2006-1487: REDUCING THE DEVELOPMENT COSTS FOR ACTIVE AND INTERACTIVE LEARNING OBJECTS THROUGH WEB-BASED COLLABORATIVE AUTHORING

Elliot Diaz, Polytechnic University of Puerto Rico

Elliot Diaz Research Assistant within the eLearning Research Laboratory at the Polytechnic University of Puerto Rico, where he is pursuing a Master in Computer Engineering. Elliot holds a BS in Computer Science.

Carlos Pacheco, Polytechnic University of Puerto Rico

Carlos Pacheco is a programmer with the Connect To Learn Project which is developing web-based authoring for the collaborative authoring of learning objects. He completed a BS in Computer Engineering summa cum laude at the Polytechnic University of Puerto Rico, he also holds a BS in Natural Science from the University of Puerto Rico. Besides his work with the project he is currently seeking fellowship opportunities for working towards a PhD. He can be reach at braincode@prtc.net .

Carlos Rosaly, Polytechnic University of Puerto Rico

Carlos Rosaly is a part time research assistant with the Connect To Learn project. He is working towards a Master in Computer Engineering at Polytechnic University of Puerto Rico. Carlos holds a BS in Computer Science with honors from the University of Puerto Rico.

Michael Figueroa, Polytechnic University of Puerto Rico

Michael Figueroa is a research assistant with the Connect To Learn project at Polytechnic University of Puerto Rico where he completed his BS in Computer Engineering and will be pursuing a Master in Computer Engineering.

Eddalis Batista, Polytechnic University of Puerto Rico

Eddalis Batista is a research assistant with the Connect To Learn project. She holds a BS in Computer Engineering magna cum laude from Polytechnic University of Puerto Rico, where she is continuing her studies towards a Master in Computer Engineering.

Othoniel Rodriguez-Jimenez, Polytechnic University of Puerto Rico

Othoniel Rodriguez-Jimenez is Full Professor at the Polytechnic University of P.R. (PUPR), and Associate Director for the Computer Engineering program. Under grants from NSF and Puerto Rico’s Industrial Development Corp., he obtained support to establish the eLearning Research Laboratory that is developing tools for web-based collaborative authoring of learning objects. In a previous life, Otho spent 15 years in the computer industry as a principal engineer and project manager. Besides teaching for the past 13 years, he has consulted for the Pharmaceutical Industry, and the Puerto Rico government. He is a member of the ASEE, IEEE, ACM, and a registered Professional Engineer in Puerto Rico. He holds a PhD in Computer Engineering and Computer Science from the University of Missouri-Columbia, an MSEE&CS from Massachusetts Institute of Technology, and a BSEE magna cum laude from the University of Puerto Rico. When not teaching, or behind a computer in the Lab, or menthoring students, you can find Otho and his children at the many local beaches, or hiking the island’s unique tropical rain forest.

Reducing the Development Costs for Active and Interactive Learning Objects through Web-based Collaborative Authoring

Abstract

The development of active Learning Objects with interactive capability, has demonstrated to be extremely costly for a single individual, or even a single organization. This paper presents an approach to reducing the costs and facilitating the development of learning objects, and what is being done in terms of developing tools towards this goal. Our project is developing open-sourced web-based tools for the collaborative authoring of learning objects. We believe that collaboration will help spread the development costs among several interested parties, thus making possible efforts which otherwise would not be cost-effective for a single individual or organization. Collaboration essentially makes "scalable" the authoring process by involving several distributed parties, empowering them with web-based collaboration and authoring tools, and tying them together under a common set of goals and a shared ontological concept space for what is being developed. This common ontology for the course concepts is mapped into learning experiences through active and interactive learning objects.

The learning objects created through the web-based tools being developed, meet interfacing standards proposed by Shareable Content Object Reference Model (SCORM), Instructional Management Systems, IEEE-Learning Technologies Standards Committee, and World Wide Web Consortium-Scalable Vector Graphics, among others. These which we like to call "Open Learning Objects" are considered "open" because they provide explicit XML metadata for the learning objects internals. This metadata defines a five (5) layer model which captures different aspects of the learning object, such as: 1) conceptual content, 2) visual representation, 3) learner interaction, 4) internal sequencing, and 5) integration. Our approach facilitates the creation and capturing of multiple versions for a learning object such as for different learning styles, different skill levels, or different cultural and language aspects, by providing this explicit XML vocabulary for tagging the internals of the learning object.

Our web-based tools facilitate the organizing of a team of remote collaborators under a course coordinator and provide for breaking-up of a course in terms of groups of learning objects. These tools allow for the collaborative creation of each learning object, define inter-learning object navigation using the SCORM Sequencing and Navigation Specs, and package these resources using the IMS Packaging Specs. We provide a set of web editors for authoring each layer and their integration into a layered open learning object.

Finally, we provide some modest facilities for introspection of the authoring and learning processes. This is done for authoring, by capturing metrics about the learning object development process, thus helping the identification of best practices. For learners, the Learning objects are instrumented to capture information about the interaction with the learner, facilitating both the learner modeling and the improvement of the learning objects. The presentation will demonstrate prototypes for most of these tools, provide details on project progress, and will identify additional useful tools which we plan to build on top of the current system.

