Multiple Choice, Hot Spots, etc. The authoring system supports content editing as well as navigation. Online access and

R.W.H. Lau et al. (Eds.): ICWL 2005, LNCS 3583, pp. 38 – 48, 2005.

multi-author support makes it possible for tutors to work online any where, any time, without having to install any software in their PCs. All these features supported in an integrated system are attractive for general non-technical users.

## 2 System Architecture

In this section, we will describe the architecture overview, account management, course content management, course navigation and template-based user interface and XML support. The description will help to outline the design of the online system.

### 2.1 Architecture Overview

The system design (see Fig. 1) is based on client/server architecture to support multi-author online access. When a client's request is sent to the server, the server will query on the OODB (Object Oriented Database) based on the request and process the query result. Once the process is finished, the result is sent back to the client in the form of XML response. The client will then parse the XML response into Flash recognizable data and render the data with a designated Flash template. OODB is chosen as the server database due to its good support in creation and management of hierarchical objects over RDB (Relational Database). An ODBMS

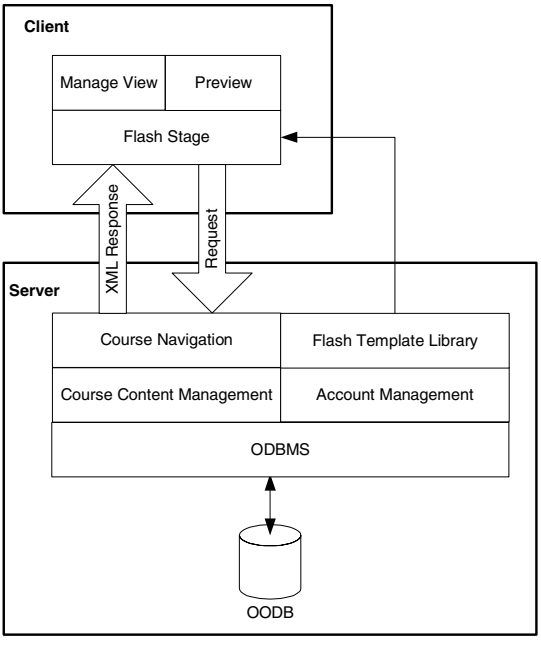


**Fig. 1.** Architecture Overview

(Object Database Management System) is then adopted to implement the CMS (Content Management System) on top of OODB. As the core of the online authoring system, the CMS supports a package of course management features such as course framework, navigation, account management and separation of presentation from content.

On the client side, a Flash Stage object (a swf file) is responsible for handling server response, parsing course content, and dressing the content with different Flash templates. On top of Flash Stage there are 2 types of user interfaces: Manage View and Preview.

### 2.2 Account Management

Account management provides 3 functions: User Registration, Authentication and Permission. When a user first logs in the system, he or she will be required to fill in the Registration form so that the system would be able to recognize him or her as a valid user according to provided information. Authentication is performed when a registered user log in, the system will validate his or her user information and authorize entry permission.

### 2.3 Course Content Management

Course content management deals with how the course content is developed and organized. While template-based interface brings simplicity in editing, it also brings limitations in structure. A two-level course structure can only fit in a two-level template. An alternative would be to have a n-level (for example n=5) course structure fixed for the system templates so that it can fit with m-level of course structure as long as m<=n (for example m=2, 3, 4, 5). Great effort will be needed to design templates in order to keep the style and navigation consistent in adjusting to different type of course structures and the number of templates will be limited as well.

### 2.4 Course Navigation

Two kinds of navigation are to be considered here: the courseware navigation for the students/readers and the management navigation for the course builder. Common look and feel courseware navigation is built in each template so that all course objects applied with a template can be navigated consistently. Linear navigation is adopted because users would not get ‘lost’ easily by only making simple decision when browsing such as next page or the previous.

Management navigation needs to provide a more flexible path because a course builder will need to perform a relatively complex operation in the management view. A course builder will usually need to quickly access back and forth the course objects located at different places for editing or referencing. Since the course content is organized in a tree structure, an expendable tree menu is provided for fast access to different levels of objects in a limited space.

