Object Oriented Architecture for Extensible Learner-Adaptive Environment

Kiyoshi Nakabayashi[a, b, c]**, Yousuke Morimoto**[a]**, and Yoshiaki Hada**[a]

[a]*National Institute of Multimedia Education, Japan*
[b]*Center for e-Learning Research and Application, Nagaoka University of Technology, Japan*
[c]*Graduate School of Instructional Systems, Kumamoto University, Japan*
naka@nime.ac.jp

Abstract: We propose a flexible architecture that is capable of function extension for a learner-adaptive self-learning environment. We have introduced the concept of a "courseware object", which is a program module that is used to implement various educational functionalities, and, thus, the proposed architecture allows incremental function extensions while maintaining the existing functionalities. To implement the functions of SCORM 2004 and its extensions we have investigated the basic behavior and interactions of courseware objects.

1. Introduction

Ensuring the interoperability of learning content and reusability is vital for providing high quality e-learning with a rich learning experience. On the other hand, learner-adaptive functionalities, which allow flexible content presentation by taking account of the learner's current level of understanding, are an effective mechanism to improve learning outcomes [1-2]. However, the interoperability and reusability of learner-adaptive content has not yet been widely established. One of the reasons is that conventional learner-adaptive systems lack a framework for function extension. Without such a framework, extending the system or adding new functions that will improve learning effectiveness is difficult because these extensions need to be authorized as new standard specifications. This authorization process takes a long time. Also, adding new functions may cause trouble for ensuring the reliability of existing learning contents because they cannot be executed correctly on the extended system. Thus, achieving both content-system interoperability and system-function extensibility in conventional learner-adaptive systems was difficult.

To overcome this problem, we propose a new learning system architecture that achieves both extensibility of learner-adaptive functions and the interoperability of learning content. We thus introduce the concept of a "courseware object", which is a program module that is used to implement various educational functionalities. Incremental function extension is possible by adding new courseware objects. Since the existing functions are not affected, it is assured that existing content always work correctly. Early investigations have been made concerning the basic behavior of courseware objects and interactions between courseware objects. On the basis of the courseware object architecture, our aim was to implement the functions and extensions of SCORM 2004 [3], which is a standard specification for learner-adaptive systems that use a hierarchical content structure.

2. Issues of Conventional Learner-Adaptive Systems

In the past evolution of learner-adaptive systems [1-2], it has become common to employ a system architecture, as shown in Figure 1, that separates content and platform. In this configuration, the content consists of learning material which are specific to a particular learning subject with a particular learning goal, and the platform implements common learner-adaptive functionality, which is independent of a specific learning subject or learning goal. Using a configuration that has content separate from the platform means that it is much easier to design learner-adaptive content. This is because the designer may concentrate on the learning subject or learning goal without worrying about how to implement learner-adaptive functionality in detail.

The drawback of this configuration is the lack of a framework for function extension. Once the platform is designed and implemented, it is difficult to extend the platform to add new functionalities because existing learning contents designed before platform extension may not work correctly on the extended system. Moreover, to achieve system interoperability, these extensions need to be authorized as new standard specifications, but this authorization process takes a long time. It is also necessary to update existing platforms to meet new specifications, which is also time-consuming. Thus, achieving both content-system interoperability and system-function extensibility in the conventional learner-adaptive systems was difficult. SCORM 2004, a standard specification for learner-adaptive systems, employs the same configuration resulting in the lack of function extensibility.
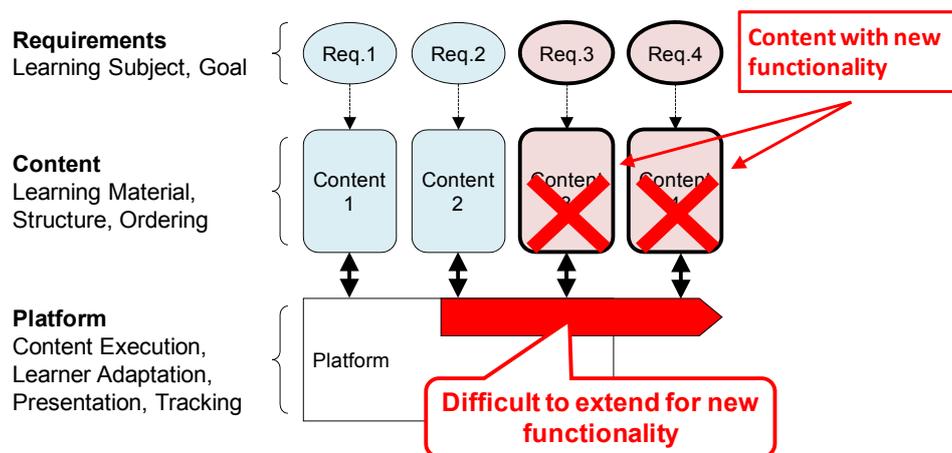

Fig. 1. Configuration of Conventional Learner-Adaptive System.

