Architecture for 3D Navigation and Authoring of Distributed Learning Object Repositories

M. Anwar Hossain, Md. Abdur Rahman, Abdulmotaleb El Saddik and Pierre Lévy
Multimedia Communications Research Laboratory (MCRLab)
School of Information Technology and Engineering (SITE)
University of Ottawa, Canada
{anwar, rahman, abed}@mcrlab.uottawa.ca, plevy@uottawa.ca

Abstract

In this paper we propose the architecture of LORNAV, a virtual reality tool to navigate distributed Learning Object (LO) repositories in 3D Virtual Environments (VE). The presented architecture facilitates authoring of LOs within the VE to create new aggregated objects and represent them in SMIL format.

1. Introduction

A number of learning communities are using IEEE Learning Object Metadata (LOM) [1] standard to describe and categorize learning objects and to store them in local and distributed repositories. This metadata in the repositories can be displayed using traditional one and/or two dimensional user interfaces while accepting the limitation of information organization, poor presentation, and lack of interaction. On the contrary, 3D visualization can provide: 1) easy navigation of the information space allowing better user interaction with the virtual objects; 2) improved perception and understanding of the displayed data through the existence of visual metaphors that indicate trends and 3) the capability to display more data at one time [3, 4]. The above formulate the need for a 3D environment to visualize LO repositories in order to navigate and identify learning objects in a natural way different than just "filling in electronic forms" [5].

Generating 3D scenes dynamically is a challenging factor when there are millions of data to be extracted from the distributed repositories. A possible solution to this problem is to generate personalized 3D environment, while allowing the user to navigate beyond his interests. This goal can be achieved by maintaining individual user profile which ensures that the 3D virtual environment can be generated with the 3D objects on demand thereby improving the rendering performance at the client side.

To be more appealing to the learning community, the 3D navigation system of the LOM repositories should have an authoring interface to combine the contents of several multimedia objects together. This would help authors constructing their own presentation using existing learning objects while navigating in the VE.

In this paper we propose the architecture of our system, called Learning Object Repository Navigation and Authoring in Virtual environment (LORNAV). The proposed architecture presented in Figure 1, allows 3D navigation of distributed learning object repositories; interaction with the displayed objects by touching and moving an object; viewing the associated metadata corresponding to that object; reading/playing the content of that object using appropriate viewer and authoring the objects in 3D virtual environment. The cornerstone in our approach is to generate personalized 3D environment for the navigating user as well as to combine 3D authoring capability with the navigational system. The authoring process adopts SMIL (Synchronous Multimedia Integration Language) [2] to produce complex multimedia presentation.

The remaining of the paper is organized as follows. Section 2 presents related research work followed by Section 3 that describes the proposed system architecture. Section 4 provides a summary of the implementation and Section 5 concludes the paper with an overview of the future work.

2. Related work

There are several techniques proposed to visualize, navigate, interact, and query database systems in virtual environment. A recent work [6] focuses on generating dynamic personalized 3D worlds and describes sample implementation in the e-commerce

domain. The system uses user profile and/or usage data to deliver personalized content to the customer.

Another similar work has been done in [7], except that they use X-VRML's capability to parameterize 3D virtual scenes. By using a language different than VRML, the authors were able to achieve some flexibility in terms of database access, queries, user preferences and the current system state.

The research work presented in [11] proposes an agent-based adaptive interaction system that monitors current user behavior in order to adapt the 3D scenes for subsequent user interaction to keep him knowledgeable about the current state of the system. The system implements user profile in combination with recorded usage data to generate enhanced 3D world on top of the basic 3D world.

All the above works focus mainly on 3D visualization of databases, personalization of virtual worlds and adaptive generation of 3D scenes. However there is no authoring facility in those systems which is an obvious requirement in the context of visualizing learning objects.