### 2.5 Template-Based User Interface and XML Support

Template-based user interface provides simplicity, robustness, and flexibility. It also supports interactivity because all templates are Flash objects. A client-side component called Flash Stage is used for integrating the course content and the selected Flash template in client side before it is shown to the users. Course object (along with its attributes) is packaged in a form of XML Response message before it is sent to the client. XML is selected not only because Flash Stage component supports XML parsing, but also the course contents can be transformed to other system by way of XML. By parsing the XML Response message, the Flash Stage component transforms the course data into Flash values and gets the attached file name of the selected Flash template. The selected Flash template is then called from the server's Flash Template Library and sent to the Flash Stage for integration with the course values.

Flash Stage supports two kinds of Flash user interface depending on the applied Flash templates: Mange View for editing data and Preview for getting the real picture. Manage View is implemented in a WYSIWYG (What You See Is What You Got) manner. For example, multiple choices, fill-in blanks and selection boxes are different in layout and each has its own way for editing. After editing, the user can simply switch to the Preview interface for a peep of the final effect, where the static data becomes interactive and animated with fun. In Fig. 2 and Fig. 3, the Management View and the Preview View of multiple choices are shown.

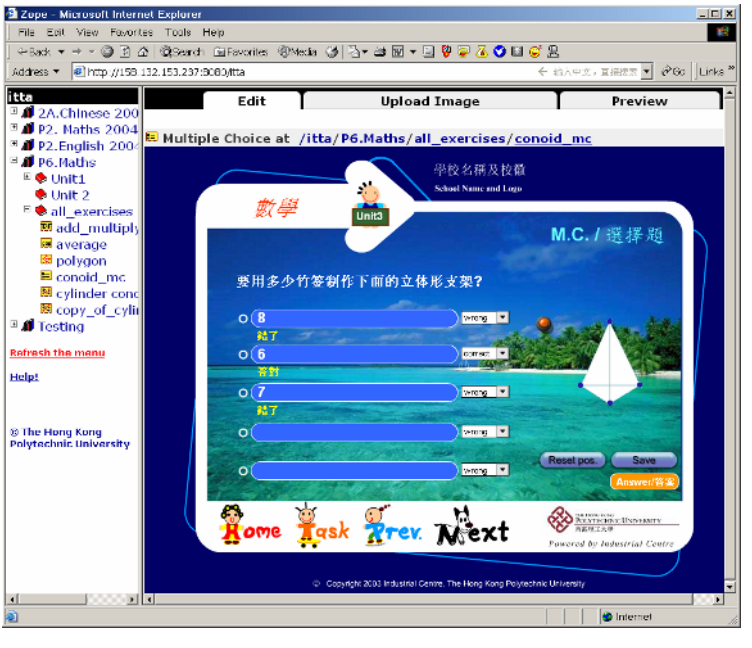


**Fig. 2.** Multiple Choices Management View

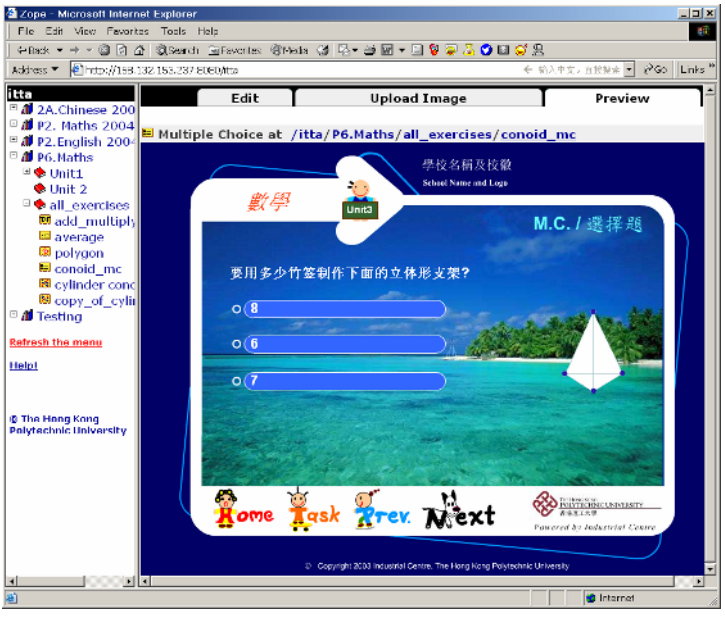


**Fig. 3.** Multiple Choices Preview View

## 3 Implementation Issues

In this paper, the following techniques are used: OODB design, extending Zope, Python, output xml, scalability of template, Flash Stage parsing and integration, navigation and embedded Flash interface. In order to implement a system based on the above techniques without too much cost, an open source application server, Zope is used as the underlying content management system. The following paragraphs will first introduce Zope 2.0, and then describe some challenges in extending Zope 2.0 to support course content management and Flash user interface. Some issues about exporting the course content will also be discussed.

