Infrastructure for Development of Intelligent Learning Environments that Manage SCORM Content

Adilson Vahldick\textsuperscript{1} and André Raabe\textsuperscript{2}

\textsuperscript{1} Blumenau Regional University, Brazil, adilsonv77@gmail.com
\textsuperscript{2} Applied Computing Master Program, Itajaí Valley University, Brazil, raabe@univali.br

Abstract: One of the major problems in continuous use of Intelligent Learning Environments (ILE) produced in academic and scientific projects is the difficulty in maintaining and actualizing the content, which is normally produced locally, have many external references or are copies of files such as PDF, DOC or images. These materials have problems to be reused in other environments since they have few resources to improve visual presentation and lack of traceability of what activities the student accessed. This paper presents an infrastructure to allow such environments to use content following the SCORM specification. The advantage of using SCORM content is the portability. Instructional material can be produced in specialized tools and can be use in any systems that are in compliance with the SCORM specification. As an ILE normally actuate promoting adaptation of educational content and pedagogical strategies according to student's individual needs, the infrastructure presented in this paper provides the means for SCORM packages to be customised for ILE environments. The infrastructure was named CELINE and was developed using components paradigm and Java technologies. The results of the development of three ILE that uses CELINE component are also discussed in this paper.

Keywords: Learning management systems, courseware, learning objects, artificial intelligence

1. Introduction

A major challenge to ensure longevity in Intelligent Learning Environments (ILE) use is the maintenance of instructional content. In most of them content is created especially for the system and cannot be easily actualized. In developing an ILE,
the researcher invests most of the efforts in the intelligence of the association between the modules of the system (student, expert and domain), but normally do not worry about making the content itself reusable and easily updatable.

In another context, there is a large effort spent by international institutions for the definitions of standards and specifications to achieve the highest degree of reuse of the instructional material, also known as Learning Objects (LO). Their efforts are directed to the way these objects can be assembled, and disassembled, for being used by learning environments.

One of these organizations is the Advanced Distributed Learning (ADL), which is responsible for maintaining the Sharable Content Object Reference Model (SCORM) specification, which aims to foster the creation of reusable learning content as "instructional objects" within a common technical framework for computer-based and Web-based learning [1]. This specification determines how to describe the contents structure, presentation and sequencing rules. It also specifies how the content can communicate with the system environment, and how this should behave with the data supplied by content.

The content structure follows the model of a tree where each node is called activity. An activity is a set of web pages, and this set is called the Sharable Content Object (SCO). The SCO is an indivisible unity of content, and a SCO cannot refer to another. The order of presentation is totally dependent on the tree of activities.

In SCORM the term Learning Management System (LMS) is used as a synonym of the environment that runs the SCORM packages. The SCORM content is independent of the LMS in which he performs. With this, you can produce and test the material without the presence of these environments, and this role is performed by the content authoring tools for SCORM. As the specification states that the content is web-based, any resource that can be rendered in a browser can be included in that material. There are also authoring tools with advanced features and easy to use, not requiring knowledge of SCORM by producers, such as CourseLab [2] and MOS Solo [3] just to name some.

Aiming to enable ILE to use more sophisticated content such as those produced by the tools mentioned above, this paper presents a component that provides resources for managing and playing SCORM content. The component, called CELINE, has responsibility for the interface layer of an ILE, allowing SCORM packages to be used in the domain model. To use CELINE the LMS developer doesn't need to understand SCORM technical details, it must only have conceptual knowledge of SCORM, that is, he should know that the content is structured in the form of a tree and there are some rules to determine the conditions of delivery of such content. The component was implemented in Java and the tests were conducted in accordance with the SCORM 2004 3rd Edition specification, guaranteed the environments to seal "LMS SCORM 2004 3rd Edition Conformant".

This paper is organized in the following way: in section 2 related works are presented as the infrastructure and adaptation of environments using SCORM. In
Infrastructure for Developing Intelligent Learning Environments that Manage SCORM Content

Section 3 discusses the features and architecture of CELINE component. In section 4 results of environments developed with the component are discussed. In section 5 are presented the conclusions of this paper.

2. Correlate Work

Conducting a search in international journals and conferences, was not found any work that proposes to offer an infrastructure to learning environments to manage SCORM content. There are ready environments that are experiencing some adaptability in the specification, but none of them provides resources for creating a new environment.

[4] define a plug-in in Moodle to generate SCORM courses dynamically, using the knowledge of the student. The courses are created based on a concept called Learning Component, which contains definitions of the content being taught (HTML pages), pre-requisites (knowledge that the student needs to access the content) and knowledge (what the student learned after the content). The plug-in creates the SCO through a plan of actions using a scripting language.