3. Design of Proposed Architecture

In order to overcome the problem described in the previous section, we propose a new learner-adaptive system architecture that is capable of achieving both function extensibility and system interoperability. To accomplish this goal, the proposed architecture introduces the concept of "courseware object", which is a program module used to implement various educational functionalities such as learner-adaptation, material presentation, and learner tracking, i.e., functions usually embedded in the platform in a conventional configuration. As shown in Figure 2, in the proposed architecture, the courseware object is clearly separated from the platform. In this configuration, incremental function extension is possible by adding new courseware objects. Since this addition does not affect functions previously implemented by existing courseware objects, existing content always works

correctly. Moreover, courseware objects can be distributed with contents and thus enables existing platforms to be immediately updated for newly developed functionalities. This eliminates the long time-lag that results from conducting the standard development process and installing platform updates.
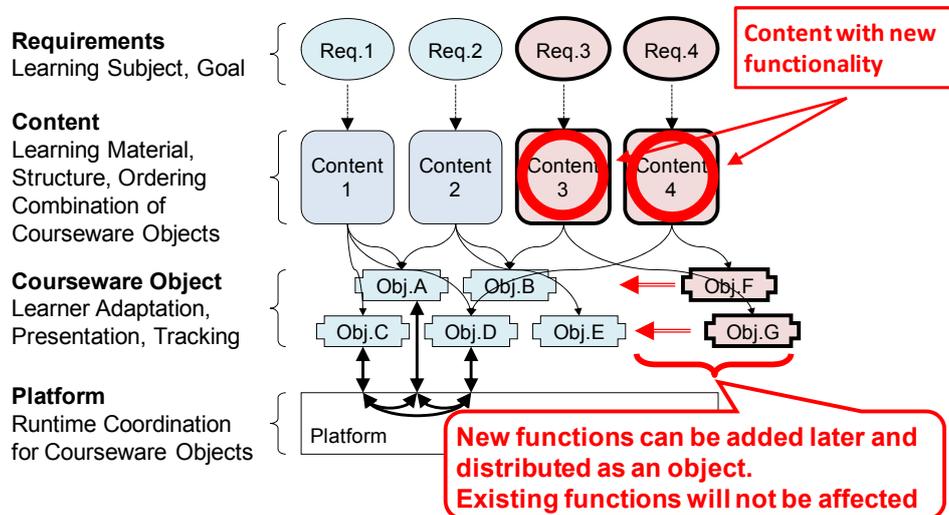

Fig. 2. Configuration of Proposed Learner-Adaptive System.

4. Design of SCORM 2004 Compliant System on the Proposed Architecture

SCORM 2004 is a standard specification for learner-adaptive systems for hierarchically structured learning content [3]. The specification is designed as in the conventional configuration described in Figure 1. The platform implements "sequencing" behavior, which provides learner-adaptation functionality based on learner tracking information. Sequencing behavior is specified by the procedural pseudo code defining basic actions such as forward and backward traversal in the hierarchical content and "skip" or "retry" conditional on the tracking information. These actions are primitive elements, which are controlled by the "sequencing rule description" in the content to implement high-level functions such as "provide hints" or "take remedial lesson". However due to their diverse variations, implementing such high-level functions requires complicated sequencing rule description. It is also difficult or impossible to implement functions that are based on various strategies such as a state-transition machine for role-play type learning material [4] or the mathematical model such as item-response theory for adaptive testing [5].

On the basis of on the proposed architecture shown in Figure 2, we examined the feasibility of implementing a full set of SCORM 2004 sequencing behaviors as well as to incorporate higher-level functions. In the design, courseware objects with sequencing, tracking and user-interface functionalities are assigned to each node of hieratical content structure defined by the manifest file. Some of the courseware objects are designed to implement SCORM 2004 sequencing behavior and others are designed for high-level functionalities. By associating courseware objects with various functionalities to the course structure nodes, a learner-adaptive environment that is capable of dealing with the SCORM 2004 compliant manifest file as well as incorporating higher-level sequencing functions will be possible. This environment allows both maximum utilization of existing SCORM 2004 content and extension of learner-adaptive function which is difficult to be implemented by the current SCORM 2004 specification.
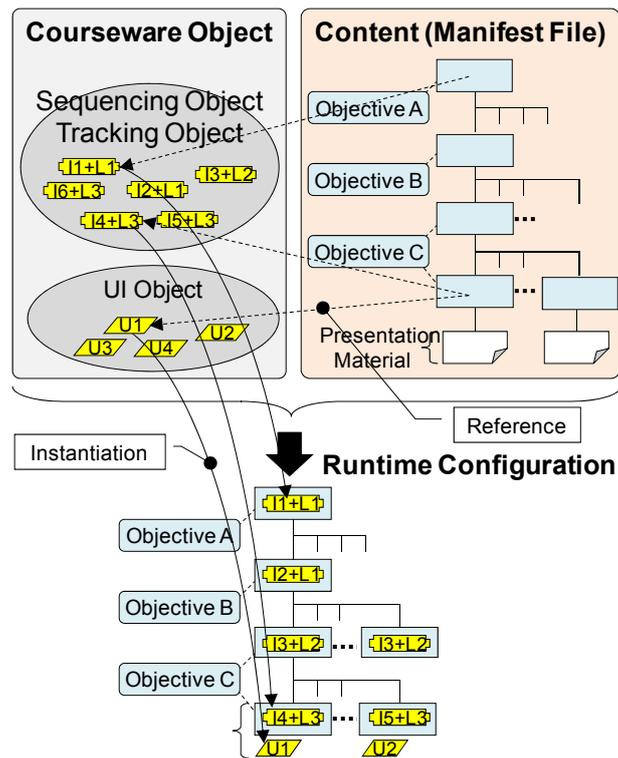
**Fig. 3. Configuration of Proposed System for Hierarchical Content.**

5. Further Issues

We need to discuss several issues related to the design and implementation of the proposed architecture.