### 3.1 Zope 2.0

Zope 2.0 [3] is an open source application server written in Python. Zope is a highly object-oriented Web development platform that provides clean separation of data, logic and presentation, an extensible set of built-in objects and a powerful integrated security model.

### 3.2 Account Management

The multi-author access to the authoring tool is easily implemented by the inheritance of a Zope account management object: ‘User Folder’, with the built-in methods that

support user registration, authentication and permission setting. Each User object (see Fig. 4) in the User Folder represents a course builder account and contains the authentication information.


**Fig. 4.** Account Management

### 3.3 Course Framework

Zope 2.0 comes with an OODB (Object Oriented Database) which allows users to represent data as objects with attributes and organize the relations of objects in a way similar to the real world. Some built-in objects are provided with basic functions such as Folder and Simple Item. These objects are defined in classes written in Python language and can be extended to build advanced course objects such as 'Subject', 'Unit', 'Topic' with special attributes and functions.

In developing the template-based authoring system, 12 classes are built by extending a built-in class 'Folder', among which 10 aggregations and 12 inheritance relations are defined and compose a course framework that seems to match most aspects of a typical course structure. Following is a list of these classes with description:

- TBAT: Template-based Authoring Tools.
- Subject: Top course object that contains Units.
- Unit: Smaller course object that contains Topics.
- Topic: Base class of smallest course object.
- CourseA: Course object (type A).
- CourseB: Course object (type B).
- CourseC: Course object (type C).
- MC: Assessment object (Multiple Choice).
- TrueFalse: Assessment object (True & False).
- Fill: Assessment object (Fill in the blank).
- Drag: Assessment object (Drag & Drop).
- Spot: Assessment object (Spot the Error).

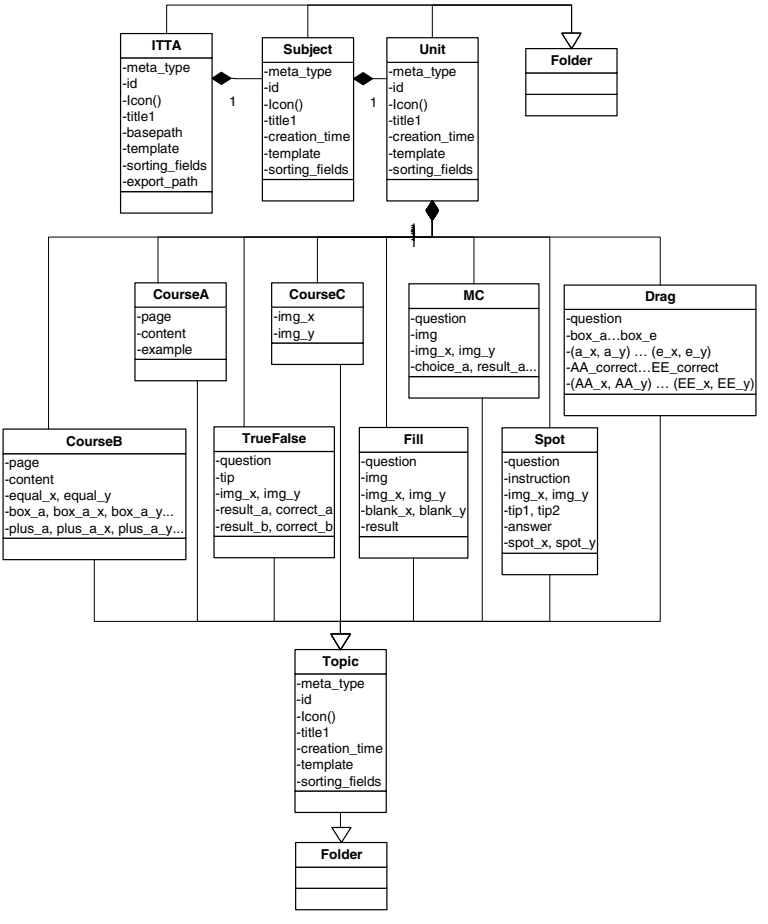


**Fig. 5.** Class Diagram

In Fig. 5, the UML Class Diagram [4] shows the inheritance and aggregation relations among the classes. For example, we can clearly see such an inheritance path:

**ITTA ← Subject ← Unit ← MC**

which means:

- An ITTA object contains and manages multiple Subject objects.
- A Subject object contains and manages multiple Unit objects.
- A Unit object contains and manages multiple MC objects.