Some works suggest extensions to the SCORM, aiming to set a content that is adapted to meet the needs of the student. These works ([5]; [6]; [7]; [8]) add new elements in the file metadata and content pages. Through these new settings, you can generate a sequence of activities, or adjust the arrangement of elements of the page, or select what items are showing, according to the learning style and characteristics of the student. The difficulty with these proposals is not to belong to the specification, so the environments in compliance will not be able to interpret and carry out such adjustments.

There are also proposals for use of Multi Agent Systems (MAS), on the transformation of learning objects into agents ([9]), about the architecture ([10]) and the adaptability of the sequence of activities ([11]). All these works stand out by incorporating a MAS in the application that want to manage SCORM packages.

3. The Developed Component

The component presented in this paper, called CELINE, was developed with Java technologies for web applications. Its main objectives are allowing SCORM packages to be administered and played as well as interfering during the student's interaction with these packages. The component, connected to the web application, intermediates various resources needed to meet these goals.

By definition, the component is an artefact of software easily attached to the application ([12]; [13]), which contains a set of complete solutions to a problem.
One component is not designed to be extensible, but contains interfaces for customization of their duties. An application uses a component.

Figure 1 shows a high-level vision of ILE architecture that uses CELINE. The JSP pages are rendered based on resources available in the component. The files with SCORM content is stored in a folder outside the application and accessed by CELINE. Through events, the CELINE communicates with the intelligent system in order to decide which content should be presented to the student.

The JSP pages are assembled through a set of tags, which access the features of CELINE. There are tags for the following purposes:

- **Manage courses**: list and register new courses; importing SCORM packages;
- **Manage users**: list and register users. There are two types: administrator and user;
- **Course interaction**: register and cancel the registration of a user to a course, list the courses of a user, view the activity tree, opening a way to interact with it; suspend the interaction to continue in another moment;
- **Show statistics reports**: list data on each course (who were the users who accessed, when was the last time, how long used it, scores, if completed) on each user (how many courses he accessed, total time spent with the courses and when was the last accessed any time that way), and details about the course that the user access (which the activities were accessed, time spent, score).

These features were based on analysis of two environments: *SCORM 2004 3rd Edition Sample Run-Time Environment* ([14]) and Moodle ([15]). The first was selected because it is the reference implementation of the SCORM. The Moodle was adopted as one of the free learning environments most popular and used, and contains a plug-in for implementation of SCORM packages.

The component supports three mechanisms for persistence: XML files, relational database (RDB) and customizable. Through these mechanisms is the component responsibility to record and search the data of registered users and courses, as well as the interactions between the user and the content. The first
option of persistence (XML files) is used when the amount of data (users and courses) is small, or as it develops a prototype of the environment. The mechanism RDB uses JDBC drivers, so that does not fix the manufacturer of the DBMS. The last mechanism allows the developer to adapt the legacy system with CELINE. For example, you may already have a user account, which the developer can use the new system, and in that case, access to such data should be provided with the implementation of the developer.

CELINE communication with the intelligent system occurs with the implementation of a class, based on an interface, and its association with the component. The CELINE using this class for the following situations:

- **List courses**: when a JSP page needs to list the courses available to the user, the intelligent system can specify what courses the student can open, for example, according to the student model;
- **Open a course**: when the user opens a course, the intelligent system can change the structure of the course, for example, according to the characteristics of the student. You can also create a course at that moment, based on pieces of other courses;
- **Changing a structure course**: while the user interacts with the course, the intelligent system can change its structure by adding or removing activities. For example, if the student is not achieving success in solving a year, the system may suggest an activity of enhancing the student;
- **Retrieve and save information**: according to [16], the pages of content can record the answers of the student through a data element called cmi.interactions. Every time that the page provide a value for this element type, the intelligent system is alerted, also when the page need the value of an item. The system can use this information to evaluate the performance of the student, and propose new ways to him.

The component was developed in two layers: the first deals exclusively with the SCORM specification independent of the context where the application will be implemented, and another layer has resources that the application access, the tags and the persistence mechanisms, which are the resources that are specifically of the component. Each layer is a distinct expertise: in SCORM layer the developer needs basic knowledge of programming language (in this case, Java) and depth of the specification, the LMS layer requires specialized knowledge of Java technologies, and low awareness of SCORM. This division allows new components to be created to take advantage of the first layer, for example, to develop applications with graphical user interface or mobile devices.