Introduction

Authoring costs have placed limitations on the level of experimentation and testing which educators would have liked to carry out with learning objects technology. This is particularly so if what is desired is the production of multimedia, interactive and active, learning objects. By these we mean learning objects that provide a substantial degree of engagement with the learner. Today this is a requirement for educational learning object material which may hope to meet the expectations of the constructivist learning-theorist, not to mention the expectations of students stepped into game technology and similar highly engaging experiences. On the other hand the possibility of having a single author develop such learning objects is severely limited by the time-consuming nature of the task. The time and difficulty involved seems to be the result of two major factors: first, the opaque and complex nature of the learning objects produced by the current set of commercial tools which rely on proprietary formats in part resulting from the lack of open standards for the description of the internals of the learning object itself, and second, the lack of tools to assist and support the author in carrying out the job and being able to do so in a collaborative mode.

Our laboratory has been carrying out a funded[7] effort to develop open sourced tools that address some of these obstacles to the efficient creation of learning objects. First, we have adopted open standards[1,2,3,4] for specifying the Sequencing and Navigation among the learning objects, and for their graphical representation. Second, we rely on previous work[6] which developed a simple but sufficiently powerful open model for learning objects, based on multiple layers represented as metadata-rich extensible markup language (XML) files. Third, we have embarked into the development of a set of collaborative web-based authoring tools geared at making the authoring process "scalable" by encouraging and coordinating the collaboration of geographically dispersed authors.

The rest of this paper discusses the open learning objects multilayer model and the tools we are developing for their deployment through the web. We describe the authoring tools already developed or under development and their main features. Then we address collaboration, and show how the simple learning object model and the tools are intended to be used with a central coordination entity. In order to limit the length of this paper we do not delve into the technical details of the current implementation. We continue with the description of an open source business model scenario for the continuous development and improvement of the tools, by providing services to organizations using them, such as authoring learning objects, their deployment into repositories, their use within Learning Management System (LMS), and their continuous improvement. Finally we show the current state of development for the project and the expected outcomes.

Learning Objects Metadata

For our work we have adopted what we consider an important distinction between the more generic XML-based tag standards dealing with the description, instructional design and navigation of learning objects from a perspective *external* to them, and the less generic but critically important description/representation of the *internals* of the Learning objects.

Adopting a model to the internal description of learning objects and in particular a layered model is important in our view because it provides several key advantages, some of these are:

1. A separation of concerns among the layers based on their functions which leads to easier creation of authoring tools for such layered learning objects.
2. The simplification of the authoring process itself by allowing to focus work at the different layers by specialized roles if so desired.
3. A better introspection of the authoring process by allowing more points for measuring and assessing the progress of the authoring tasks.
4. The possibility for introspecting the learner-to-learning object interaction (when an appropriate layer for interaction description is provided) by the capture of the interaction trace for subsequent analysis, which can support the modeling of the learner and improvements to the learning object itself.
5. Use of the xml metadata tags for the extraction of information about the learning object which can facilitate its classification and indexing within learning object repositories, possibly in a semi-automated way.

All these advantages are the result of the layered inner-metadata strategy adopted and not necessarily of a particular set of XML vocabulary schemas adopted for describing the metadata within the different layers, as long as these sets are mutually consistent.

Current Learning Objects Standards: Outer Metadata

The learning object standards adopted for our development are those from the Instructional Management Systems Consortium (IMS-Consortium) and their Simple Sequencing Specification (SSS), the IEEE-Learning Technologies Standards Committee (IEEE-LTSC) and their Learning Objects Metadata (LOM), and those from the Advanced Distributed Learning-Shareable Content Object Reference Model (ADL-SCORM)[1,2,3] specifically the specifications of the learning object to Learning Management System (LO-LMS) interface. To leverage the open source and Lesser General Public License (LGPL) character of our work we have also adopted the Moodle as our first target LMS with which our open learning objects will interface.

Open Learning Objects: Inner Metadata

The Open Learning Objects being created are described by a five (5) layer model. Each layer is responsible for particular functions as is typical of any layered architecture. The content and behavior of the learning object is captured in a data-driven fashion through the XML tags vocabulary provided by the XML schemas or XML Document Type Definitions (DTDs) used to describe each layer. Note that it is possible to change or evolve these layers with some lateral impacts in the tools used to author the layer, but in general such changes will tend to remain contained within each layer and its tools. This is so because the bridging of the semantic gap between different layers is carried out by the persons authoring subsequent layers. The five layers are: Conceptual Layer, Presentation Layer, Interaction Layer, Animation Layer, and Integration Layer. Here we provide a brief description of each layer, see[6] for a more detailed explanation of the layers..

The Conceptual Layer is responsible to capture as text a large portion of the concepts embodied in the learning object. It helps to guide the author in identifying the meaning of these statements by relying on the set of vocabulary tags. These tags support the embedding of such text within other layers of the learning object during the rest of the authoring process. They may also support their classification and indexing within learning object repositories. The content of this layer depends on the concepts and subject matter of the learning object.