Though a lot of research has been invested to build authoring tool to generate SMIL presentation, there is, to the best of our knowledge no tool that provides 3D navigational interface to author multimedia learning objects. As an example, the Smiley [8] authoring tool allows editing a multimedia document in the WYSIWYG paradigm. Smiley also helps an author to specify the temporal organization of documents by providing an execution report displayed through a timeline view. However, the tool may not be suitable to hook up with the 3D virtual world.

GRiNS [9] creates a SMIL document by specifying the logical structure, the virtual timeline and the playout view. However, it does not provide a 3D virtual reality interface where the user could navigate and select the 3D objects to author.

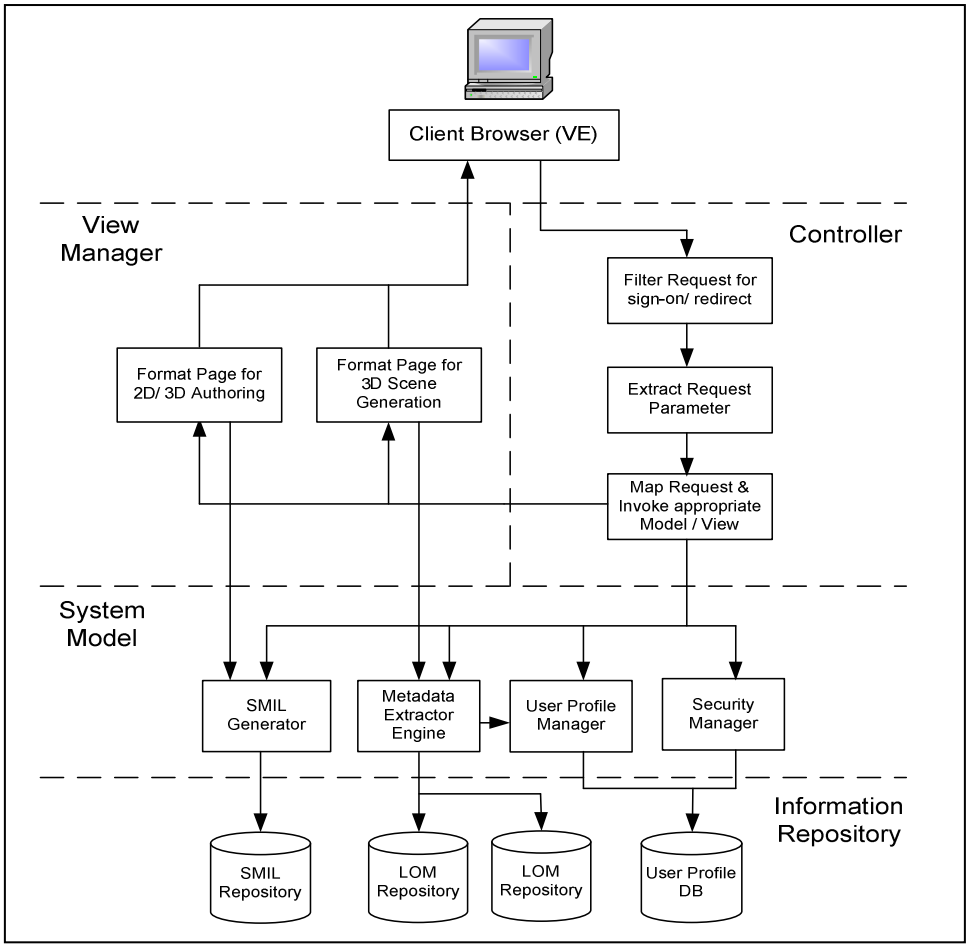

Figure 1. The LORNAV architecture

3. System architecture

3.1 High-level overview

The proposed LORNAV system requires multi-tier client-server architecture to accommodate all the features mentioned above. Figure 1 shows the high level architecture of the system which is based on the three-tier architecture (client-tier, web-tier and Enterprise Information System tier) to increase system performance, to allow flexibility, to facilitate maintainability and reusability and to support scalability and interoperability while hiding the complexity of the distributed processes from the user [12].

