COLLABORATIVE COURSEWARE AUTHORING BASED ON SCORM METADATA
Timothy K. Shih¹, Jason Chi-Shun Hung², Wen-Chieh Ko¹, Wen Chih Chang¹, and Nigel H. Lin¹
¹Department of Computer Science and Information Engineering, Tamkang University, Taiwan
²Department of Information Management, Kuang Wu Institute of Technology, Taiwan
e-mail: tshih@cs.tku.edu.tw, jhung@cs.tku.edu.tw

ABSTRACT
Sharable Content Object Reference Model (SCORM) has become an important standard of courseware format and platform development in distance learning. We follow the metadata definition of SCORM 1.3 draft, and developed a system which supports collaborative courseware authoring. The system is able to suggest an optimal sequence, with a set of automatically generated lecture components. A CSCW-like collaborative environment is also implemented under the Windowing environment. A demonstration of the system is available at: http://elresearch.mine.tku.edu.tw/.

1. INTRODUCTION
Presentation authoring is essentially important for distance learning. One of the difficulties which discourage instructors to offer distance learning courses is due to the tedious and time consuming process of developing high quality learning materials. To cope with the inconvenience, Sharable Content Object Reference Model (SCORM) [1] allows reusable contents to be separated from their aggregations. With additional description of each lecture component (i.e., metadata), reusable components can be searched and reused easily. However, efficient courseware authoring system based on the SCORM specification is still under development. Thus, the development of authoring tools and learning management systems (LMSs) are encouraged by the Advanced Distributed Learning Initiative (http://www.adlnet.org). In this demonstration, we will show a newly developed authoring system based on the SCORM 1.3 Draft Specification.

2. THE COURSEWARE SYSTEM
According to the draft specification of SCORM 1.3 [1], we use the following metadata in the automatic generation of lectures:

- **Identifier (hierarchical number 1.1):** A globally unique label that identifies the learning object.
- **Educational/Semantic Density (5.4):** The degree of conciseness of a learning object, which is independent of its difficulty level.
- **Educational/Interactivity Level (5.3):** The degree to which the learner can influence the aspect or behavior of the learning object.
- **Educational/Learning Resource Type (5.2):** The model of learning object, such as figure, table, exam, lecture, experiment, etc.
- **Educational/Interactivity Type (5.1):** Predominant mode of learning supported by this learning object.
- **Educational/Typical Learning Time (5.9):** Approximate or typical time it takes to work with or through this learning object for the typical intended target audience.
- **Educational/Difficulty (5.8):** How hard it is to work with or through this learning object for the typical target audience.

Most of these items are optional according to the SCORM Metadata Application Profile Requirements [1]. However, we believe these elements represent important sources of the underlying presentation knowledge. Our system encourages the user to provide the above information in each Sharable Content Asset (SCA). In addition, we follow the specification of the AICC prerequisites scripting language. The prerequisite specification uses the identifier (1.1), which is represented in a consistent name space through our system. The four metadata items (5.1 to 5.4) are used in a set of activity definition rules. We believe that, a SCA may have different usages under different lectures. Thus, the definition of our courseware generation rules can be divided into two levels. The first level (the activity rules) is a precise definition of the roles of each SCA. However, to relax the author from using tedious activity rules, the second level rules only include 4 choices of activity elements, which is easy to use. As a consequence, we
define a set of first level activity rules and another set of secondary rules for courseware generation. The authoring interface in figure 1 is to facilitate drag-and-drop design of SCORM-based courseware. The interface has three levels of design windows (on the left side) and two preview windows (on the right side). On the first level, the author can use one of the following types of SCAs:

- **Private Lecture Component** is created and used by an individual designer. An author name tag is associated with the SCA.
- **Public Lecture Component** is stored in a common library. Public lecture components can be used by any designers.
- **Extracted Lecture Component** is extracted form a public or private lecture for reuse.

On the left side of the first level design window, three buttons are used. The load component button (i.e., ![Load Component](image)) allows users to input a private lecture component. The import component button (i.e., ![Import Component](image)) adds a public lecture component. It is also possible for the user to load a lecture which contains several components and to extract these components for reuse. The delete button (i.e., ![Delete Button](image)) allows the user to delete an object in general. The second level design window allows the user to organize a set of lectures. Each lecture can be generated according to the selection of activity rules. The new button (i.e., ![New Lecture](image)) creates a new lecture. The user can use drag-and-drop mechanism to add lecture components to the new lecture, which has its detailed content illustrated in the first preview window (on the right side, with seven buttons). There are two types of lectures:

- **Private Lecture** is created and used by individual designers.
- **Public Lecture** is loaded from the public library.

Based on prerequisites of SCAs, the deduced structure of a lecture is a DAG (i.e., Directed Acyclic Graph). However, due to limited flexibility of DAG visualization, we display the DAG as a tree, with duplicated nodes shown in bounding boxes. The user can use activity rules to design a lecture, and use the import lecture button (i.e., ![Import Lecture](image)) and the export lecture button (i.e., ![Export Lecture](image)) to load and save any lecture. Note that, on the top of the second level design window, a post condition window allows the user to check the summary of metadata with respect to a lecture. The third level design window allows the user to create a courseware, which is a collection of lectures. When the user click on the link lecture button (i.e., ![Link Lecture](image)), the system will sort the lectures according to their prerequisite relations.

In the first preview window, seven buttons are used. The first five buttons allow the user to select an activity element: ![Lecture](image) for Lecture, ![Exam](image) for Exam, ![Training](image) for Training, ![Experiment](image) for Experiment, and ![Any](image) for Any. The save button (i.e., ![Save Button](image)) allows the user to save a lecture for future reuse. The second preview window is a read-only box, which allows the user to browse the content of a lecture in the courseware.

In addition to the courseware editing tool, we also developed a CSCW environment, which allows a group of designers to assemble lecture components through an online discussion. Figure 2 shows such a tool, which has a similar functionality to the second and the third design windows of the courseware authoring system, with a simple chat room tool for on-line discussion. The CSCW environment maintains a revision control mechanism. A private lecture has a different icon (i.e., ![Private Lecture Icon](image)). While an author check out the lecture, the icon is changed (displayed as ![Check Out](image)). However, a public lecture (i.e., ![Public Lecture Icon](image)) can not be check out. And, there is no author name tag attached to a public lecture.

### 3. CONCLUSIONS

The demonstration will last about 10 minutes for a sample run. We use the system to design a courseware for the data structure course. Commercializing the system is our future work.

### REFERENCES