SCORM-COMPLIANT LEARNING SYSTEM WITH ANNOTATION CAPABILITY - POWER RTE

Dowming Yeh, Han-Yung Yang, Wen-Shen Liu
Graduate Institute of Information and Computer Education , National Kaohsiung Normal Universit,
116 Hou-Ping 1st Road, Kaohsiung, 802, Taiwan

Keywords: e-Learning, Annotation, SCORM, Software Reengineering.

Abstract: SCORM (Sharable Content Object Reference Model) standard simply records learner's evaluation grade and traces their learning path. However, learning portfolios contain not only learner's evaluation grade and the browsing sequence of their learning contents, but also annotations which the learner add onto learning objects, for example: comments, highlighting marks, digests, and so on. Therefore, a popular open source SCORM-compliant browser - ADL Sample RTE only records grades for pre-test/post-test and learning path, lacking the capability of storing annotation information. This research applies software reengineering techniques to enhance ADL Sample RTE so that learner can store annotation information with the enhanced version. The software Reengineering process is divided into three phases. Static analysis is performed first to dissect the system architecture, followed by dynamic analysis, which is proceeded interactively through web fronts to understand how various dynamic pages startup. Finally, annotation functionality is added in system reconstruction phase. The outcome of the reengineering process is called Power RTE. Besides its annotation power, Power RTE also enables sharing annotations among its users; therefore, learning objects with annotation information could be reused on this e-Learning platform.

1 INTRODUCTION

As e-Learning technology matures and its application flourishes, more and more digital educational materials are produced. However, courseware is not the only factor for successful e-Learning implementation. Learning systems also play an important role in leading to the success of e-learning. In the past, the contents of teaching materials were authored by teachers to cover the need of all learners. But once the content is on line, it is fixed and seldom changed. The different needs of individual learners cannot be addressed. Although there are adaptive learning systems under research and development, learners are still constrained by the material presented by the learning system.

A truly successful e-learning system should be centered on learners and empower learners with various sorts of learning tools to construct his own knowledge. Tools useful in traditional learning such as ink annotations could also be useful in the e-learning setting. Ink annotations serve as the most convenient medium to make ideas clearer in documents, in such forms as sketches or handwritten text notes (Fogli et al., 2004, Yang et al.,2004). In learning systems without the capability recording users' annotation, a learner can only read the learning objects in the system, and he cannot mark his ideas or obtain ideas marked by him or others in the past (Chong and Sakauchi, 2001). The learning effect would be lower than systems that support learners with annotation capability.

In November 1997, the Department of Defense and the White House Office of Science and Technology Policy launched the Advanced Distributed Learning (ADL) Initiative. The missions of the initiative are to solve the problem of the communication between each e-learning system and learning objects sharing. Among these things, SCORM specification describes a run-time environment (RTE) for executing reusable learning objects. RTE contains three parts: launch mechanism, application program interface (API), and data model. Learners (client) launch the learning objects (LOs) with mechanism. API manages the pass of learners' information. Data model defines how to access data such as learners' portfolio (Mor and Minguillón, 2004).

Yeh D., Yang H. and Liu W. (2006).
SCORM-COMPLIANT LEARNING SYSTEM WITH ANNOTATION CAPABILITY - POWER RTE.
In *Proceedings of WEBIST 2006 - Second International Conference on Web Information Systems and Technologies - Society, e-Business and e-Government / e-Learning*, pages 383-386
DOI: 10.5220/0001243003830386

RTE does not address issues on annotation; therefore, all its implementations do not possess any annotation functionality. In this work, an open-source RTE implementation, ADL Sample RTE 1.3 beta 3, is reengineered to produce a new learning system with annotation capability, named Power RTE.

Our software reengineering techniques and the original system, ADL Sample RTE are discussed in section 2. The reengineered system are presented in section 3. The related work is discussed in section 4. Finally, conclusions are given.

2 RTE REVERSE ENGINEERING

After evaluating the complexity, interactivity, and functionality of the ADL Sample RTE, we adopted a reengineering process to construct Power RTE. The two major steps of reverse engineering process are as follows:

(a) Static analysis: During this phase, the system is analyzed using some tools to discover its structure such as module relationships.

(b) Dynamic analysis: In this step, two slices would be executed. One is to examine the runtime behavior of the system such as the interaction between server and client. The other is to find out what source code is triggered to execute.

The Reconstruction phase is the most complex one of the four phases. The main tasks of this phase are to design, develop, and deploy the evolved system. At last, a new system will be produced in this step. After reconstruction, the reengineered system is evaluated to determine whether it conforms to requirement or not.

Good tools are prerequisite to the successful execution of a reengineering work. Tools used in this study are Macromedia Dreamweaver and Borland J++ Builder (BJB). Dreamweaver is useful in static analysis and reconstructing application interface in the last phase. BJB supplies code structure window that is very powerful in tracing source code. BJB is mainly applied for dynamic analysis and modifying the program code.