The following section provides a brief summary of the functional units in each of the tiers.

3.1.1 The client tier. The client tier provides a browser-based 3D virtual environment where the user navigates and authors the displayed learning objects. The client requires additional 3D plug-in to support the 3D visual interface. The two main modules in this tier are:

3D virtual environment: It is the 3D front-end of the system where user navigates around and interacts with the 3D representations of the learning objects. It also contains the authoring interface.

Authoring interface: This 3D interface is integrated with the front-end and only accessible to authorized users. In this environment authors view, drag and drop, change temporal and visual properties of the existing learning objects displayed in the virtual environment in order to create new combined learning objects.

3.1.2 The web tier. It follows the Model-View-Controller (MVC) pattern [10]. The functional modules in this tier are the Controller, the View Manager and the System Model.

The Controller is the core component of the architecture. It uses several active components (Java Servlets) for processing HTTP/HTTPS requests and selecting appropriate views based on the user profile.

The View Manager has two basic responsibilities. These are: formatting the page dynamically using several different templates and send it to the client's browser according to the request of the controller; and receiving the required data from the model to generate the view. To perform these tasks the View Manager has the following two components:

3D scene generator for navigation: It generates appropriate 3D views according to the interaction of the user with the system and its current state. This module produces the 3D representation of the learning objects through the use of different predefined templates. The 3D scene generator module receives commands from the controller to determine what view to generate and accordingly interacts with the appropriate model to obtain the needed data to complete the view generation process.

2D/3D authoring interface generator: This component allows the user to enter some SMIL specific authoring parameters and generates the 3D authoring environment.

The System Model encapsulates all the functional logic of the system. It separates the information repository from Controller and View Manager. Any request for retrieving data from the EIS layer is handled by the appropriate model. This ensures information abstraction from the client side. This layer consists of the following four components:

Security manager: This module performs several security related tasks including managing user account, checking privilege for each user and preventing hackers from accessing the system.

User profile manager: This component manages individual user profile that is used to determine the personalized view in the 3D VE.

Metadata extraction engine: This module is responsible for retrieving LOM from the distributed repositories in either of the two cases: a) to create the initial view and b) to populate subsequent views when user navigates around.

SMIL generator/extractor engine: This component encapsulates the functionality of generating/extracting SMIL presentation which refers to the aggregate learning object resulting from the authoring process.

3.1.3 The EIS tier. The EIS tier is composed of several information repositories such as LOM repositories, SMIL file repository and User information database.

LOM repository: This is a distributed source of learning object metadata. The actual repository format could be any of the popular database information systems such as SQL, XML and legacy.

SMIL file repository: It stores the SMIL presentation files generated by the system.

User information DB: It holds users' account information and their profile.

3.2 Interaction model

In this section we describe the major interaction model of the system through a use case and two sequence diagrams.

The use case diagram presented in Figure 2 shows a high level interaction model in which several actors

might exist, e.g., learner, teacher and authoring user. Possible interactions of a learner include logging on to the system, managing his/her profile to personalize visualization, navigating around the VE system, interacting with the objects in VE, and searching objects of interest.

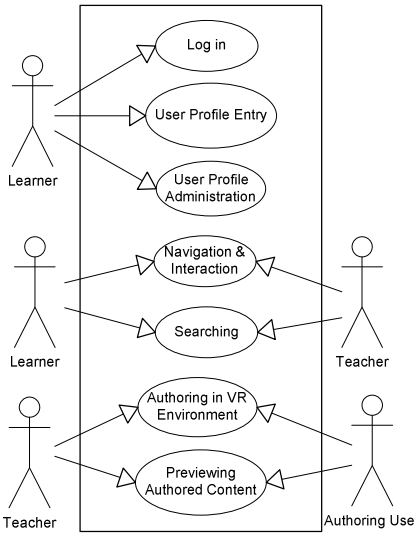

Figure 2. USE CASE diagram of LORNAV

The interactions of a teacher and/or an author with LORNAV include the learner's capabilities as well as the capability of authoring multimedia objects in VE to create new combined objects.