The Presentation Layer is responsible for the graphical features of the learning object. We have adopted the Scalable Vector Graphics (SVG) standard from World Wide Web (WWW) Consortium[4] for producing two-dimensional graphics. One of the advantages of the functional encapsulation facilitated by layering is that we may evolve to a three dimensional format (i.e. X3D) in the future. The scalability features of SVG simplify the deployment of the learning object within a large range of devices with different screen sizes. The SVG viewers for each particular device are responsible for automatically generating the required scaling. There is a hierarchical dependence of the Presentation layer on the Conceptual layer, as the text captured in that first layer, is embedded as SVG text elements in the second.

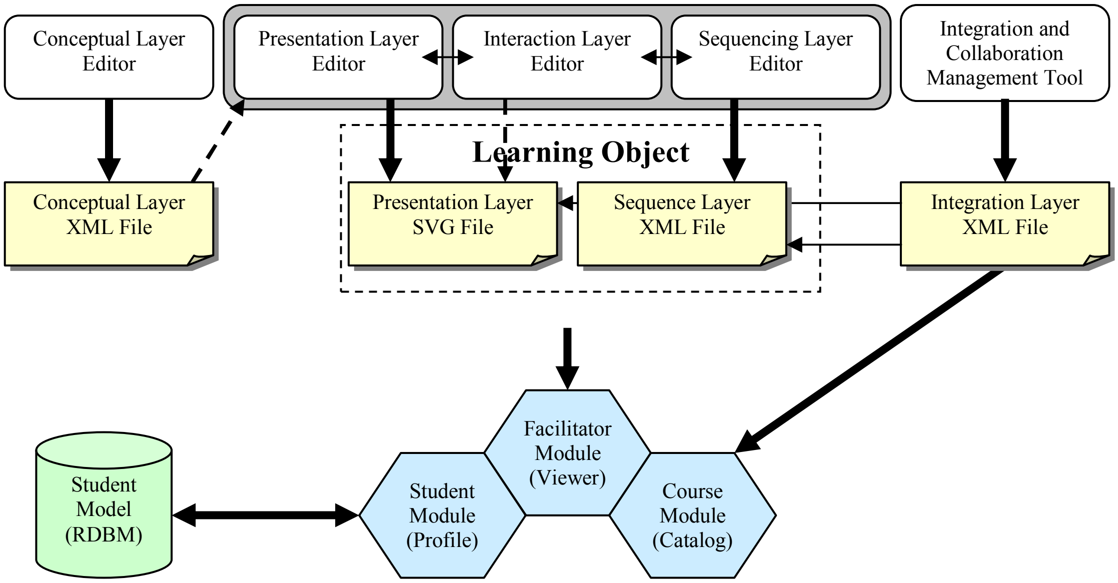
The Interaction Layer is made up of the set of potentially active and responsive regions defined by the grouping of graphical elements in the Presentation Layer. To create this layer one must carry-out a gathering of graphic elements into SVG groups. Once this is done, the author of this layer defines the combination of a group and a primitive mouse or keyboard gesture as a semantic event. These semantic events are used to steer the animation and interaction of the learning object and Learner as specified within the next layer. There is a hierarchical dependence of the Interaction layer on the Presentation layer, as the groups defined in the Presentation layer are required to assign meaning to the semantic events defined by Interaction layer.

The Animation Layer is responsible for specifying the behavior of the learning object and provides it with active and interactive features. The interactive behavior desired is captured by a hierarchical state machine, with states, and semantic event-driven transitions to other states with associated transformative actions applied to specific features and groups of the presentation SVG graph. Each new state is a stable and meaningful configuration of the SVG graph. These transformations include translations, rotations, scaling, transparency and visibility changes. There is a hierarchical dependence of the Animation Layer on the semantic events defined by the Interaction Layer.

The Integration Layer brings together all the XML documents comprising the learning object. It also provides additional explicit classification information for the learning object along several dimensions. For example suitability for particular learner learning styles, languages of instruction, or expected learner skill levels.

Open Learning Objects deployment support modules

There are three (3) main modules which support the deployment of the open learning objects. These are generic modules which are compatible for any number of open learning objects developed following a compatible set of XML vocabulary tags for the different layers. Switching to a different set of XML schemas will require corresponding changes to these modules. The modules are: the Facilitator module, the Learner module, and the Course module. Of these the Facilitator module is deployed at the browser or learner side of the web, while the Learner and Course modules are deployed at the server side of the web. **Figure 1** below provides a simplified diagram of the interaction between the various tools and modules being described below.