Any SCORM materials contain a XML file, named ‘imsmanifest.xml’. It describes the general information of the material such as the internal structure, course names, and so on. In the process of course browsing, RTE system would parse this file to present the course structure and all course units.

2.1 Static Analysis

In the static analysis phase, we used Dreamweaver to discover the structure of ADL RTE. Dreamweaver supplies site-map capability that searches and lists all files related to the home page of a web site. In our case, we set LMSMenu.jsp as the home page, and Dreamweaver provides the site map based on LMSMenu.jsp as shown in Figure 1. The root of the tree structure is LMSMenu.jsp and the leaves are the related jsp files, java files, and so on.

Since the focus is on annotation capability and annotations are closely related to browsing functions, we searched for course browsing. And it is easy to identify the files with the function “View Registered Courses”, viewCourses.jsp and sequencingEngine.jsp, from the site map.

2.2 Dynamic Analysis

The files identified from the previous phase are analyzed further to understand their behaviors. We adopted event-trigger method to analyze source code with BJB. During the phase, a series of code tracing is made. We start with LMSMenu.jsp by clicking the “View Registered Course” button in LMSMenu.jsp in BJB. BJB returned the viewCourses.jsp automatically.

Module viewCourses.jsp displays a list of courses that a learner registers for. We just have to analyze what will happen when one of the course names is chosen. And we find that the selected course name is transferred to the module sequencingEngine.jsp. File sequencingEngine.jsp determines which item should be launched in the current course. It responds to the events such as Next-Launch or Previous-Launch.

We also discovered the code for gathering the path of single learning object by locating the code session.setAttribute("pagePath", nextPath) in sequencingEngine.jsp. File LMSFrame.jsp contains the API Adapter applet and the buttons of the Run-time Environment such as login, next, previous, suspend, and quit.

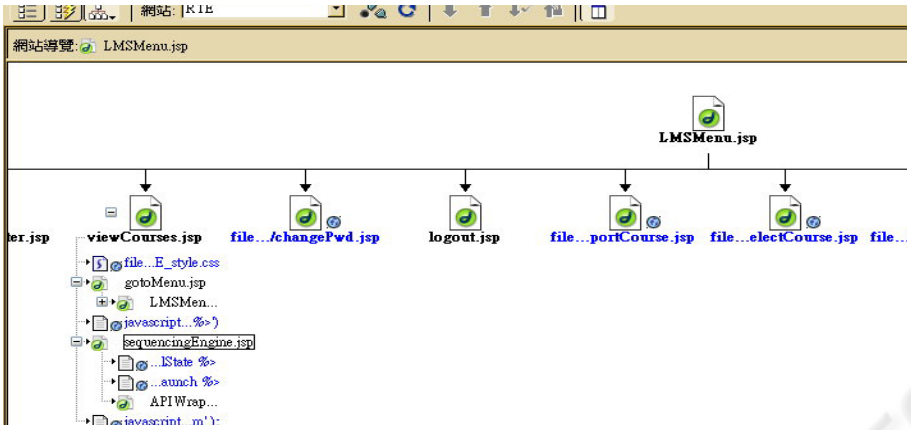

Figure 1: Site Map.

By discovering the presentation code of the buttons and the path code of the single course content in this phase, we are ready to build a new button for annotation function, and the content can be transferred to a new annotation web page by the path of a learning object.

3 REENGINEERING POWER RTE

After reverse engineering ADL Sample RTE to determine the modules that are to be modified, the next step is to introduce the annotation functionality to the original RTE code. The presentation of course content in RTE is still in HTML format, and since most annotation functionalities such as highlighting and underlining are in fact realized by general HTML editors. Again, an open source HTML editor solution is sought and a tool called HtmlArea is chosen to be integrated into RTE to realize the annotation functionality. HtmlArea is implemented in Java Scripts and can be loaded into any browsers easily.

As indicated by the result of dynamic analysis, the command buttons available in browsing course content are defined in LMSFrame.jsp. Therfore, a button for invoking annotation functionality is added in LMSFrame.jsp and the proceduce invoked is defined in a file, anno.jsp.

The procedure in anno.jsp finds the source file of the course content, load the file into annoContent area, and invoke the HtmlArea to provide the annotation functionality. From the previous analysis, the path of the course content is set by the module sequencingEngine.jsp, and we can access the name of the path through the method of the *session* object:

```
object path = session.getAttribute("pagePath")
```

Once the name of the path is accessed, the content of the course can be loaded with the following function:

```
response.sendRedirect("http://localhost:8080"+path)
```

The functionalities of HtmlArea are customized and removed of features not related to annotation. The remaining functions are

- Font setting, including setting a particular font, size,
- Type setting such as bold, italic, and underline
- Color setting for foreground and background color of text for highlight
- Commentary insertion: The image insertion function in HtmlArea is modified so that a user can insert commentary with text or image and the content where a commentary applies would be marked with an icon. When the user browses the content afterwards, he can review the commentary by moving the mouse over the icon. Finally, the annotation information added by the user is stored by the Save button.