The sequence diagrams shown in Figures 3 and 4 represent two main functionalities provided by LORNAV:

a) ***Personalized 3D View generation overview:*** As presented in Figure 3 a personalized 3D environment is based on a user profile in the system which is used to validate the user and to deliver personalized content to the client's virtual environment. When a user logs in the system with his/her identification, the Controller module intercepts and redirects the request to the Security Manager to validate the user information. Once the validation is done, the controller invokes the Metadata Extractor to extract the LOM data from the distributed LO repositories. LO repository contains information about learning objects such as title, keyword, language, learning resource type, learning context, and intended user etc. These set of metadata are essential to build the 3D representation of learning objects in VE.

The Metadata Extractor first requests current user profile from the profile manager. A user profile usually stores information such as language preference, types of learning resources, preferred format of learning resource, learning context etc. The Profile Manager extracts the needed profile information from the Profile Database and returns them to the Metadata Extractor in XML format. Upon receiving the profile, it invokes an asynchronous web service call and sends an XML-SOAP message to the LOM repository. In response the LOM repository uses web services technique to send the requested LOM data in XML format. The Metadata Extractor then sends a handle of this LOM data to the Controller which selects the appropriate view generator. The view generator contacts the appropriate model with the handle to receive the XML data. It then parses the incoming data, uses some predefined 3D templates representation of the learning objects, associates the metadata with the template and finally generates a 3D virtual environment reflecting the user's profile.

b) ***3D authoring system overview:*** The authoring system works on top of the 3D virtual environment stack. Figure 4 shows the sequence diagram of the authoring process. Prior to authoring, the user needs to create his/her VE first as described in the above section. As part of the distributed system, the authoring system also follows the MVC pattern for its basic operations. Any user request for the initialization of the authoring environment is captured by the Controller module to check user permissions and authorization. A predefined privilege is required for authoring.

Once validated, the Controller selects the initial authoring view which requests the user to format the layout and regions for his/her presentation. The definition of the layout requires several iterations presented through a loop in the sequence diagram of Figure 4. The input parameters in each of the above authoring steps are stored in user session. The "next view" selected by the Controller generates 3D authoring view based on the user's input parameters stored earlier in the session. Now the user has an authoring panel as part of the VE. The user navigates in the VE, chooses his/her objects of interest and drags and drops the chosen objects to any region in the authoring panel. The same process can be repeated until the user is done with the selection of the multimedia objects he/she wants to combine. In the final step the user needs to specify the temporal relationship among the learning objects.

We use SMIL's synchronization elements for temporal relationship. These include the <seq> element, the <par> element, and the class of media object elements, e.g., <img>, <video>, <audio> and <text>. The <seq> element defines a sequence of elements in which the elements are played one after the other. The <par> element defines a simple parallel time grouping in which multiple elements can be played back at the same time. In the synchronization window the author chooses media objects and configures their