We can also find such an aggregation path:

**Folder ← Topic ←MC**

which means:

- All MC objects inherit properties and methods from Topic.
- All Topic objects inherit properties and methods from Folder.
- All MC objects inherit properties and methods from Folder.

### 3.4 Course Content Management

The built-in content management interface of Zope 2.0 provides almost everything a developer need to build the course content management interface for the authoring system. A new feature is added to the course content management (Fig. 6) that allows the course builder to 'order' the objects. By 'ordering the objects', we mean that a course builder can change the presentation order of an object by *moving* the object.

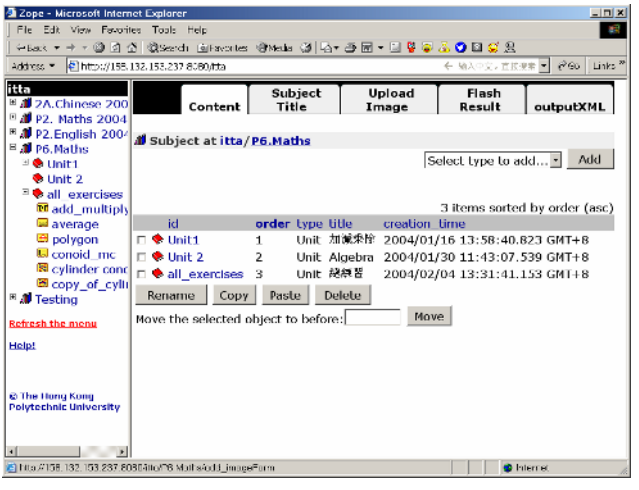


**Fig. 6.** Subject Management

### 3.5 Integrating Content and Style

By attaching a style page called ZPT page, Zope can dress its objects into any HTML style. To extend this ability and integrate course content with Flash Style – Flash Template, several issues should be considered:

- Where should the integration happen? Client side or server side?
- In which component?
- What is necessary information for the integration?
- How to integrate once we have this information?
- How to do it without changing much of the existed Zope publishing?
- Is it possible to make the integration transparent?
- How should Zope communicate with the Flash template?
- What information should Zope publish?
- When and how to assign a Flash template for the specific course object?

It is possible to make the integration transparent because the object-based structure of Zope allows an object to have its own methods in transforming the data. There is no limit in what the transformation result must be – it could be HTML, XML or Flash. The Zope object's properties also contain enough information necessary for the integration. For example here's a list of the properties and their description of a 'Drag & Drop' object:

–Preferred Flash template: a Flash file name.
–URL of previous, next, home page: for navigation.
–Current Subject: subject name.
–Current Unit: unit name.
–Drag & Drop question: sentence.
–ID of 5 box-items: items to be matched. Missing ID indicate a removed item.
–Correct destination of 5 drag-items: non null.
–Recorded position of each drag-item: remember where the user put them last time.
–Calculated position of each box-item: items are placed with equal interval according to the number.

Communication is expected between Zope and Flash in integrating content coming from Zope and style coming from Flash template. The choice of language is not much since only Python and ZPT are recognized by Zope and only Action Script is recognized by Flash. Fortunately both Flash and Zope support XML by providing XML parser and XML publisher, which make indirect transformation from Zope object to Flash Action Script values possible.

The Stage object – a Flash movie is the perfect environment for that kind of transformation not only because it can load course content by XML but also load course template in the form of a Flash sub-movie.

The Template Library – a Zope Folder object linked to server file system provides the storage of 8 templates. Web access of each template is supported by Zope object's URL. Synchronization between the Library and the server file system make it a simple matter for the course designer to edit each template without interfering the job of the course builders.