Conceptual Layer Editor
Presentation Layer Editor
Interaction Layer Editor
Sequencing Layer Editor
Integration and Collaboration Management Tool
Learning Object
Conceptual Layer XML File
Presentation Layer SVG File
Sequence Layer XML File
Integration Layer XML File
Student Model (RDBM)
Facilitator Module (Viewer)
Student Module (Profile)
Course Module (Catalog)

Figure 1 Collaborative Authoring Tools, Learning Object and Deployment Modules

The Facilitator or interpreter module main responsibility is to carry-out the active and interactive features of the open learning object. It is responsible to load all the documents comprising the integrated open learning object. Facilitator Module is a small client-side JavaScript application able to handle all the necessary learning object components required to deploy it. Even thought, it requires a SVG-enabled browser to render the learning object visual content, the module enriches the visual experience adding time-driven animation(s) and event-driven interactivity to the content.

Using a sequence descriptor (unique for each learning object), this module is capable to handle user-driven semantic events, manipulate the SVG content, to execute JavaScript macros (code fragments), and to compile the user interaction with the learning object. The sequencing descriptor defines all the possible states each learning object can have. Each state defines a unique user interaction environment using relationships between semantic events and defined actions. Using generic JavaScript event handlers, all SVG tags that generate events can respond to user gestures (primitive events) such as mouse and keyboard inputs. Our module is able to capture the user behaviors and gestures, and produce semantic events that are subsequently interpreted using the sequencing descriptor to perform the related actions. Because the SVG file is represented as a Document Object Model (DOM) tree in the browser, the manipulation capabilities of the module include, but are not limited to create, modify or delete any SVG node or attribute. Through macro executions, complex DOM tree manipulation can be performed on the SVG.

The facilitator also stores a trace of all the interaction events with the Learner which it will forward to the Learner Module at the server side whenever the interaction with the open learning object is interrupted. In a subsequent access by the same learner to continue the interaction with an open learning object whose study has been suspended, the saved interaction trace will be loaded together with the open learning object. This will cause the Facilitator module to use the events stored in the interaction trace to re-initiate the open learning object in the same state in which it was suspended. Finally the Facilitator also implements the set of LTSC and SCORM-defined Shareable Content Object (SCO) to LMS interface. This interface allows the open learning object and a SCORM-compatible LMS to cooperate to support the navigation across a set of Learning objects comprising a course which has been packaged according to the SCORM-IMS [2] Simple Sequencing Specification. **Figure 2** below depicts some internal details of the Facilitator module.

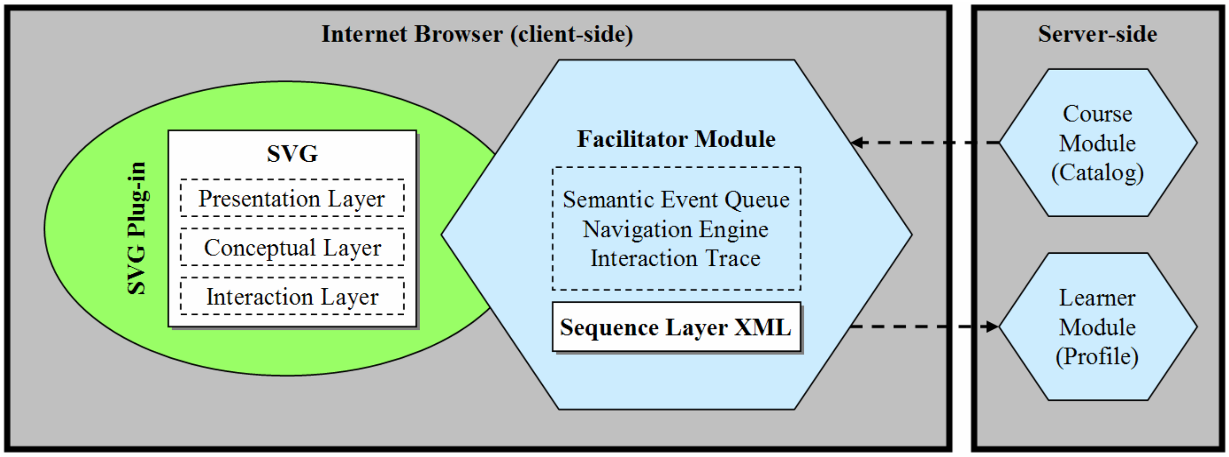


Figure 2 Facilitator module details diagram

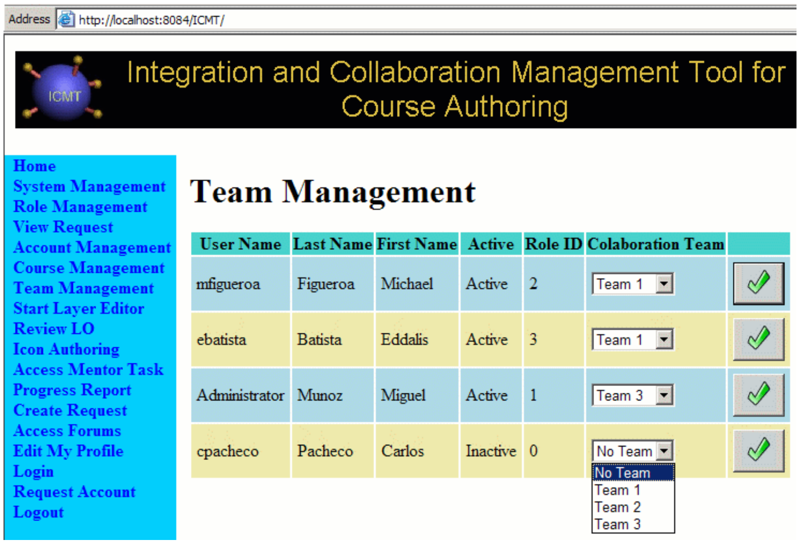
The Course Module has the task of constructing upon request a Hypertext Markup Language (HTML) document with the open learning object embedded as a set of references to the rest of the XML files comprising the open learning object. It also embeds into an open learning object the Facilitator module in the form of JavaScript code and Java applets. The Course module has a LMS-specific sub-module which it uses to receive a request from the Moodle LMS on a particular open learning object for a particular Learner. Additionally the Course module will communicate with the Learner module to receive any previous interaction trace for the learner on an open learning object whose study the learner might have suspended.