The most essential issue is the investigation concerning the behavior and communication of courseware objects. To achieve the goal of the proposed system, we need to be able to combine various courseware objects designed by different designers at different times so that they work together. For this purpose, the standard behavior and communication schema of courseware objects need to be defined. A possible schema for communication between courseware objects is shown in Figure 4. In this schema, a leaf node, which is currently presented to the learner, receives the next command from the learner. If the node is not able to deal with the command, it is passed to its parent node. This is repeated until it encounters a parent node capable to deal with the command at a certain level. The parent node then tries to select suitable child node, which is presented to the learner as the next learning material. This selection is performed in accordance with sequencing behavior implemented in each courseware object assigned to each node. Similar communication schema need to be designed for updating tracking information.

Other issues include the extension of manifest file format defining courseware structure and programming environment to implement the proposed architecture. It is necessary to extend current SCORM 2004 manifest file format so that it is capable of dealing with the assignment of courseware object to each content node. Programming environment issues need to be considered in terms of capability for object-oriented programming as well as the independence from specific programming language. In order for efficient implementation of the proposed object-oriented architecture, the programming environment must provide enough facilities to support object-oriented programming such as data encapsulation and message passing. On the other hand, the programming environment should not depend on the use of a particular programming language. A way to implement

courseware object should be considered so that it can be implemented both as a program module using certain programming language and as a Web-service that is independent from a particular programming language.
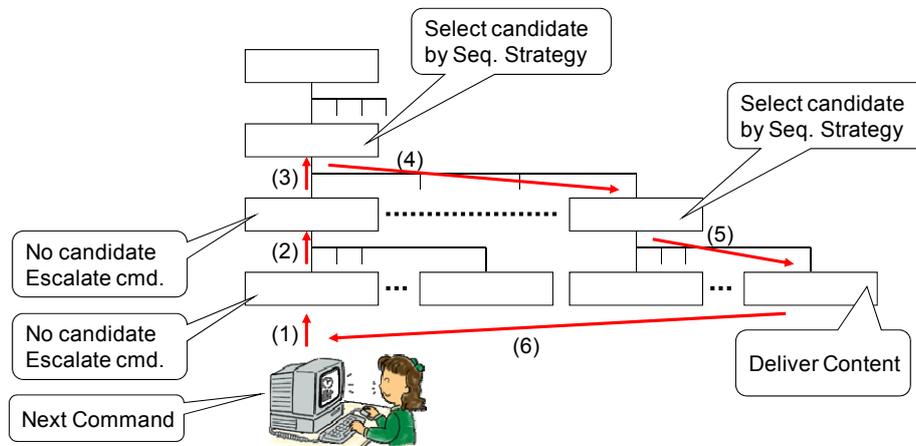
Fig. 4. Communication Schema for Courseware Objects.

6. Conclusion

We discussed a proposal to use of a flexible learner-adaptive architecture that is capable of function extension. By introducing the concept of a "courseware object", which is a program module that implements various educational functionalities, we are able to propose an architecture that is capable of incremental function extensions while maintaining existing functionalities. Investigations have been made into the basic behavior and communication of courseware objects that implement the functions of SCORM 2004 as well as higher-level functions. This configuration maximizes the backward compatibility of existing SCORM 2004 content and allows extension of learner-adaptive function which is difficult to be implemented by the current SCORM 2004 specification. Future work includes making a detailed design of communication schema between courseware objects, manifest file extension, and the programming environment. We are planning to implement prototype system by March, 2009 to demonstrate the feasibility of the proposed architecture.

Acknowledgments

References

[1] Wenger, E. (1987) *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge.* Morgan Kaufmann, San Francisco, CA.
[2] Murray, T., Blessing, S. and Ainsworth, S. (Eds.) (2003) *Authoring Tools for Advanced Technology Learning Environments.* Kluwer Academic Publishers, Dordrecht.
[3] Advanced Distributed Learning Initiative (2006) *SCORM® Shareable Content Object Reference Model SCORM 2004 3rd Edition.*
[4] Nakabayashi, K. et al., (1996) An Intelligent Tutoring System on the WWW Supporting Interactive Simulation Environment with a Multimedia Viewer Control Mechanism, *Proc. WebNet 96.*
[5] Wainer, H. (2000) *Computerized Adaptive Testing: A Primer.* Lawrence Erlbaum Assoc. Inc., Philadelphia, PA.