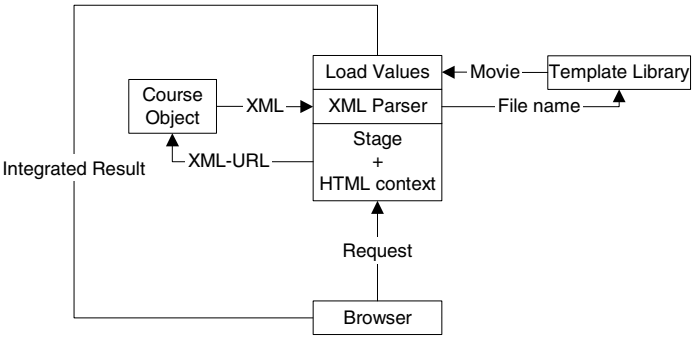


**Fig. 7.** Integration Process

As shown in Fig.7, after a user's request is directed to the Stage object, in order to fill the Stage with meaningful course content, the Stage object calls the XML publish method to get the XML representation of the object. It then uses a XML Parser to transform the result into Action Script values. From these values the preferred Flash template's file name is extracted and the movie is loaded from the template library into the Stage object. The final integration is done by loading the Action Script values into the Flash template movie.

### 3.6 Export Course to Local

Export function is provided in the template-based authoring tool similar to the save function of standard PC software. The exported files include the Stage object, the XML files that represent the course content with navigation information and the Flash template movies for style selection. Since the collection of files can work locally without server support, it can be published to a CD or another online platform such as WebCT.

### 3.7 Working with WebCT

WebCT is an online course management system with tools like course syllabus, multimedia, discussions, chat, assignments, quizzes and so on. Despite the various management and delivery functions it provides for the teaching professionals, WebCT has virtually no support for rich content and interaction. The course authors using WebCT have an option to learn to use advanced software such as Macromedia Flash and upload the resulted Flash Movie into WebCT. But this requires advanced programming experience and the resulted movies are usually isolated files without support for course structure and navigation.

On the other hand, the online template-based authoring system provides an easy way to build rich and interactive course content without advanced programming knowledge. The result of template-based authoring system can be easily exported and uploaded to WebCT. By integrating the template-based authoring system result and the WebCT, course authors are able to develop course contents with interactive templates and still enjoy the many benefits provided by the WebCT.

## 4 Primary School Courseware Development

In this paper, a primary school courseware is developed with the template-based authoring tool. The development took place in a primary school server installed with the authoring tool. A course builder then registers and logs in to his or her own 'course space' on the Internet, and starts to develop a rich media and interactive courseware (in office or home) in a few steps. The following is a fraction of operation sequence in developing the primary school courseware:

- Add a Subject 'chinese_terms' in Root folder.
- Enter Subject 'chinese_terms'and upload a background image for the Subject.
- Add 2 Units 'word' and 'sentence' in the Subject.

- Enter Unit 'word'.
- Add a 'Fill in the Blank' object: 'word1'.
- Enter 'word1' to input the answer and the correct answer, and position the input box in the Flash edit interface.
- Return back to Unit 'word'.
- Copy the 'word1' object and paste it in the same Unit and name the new object 'word2'.
- Enter 'word2' and change question, answer and their position.
- Use the tree menu to navigate to Subject level.
- Click Preview to see the final Flash result.
- Click Export to export the movie.

## 5 Conclusions

An online template-based authoring system is designed and developed to assist non-programmer course builders in rapid course development. Account and content management features from Zope 2.0 are extended and applied to the core of the system. A multi-authoring space and 3 classes of course object are built in supporting the 3-level course structure. A library of 8 Flash templates is produced for content integration and a linear or tree menu is provided in navigating the system.

The demonstration of the system in a primary school course development shows that a course builder can access the authoring system online any time, anywhere. The course builder can build a course easily and quickly in a few steps without much technical efforts. Because the system is based on Flash templates, a tutor is able to achieve professional-like visual design and interactivity. The tutor can also export the courseware in a CD or online.

## References

1. Banks, B., McGrath, K.: E-Learning Content Advisory Paper. FD Learning (2003) 11-12
2. Heins, T., Himes, F.: Creating Learning Objects With Macromedia Flash MX (2002) 8-9
3. Spicklemire, S., Frienly, K., Spicklemire, J., Brand, K.: Zope Web Application Development and Content Management, New Riders (2002) 17-36
4. Jager, D.: Using UML for software process modeling. Software Engineering, ESEC/FSE'99, Proceedings, Springer (1999) 91-108