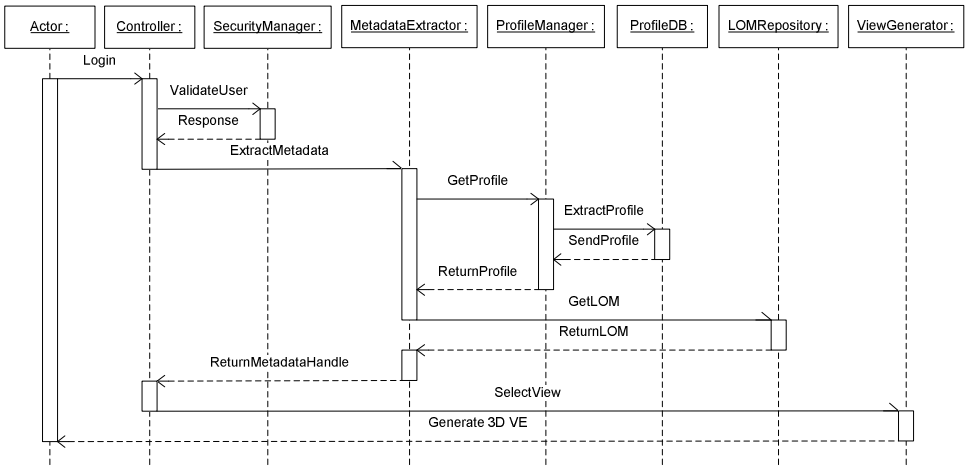

Figure 3. Sequence diagram for personalized dynamic 3D view generation

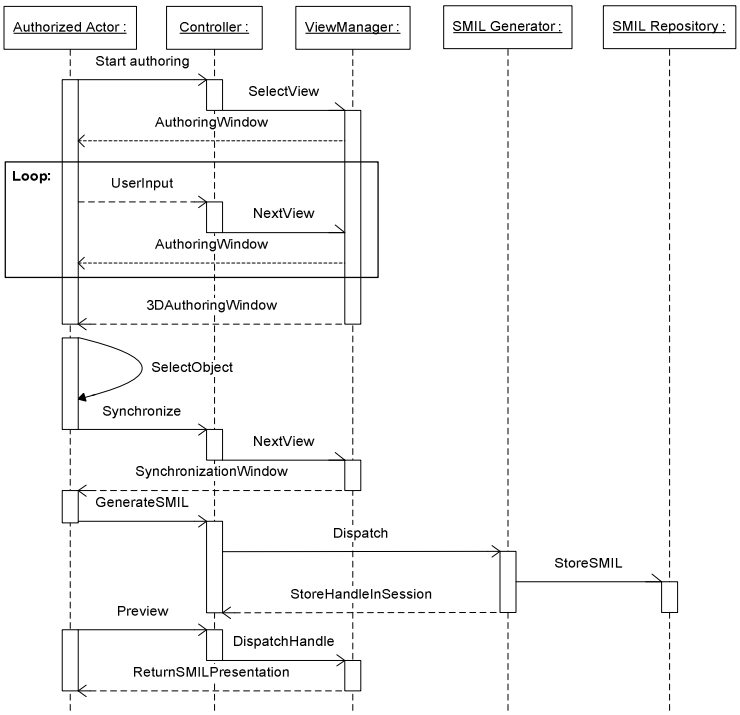

Figure 4. Sequence diagram for 3D authoring

temporal relationship. This window gives users a high level view of performing those complex synchronization tasks. Once temporal relationship information is submitted, the user can generate and preview his/her presentation. The Controller sends the request to the SMIL generator module which contains the SMIL generation logic. This engine creates the SMIL file, saves it to a repository for future editing and sends a handle back to the Controller which in turn sends it to an appropriate view. The view automatically selects the external viewer, e.g., RealPlayer for the content to be presented in.

4. Implementation

A prototype of the LORNAV architecture as discussed in Section 3 has been implemented. We used Apache Tomcat as the application/web server that hosts the controller module, the view generators, and system models. The controller and system models are implemented using several Servlets while the view generators are based on JSP technology. The 3D user interfaces are implemented using VRML and are generated dynamically by the view generator.

We bypassed the EAI to communicate with VRML scene and instead used active server side components to dynamically generate and manipulate the 3D objects. The reason behind is that the VRML EAI has to rely on Microsoft VM which is based on java 1.1 and does not support Servlets, JSPs and web services technology we have used.

We used Cortona as VRML plugin for Internet Explorer 6.0 to test the display of the 3D scenes. LORNAV supports both http and https protocol for the communication between client and server. The server side modules use SOAP protocol to communicate with the distributed repositories using web services technology.