The Learner Module is responsible for keeping track of the progress of the learner through the course and of saving the learner interaction trace collected and forwarded by the Facilitator module.

Besides these modules, open learning object deployment also requires the installation and/or activation in the browser of an SVG viewer and the Java Virtual Machine plug-ins.

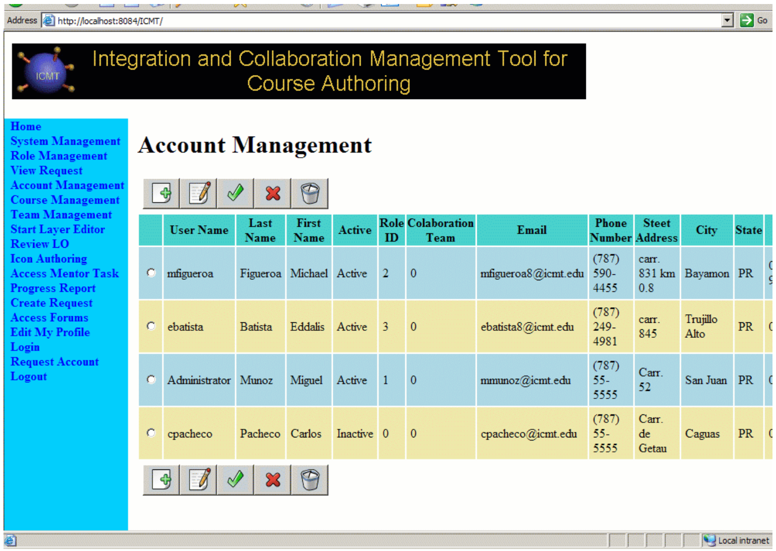
Collaboration Management Tool

The remote team collaboration approach we are pursuing with this tool is one which supports both peer (horizontal) relationships and coordination (vertical) relationships. Toward this end we define a set of roles related to the authoring and coordination tasks required for the development of a Course based on open learning objects. The roles defined for the team members are: Tool Administrator, Course Coordinator, Layer Author, Icon Author, Reviewer, Release Auditor and the Apprentice role. These team-member roles are supported through the creation of a set of accounts which provide selective access to the various open learning object per-layer authoring tools and the open learning objects assigned to each team member in its particular role. This tool will facilitate the collaborative creation, evaluation, auditing, and publishing of courses based on open learning objects. The Course Coordinator will be initially responsible for defining the general outlines for a course composed of open learning objects, to be collaboratively developed. A course is subdivided into groups of learning objects, and each group assigned to a team of collaborators. Every open learning object will have the five layers defined and collaborators will be assigned to different open learning objects related to a course defined by the Course Coordinator. **Figure 3** below shows two screenshots of the Collaboration Management Tool prototype.



Team Management

User Name	Last Name	First Name	Active	Role ID	Colaboration Team	
mfigueroa	Figueroa	Michael	Active	2	Team 1	✓
ebatista	Batista	Eddalis	Active	3	Team 1	✓
Administrator	Munoz	Miguel	Active	1	Team 3	✓
cpacheco	Pacheco	Carlos	Inactive	0	No Team	✓



Account Management

	User Name	Last Name	First Name	Active	Role ID	Colaboration Team	Email	Phone Number	Steet Address	City	State
○	mfigueroa	Figueroa	Michael	Active	2	0	mfigueroa8@icmt.edu	(787) 590-4455	carr. 831 km 0.8	Bayamon	PR
○	ebatista	Batista	Eddalis	Active	3	0	ebatista8@icmt.edu	(787) 249-4981	carr. 845	Trujillo Alto	PR
○	Administrator	Munoz	Miguel	Active	1	0	mmunoz@icmt.edu	(787) 55-5555	Carr. 52	San Juan	PR
○	cpacheco	Pacheco	Carlos	Inactive	0	0	cpacheco@icmt.edu	(787) 55-5555	Carr. de Getau	Caguas	PR

Figure 3 Collaboration Management Tool, Team and Account Management screens

In addition a discussion thread facility is provided to support the horizontal collaboration between team members. Each course and its associated team have a separate discussion thread with sub-threads pre-defined for each open learning object being developed. Here team members can interact asynchronously with other team members and share suggestions for improvements, request support, etc., leading to improving the quality of the open learning object created. A Reviewer will evaluate the content of the open learning object before it is submitted to the Release Auditor for its final approval and release. To improve the quality of the open learning object content, the Reviewer will provide the open learning object Layer Author with suggestions. A Release Auditor will evaluate the final content and quality of the open learning objects to be released for use by learners.