The stored annotation information can be shared with others with the same mechanism for content sharing since all the annotation information are in fact embedded in the HTML files contain the course content. Therefore, an export function is provided in LMSMenu.jsp to package the annotated content back to a SCORM format for sharing with other learners with readers that support SCORM 1.3 beta 3.

4 RELATED WORK

Many annotation systems are based on XML. One of

them is HATS (Hypertext Annotation and Trail System) (Kim et al., 2004). HATS is based on WebDAV, which is a XML-based protocol to support collaborative authoring and manage namespace. HATS is implemented as a plug-in facility onto the Mozilla browser. When a user invokes the annotation functionality, a menu frame containing an input area and previous annotation information would be displayed. All the annotation information is inputted through this menu frame; hence, the original page content stays intact. HATS can also provide annotation to PDF files and even multimedia files with pictures.

Ramachandran et al. develop a system for pen-based annotation information (Ramachandran and Kashi, 2003). Its user applies pen-based input devices to annotate web pages. The annotation is stored in XML format to record various attributes such as ink point, text area, and so on. Since SCORM is also based on XML, the extension of these works to annotate SCORM learning objects may be plausible, but integrating the XML structures require certain amount of work.

Besides XML standard, some researches follow industrial standard such as RDF. Annotea is a web-based annotation sharing infrastructure following open RDF standard (Kahan and M.-R. Koivunen., 2001). Annotea treats annotation as additional information of the original documents, and stores annotations on several annotation servers. Annotea achieves annotation sharing and reuse through open RDF. When a user wants to annotate some texts in the document, he simply marks these texts and presses the associated key, and the annotation information is then stored in RDF format.

5 CONCLUSION

This research aims to develop a learning content presentation system, Power RTE, which support annotation functionality and comply with SCORM standard. On such a platform, learners could annotate their learning materials by making their notes, highlighting key point, and so on to improve comprehending and reviewing these digital materials as well as personalizing their learning processes.

Power RTE is based on ADL RTE 1.3 Beta 3. Through software reengineering techniques, ADL RTE 1.3 Beta 3 is analyzed statically and dynamically so that modules for annotation functions may be introduced and integrated. The annotation functionality, including font setting, type setting, highlighting, and commentary insertion, is derived from another open source solution, HtmlArea. The annotation produced by a user can be packaged into a SCORM object and shared to other learners.

The future development of Power RTE is to support more diverse types of annotation, say, allowing voice or even video annotation. The opinions of users on Power RTE should also be collected to adapt and enhance its functionality. Finally, the effectiveness of Power RTE in promoting learning effect needs to be studied carefully to prove whether such tool meets the goal of its development.

REFERENCES

Chong, N. S. and Sakauchi, M., 2001 “Creating and sharing Web notes via a standard browser.” SIGCUE Outlook 27, 3, pp. 4-15.

Fogli, D., Fresta, G., and Mussio, P. 2004. “On electronic annotation and its implementation.” In Proceedings of the Working Conference on Advanced Visual interfaces, ACM Press, New York, NY, pp. 98-102.

Kahan and M.-R. Koivunen., 2001 Annotea: an open rdf infrastructure for shared web annotations. In The tenth international World Wide Web Conference on World Wide Web, Hong Kong.

Kim, S., Slater, M., and Whitehead, E. J., 2004. WebDAV-based hypertext annotation and trail system. In Proceedings of the Fifteenth ACM Conference on Hypertext and Hypermedia (Santa Cruz, CA, USA, August 09 - 13, 2004). ACM Press, New York, NY, 87-88.

Mor, E. and Minguillón, J., 2004. E-learning personalization based on itineraries and long-term navigational behavior. In Proceedings of the 13th international World Wide Web Conference on Alternate Track Papers &Amp; Posters (New York, NY, USA, May 19 - 21, 2004), ACM Press, New York, NY, 264-265.

Ramachandran, S., Kashi, R., 2003 "An Architecture for Ink Annotations on Web Documents," icdar, p. 256, Seventh International Conference on Document Analysis and Recognition (ICDAR'03) - Volume 1.

Yang, S. J., Du, V. M., Shao, N. W., and Chen, I., 2004. Applying Personalized Annotation Mechanism to E-Documentation. In Proceedings of the E-Commerce Technology For Dynamic E-Business, IEEE international Conference on (Cec-East'04) - Volume 00 (September 13 - 15, 2004). IEEE Computer Society, Washington, DC, 142-145.