5. Conclusion and future work

The LORNAV architecture presented in this paper implements several well known techniques as described in [6, 7, 11] to generate dynamic 3D scenes using server side scripting language (e.g. Java Server Pages) and to construct personalized virtual environment using user profile. LORNAV differs from many existing work by taking into consideration the distributed learning object repositories as its domain. It provides several added features such as adopting software engineering design pattern; web services based metadata extraction; integrated authoring interface within the virtual environment and generating SMIL presentation by combining several learning objects.

The current version of LORNAV does not support collaboration among multiple users. At present the interaction with the 3D objects in the virtual environment is done via mouse and keyboard. However, work is ongoing to include haptic devices so that users have a real life feeling of the 3D objects.

The authoring capability provided by LORNAV allows the user to combine several Learning Objects and generate new aggregated learning objects in SMIL format. The generation of metadata for newly aggregated objects is out of the scope of this paper. However metadata aggregation is an ongoing research and we will try to integrate the first results of this issue in future development.

Acknowledgments

The authors acknowledge the financial assistance of the National Capital Institute of Telecommunications (NCIT) and of the Natural Sciences and Engineering Research Council Canada (NSERC) through its Learning Objects Repositories Network (LORNET).

References

[1] IEEE Learning Object Metadata, final draft standard, Jul. 15, 2002 (IEEE 1484.12.1), http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf

[2] W3C Recommendation, "Synchronized Multimedia Integration Language (SMIL 2.0) Specification", Aug. 7, 2001, http://www.w3.org/TR/2001/REC-smil20-20010807.

[3] R. S. Owor, "A Data Model and Architecture for Hypermedia Database Visualization", *Proc. 7th Int'l Conf. 3D Web Technology* (Web3D'02), pp. 121-126, Tempe, Arizona, USA, Feb. 24-28, 2002.

[4] G. Geisler, "Making Information More Accessible: A Survey of Information Visualization Applications and Techniques", Jan. 1998, http://www.ils.unc.edu/~geisg/info/infovis/paper.html

[5] E. Duval and W. Hodgins, "A LOM Research Agenda", *Proc. 12th Int'l Conf. World Wide Web* (WWW2003, May 20-24, 2003, Budapest, Hungary).

[6] L. Chittaro and R. Ranon, "Dynamic Generation of Personalized VRML Content: a General Approach and its Application to 3D E-Commerce", *Proc. 7th Int'l Conf. 3D Web Technology* (Web3D'02), pp. 145-154, Tempe, Arizona, USA, Feb. 24-28, 2002.

[7] K. Walczak and W. Cellary, "Building Database Applications of Virtual Reality with X-VRML", *Proc. 7th Int'l Conf. 3D Web Technology* (Web3D'02), pp. 111-120, Tempe, Arizona, USA, Feb. 24-28, 2002.

[8] M. Jourdan et al., "Authoring SMIL documents by Direct Manipulations during Presentation", *World Wide Web*, Balzer Science Publishers, vol. 2, no. 4, Dec. 1999.

[9] D.C.A. Bulterman et al., "GRiNS: An Authoring Environment for Web Multimedia", World Conf. ED-MEDIA, Seattle, Washington, USA, Jun. 19-24, 1999.

[10] Gamma E. et al., *Design Patterns*, Addison-Wesley, ISBN 0-201-63361-2, Oct. 1994.

[11] A. Celentano, M. Nodari, and F. Pittarello, "Adaptive Interaction in 3D Virtual Worlds", *Proc. 9th Int'l Conf. 3D Web Technology* (Web3D'04), pp. 41-50, Monterey, California, USA, Apr. 5-8, 2004.

[12] Stearns B. et al., *Designing Enterprise Applications with the J2EE™ Platform, Second Edition*, http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/DEA2eTOC.html