The Icon Author will create a course-specific library of icons that will be used by the collaborators to maintain a consistent graphic vocabulary among open learning objects of the same course. The icon library allows the reuse of icons, eliminating the need to design new icons for each learning object.

An Apprentice will learn the duties and responsibilities of its mentor, who will be either a Reviewer, a Release Auditor, or a learning object Layer Author. Once the proper use of the role have been learned by the Apprentice, if it is requested, the Course Coordinator can change the role status of the apprentice collaborator to the proper role of its mentor.

All users and content creators, as well as the Course Coordinator and the Administrator will have the opportunity to monitor the progress of every aspect of the online content creation process. The parameters currently being monitored include capturing information related to the log-in frequency and duration times, file-sizes, and reported status of the tasks assigned to each team member who is collaborating in remote-fashion through the web. This information is available as a set of reports which provide some introspection on the development process and on the progress and difficulties being experienced by the development team. We expect that this tool will be further instrumented in future versions to capture additional metrics and also report these in more interesting ways.

The distinguished role of Course Coordinator is responsible for setting up the general development tasks. These include defining the Course name and the list of names for the open learning objects to be developed with a brief description of the topic and scope of each learning object. After this stage is completed the list of learning object references is exported as an XML document to be imported into the Sequencing and Navigation Authoring tool for creating the sequencing specification and the packaging of the open learning objects comprising the course. The Coordinator also proceeds to open accounts with particular role privileges for the various collaborators and delegates the development of any number of open learning objects to each of the remotely located collaborating team members. During the instantiation of accounts and open learning objects, the collaboration management tool also allocates directory structures and base template files to be filled during the authoring process with the contents of each layer for each open learning object. While the learning objects are being developed by the rest of the team the Coordinator can work on designing the Navigation specification for the learning objects belonging to this Course package.

Navigation/Integration Authoring Tool

This tools which is currently under development is responsible for specifying the inter-learning object navigation following the IMS-SCORM Simple Sequencing Specifications. The tool provides a user friendly environment in which to specify the navigation rules for clusters of activities many of which are associated with Learning Objects.

Per-Layer Authoring Tools

The Conceptual Layer Editor is a web-based application which allows creating the XML document corresponding to the Conceptual Layer. This document follows a DTD (or Schema) format which captures the contents of all the textual portions of the learning object. This XML file is saved at the server side and is later on provided to the Presentation layer editor. This editor is implemented as a Java Applet which is accessed after login into the Collaboration Management tool in the learning object Editor role. Future versions are expected to resort to other technologies such as Asynchronous JavaScript and XML (AJAX) to lessen the size of the downloaded code.

The Presentation Layer Editor and the Interaction Layer Editor are bundled as a single web-based application. It is invoked from the Collaboration Management Tool, and it is loaded with the latest open learning object SVG document the team member had been editing. In addition it provides for loading into a separate frame the Conceptual layer XML file for that same learning object to facilitate extracting the sentences which will be included as text elements within the SVG document being generated within the Presentation layer editor. The majority of elements and behaviors available for SVG are implemented. Additional features such as the creation, saving and loading of libraries of Icons are also included. The Interaction Layer is embedded into the SVG file by a menu that allows selecting the primitive events to be associated with any

SVG group elements created. Options are provided to switch to the editing of any other learning object if so desired.

The Presentation Layer editor web application is capable of generating graphical content in SVG file format. The editor itself is coded in SVG and JavaScript. The software has been tested with the Internet Explorer using the Adobe SVG plug-in. The software also uses Java's applet and servlet technologies to communicate with the server in order to open and save the SVG files. The editor allows it's user to draw by using lines, paths, rectangles, and ellipses. The user can also insert text, and edit it afterwards whenever he desires. It is capable of modifying or removing the fill and line color. Additionally, the user can change the opacity of the graphical elements. The editor allows the user to scale, rotate, and change the depth or order of a graphical element. The user can also unite several elements in a group, perform any of the changes mentioned and split up the group, if he so desires. **Figure 4** shows element editing.

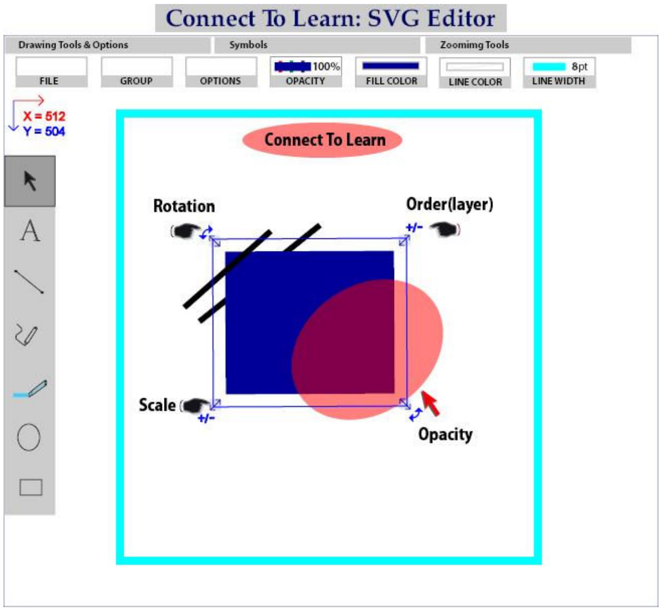


Figure 4 Sample SVG element editing using the Presentation Layer Editor

The editor provides several icons the user can insert into the drawing. It is also currently capable of inserting SVG drawings previously saved to create composite drawings. In the future work shall be performed towards uniting the library of icons and the SVG files saved for the purpose of creating composite drawings. This way a user, upon saving, would be able to specify whether the graphics should be considered a building block, and would be later accessible through the icon library. This will facilitate the insertion process and would make it available to a number of other users. The user would be expected to specify the context of the drawing so that it can be saved in an appropriate library. Other users would load the libraries desired to the icon menu.

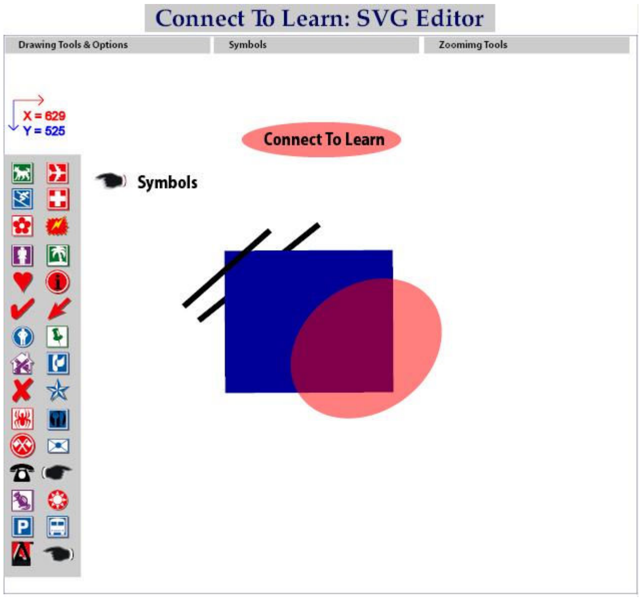


Figure 5 Sample Icon Selection using the SVG Presentation Editor

The editor currently allows users to draw in an infinite canvas. It provides tools for panning (moving to any area of the canvas), zooming-in or -out (without limitations), zooming and moving to the original size and location, and zooming so that the viewed section of the canvas matches the drawing size without altering its aspect ratio (zoom all). The zoom-in function provides two ways of operation: zoom point and zoom window. If the user clicks on the canvas the function would increase the size by 10% and center the spot clicked on the screen. Otherwise, if the user presses the right mouse button and drags it, the application would create a rectangle representing the area of the drawing that will be viewed upon releasing the mouse.

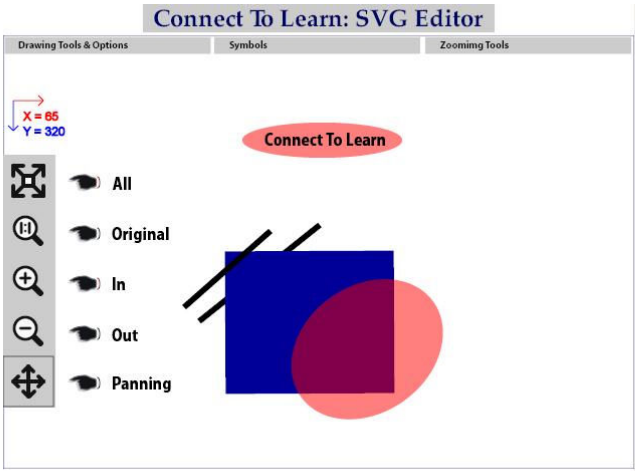
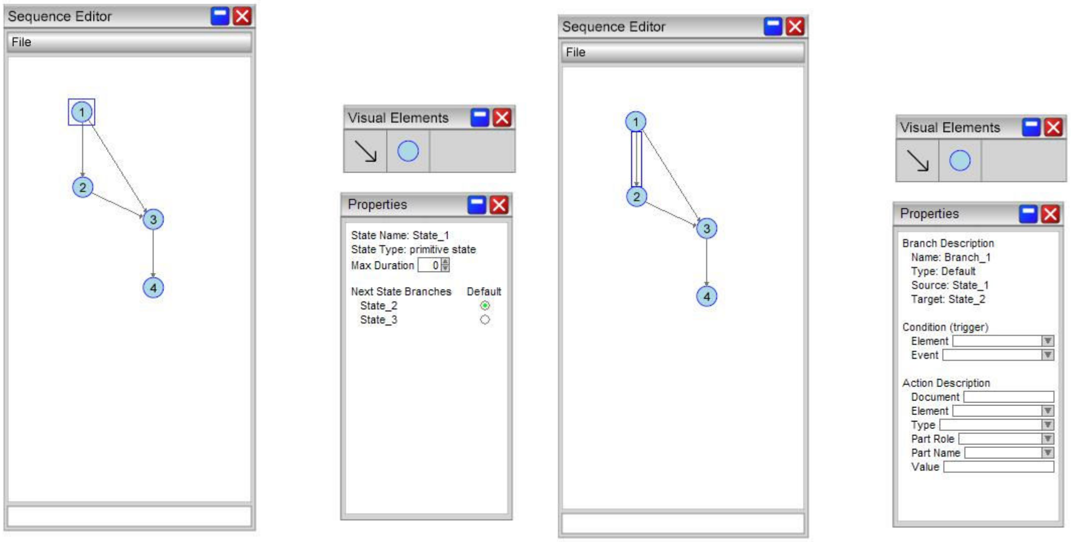


Figure 6 Sample zoom toolbar use for the Presentation Layer editor

The Animation Layer Editor is an SVG/JavaScript web application embedded in the SVG Presentation editor. It is also accessed from the Collaboration Management Tool. It provides a dual view of the SVG file being animated and a graphical view of the hierarchical state diagram being generated. This editor creates a sequence descriptor using a simple graphical user interface (GUI) that represent states and branches (semantic event and actions relationships) using icons. Each branch may represent the transition between two states or specify itself as the target state. The integration of the sequence editor and the SVG Presentation editor allows defining a semantic event selecting any visual representable SVG element and assigning it a trigger (JavaScript event-handler). Actions are defined by a combination of multiple choice selections or writing a macro (JavaScript code fragment) to be executed. At the end, it is possible to have multiple states, each state with multiple branches, and each branch triggered by a semantic event with multiple actions. Finally, the Animation Layer Editor saves the description in an XML file at the server as part of the current learning object Sequence Layer. This editor allows the author to rehearse the interaction with an open learning object up to the state where the animation has been specified to give immediate feedback to the author. **Figure 7** below shows some screenshot of the Animation Layer Editor.



State Name: State_1
State Type: primitive state
Max Duration 0
Next State Branches Default
State_2 ●
State_3 ○

Branch Description
Name: Branch_1
Type: Default
Source: State_1
Target: State_2
Condition (trigger)
Element
Event
Action Description
Document
Element
Type
Part Role
Part Name
Value

Figure 7 Animation Layer Editor: State and Branch Properties

The Integration Layer Editor is provided by using the same XML editing Applet used in the Conceptual Layer editor but instantiating it with the XML schema defined for the Integration layer.

Project Status

Table 1 below depicts the current development status of the various modules described.

Tool	Status
Collaboration Management Tool	Prototype 90% completed
Navigation Authoring Tool	Specifications completed and reviewed

Conceptual Layer Editor	Prototype completed
Presentation & Interaction Layer Editor	Prototype completed
Animation Layer Editor	Prototype 80% completed
Integration Layer Editor	Prototype completed
Facilitator Module	Prototype completed
Course Module	Specifications completed
Learner Module	Prototype 70% completed

Table 1 Web-based Authoring tool development status

The development process has been besieged by student turn-over. Of the 14 students which have participated at one time or another during the last 20 months, 10 have left the project. This has had its toll in terms of lost learning curve, and continuity. If we were to start the project again we would seriously consider hiring 1 or 2 full time programmers and paying them a higher rate, while teaming these with undergraduate or graduate students on a part-time basis.

Conclusion

The development of multimedia, active and interactive learning objects which support research in new technologies for education seems to be an effort-intensive tasks which can benefit from a collaborative development approach. This paper and the project it describes presents one approach which by its open sourced nature, the simple data driven approach, and the web-based distributed collaboration it enables, we believe holds great promise to reach this goal.

Bibliography

[1] SCORM, Sharable Content Object Reference Model, Advaned Distributed Learning Network, Sequencing & Navigation Specifications, http://adlnet.org/.

[2] IMS, Global Learning Consortium, Simple Sequencing Specifications, http://imsproject.org/.

[3] IEEE-LTSC, Learning Technology Standards Committee, Learning Object Metadata (LOM) Standard, http://ltsc.ieee.org/.

[4] W3C, World Wide Web Consortium, Scalable Vector Graphics(SVG) 1.0 Specifications, http://www.w3.org/TR/SVG/.

[5] Rodriguez, O; Shi, H.; Shang Y.; Chen, S., Open Learning Objects: The Case for Inner-Metadata, WWW2002 Conference, Hawai, May 2002

[6] Rodriguez-Jimenez, O., Hierarchical, Adaptive Learning Objects, Doctoral Dissertation, University of Missouri-Columbia, MO, May 2003.

[7] Rodriguez-Jimenez, O., Connect To Learn Project Proposal, Puerto Rico Research and Commercialization Alliance, Puerto Rico Industrial Development Corporation, approved October 2003.