The ADL makes available on its site a number of packages and tools for implementation of tests in order to ensure conformity between the specified in the SCORM implemented. This set, called the Advanced Distributed Learning (ADL) Sharable Content Object Reference Model (SCORM ®) 2004 3rd Edition Conformance Test Suite, allows testing LMS, SCOs and content packages. Tests of agreement at the LMS level are the most appropriate for verification of CELINE. Such tests of agreement also guided the development of the component,
as they perform an exhaustive battery of consistencies. As the tests were conducted successfully, you can say that the environments that use CELINE are in agreement with the SCORM specification.

4. Environments Developed with CELINE

To evaluate the component, in order to verify the level of difficulty in creating an environment that runs SCORM packages and to see the functioning of the component, there have been developed three environments.

The first one was an Intelligent Tutor System (ITS), developed by [17]. It used a technique known as a overlay model ([18]; [19]), which is the overlapping domain model with the user model. This model uses a structure of directed graph, where each vertex corresponds to a concept to be learned, and the edges are the relations between the concepts. First is defined which of the vertices of the initial graph correspond to the issues that the student must learn. Conditions in the edges are allocated to enable the next summit. These conditions relate to the degree of knowledge that the student must have at the apex of origin, which may be: unknown, partially known, knows, completely known, learned or learned completely. Figure 2 illustrates the domain model used in work of [17].

![Domain model from [17]](image)

Each concept is associated with a set of SCORM packages. When the user imports a package, it needs to be categorized as the content or exercise. The exercise also needs to be classified among three levels of difficulty: easy, medium or hard.

When it is necessary to list the courses that the student can open, the ITS provides a list of concepts that the student is able to learn or practice. In this work, courses are the same as concepts. When the student opens a course, the ITS assembles the structure of the course with content packages and choose the type of
exercises based on the difficulty of the exercise which the student is ready to perform, according to their results in previous exercises, which are valued according to the time spent, the amount of hits and the level of difficulty of the activity performed.

This ITS was not used by students. However, two other environments were developed and used in the classroom: ALICE 2.0 and ASTRO.

In its first version, developed with PHP, ALICE has been used for seven semesters in the disciplines of Algorithms I and II in a course of Bachelor on Computer Science. ALICE is an ITS that incorporates the teacher in the learning process of students, different from what happens in traditional models of ITS. The teacher becomes a partner in the system, providing information about the students and receiving in exchange analyzes the performance of them in the sections of use of the tool. Like the majority of ITS, all content is inserted directly into the tool, without the use of more elaborate structures in content. This environment has been rewritten in Java to use the resources of CELINE, and to be allowed to use SCORM content.

This environment was used in three different classes with a total of 34 students. The SCORM content was built using MOS-Solo authoring tool and the subject was introductory C language. As reported by the teacher, the students were motivated to read the content and implement the proposed activities in the material. No extra instruction was given by the teacher. The source of learning of these students was only the SCORM content.

Because the previous version was used for several semesters, they want reuse consolidated parts, like the exercises base. There were two alternatives: produce them again in a SCORM package, or dynamically create the activities structures by the environment. The second option was chosen because the work to be spent would be lower.

When the course is open, the ITS creates a new activity in the final, called Exercises. With a database of exercises of the previous version, the ALICE selects five issues and add them in the activity Exercises. When one of these issues is viewed by the user, the system builds dynamically the content page following the specification of the duties of an SCO.

The ALICE is being directed to only show the course of C language, and with that he has no resources for administration, as registration of courses and users. The table was populated by users through a script importing such information as the previous version.

The latest environment developed, baptized as ASTRO, was conceived as a flexible system. There are three types of users: administrator, teacher and student. The first is responsible for the registration of teachers and creation of classes; the second by the administration of students and content; and the last by the use of such content.

The teacher adds SCORM packages and links them to some of their subjects. By accessing the system, student is free to register and inform his nickname and password. When entering the system, all disciplines that he can register are listed.
When requesting the registration of a discipline, the student must wait for the release of the teacher. Once released, the student has access to the entire content of the discipline. Figure 3 illustrates the page that lists all courses available with the names of their teachers. Figure 4 presents one of the open courses, developed with the Lab Course.

Figure 3 Page registration to a class.

Figure 4 Page with an opened course.

Access ASTRO is released to the outside, and with that, students could access from their homes or work.

The environment was tested with two classes: the first, taught by one of the authors of that article, contained 31 students, and the second class 15 students.
Both teachers also noted the interest of students to use and interact with the material.

For the development of ASTRO there was produced a document with the use cases and navigability diagram of the application, and two students were allocated, with no experience in web development with Java and even with no knowledge of SCORM. They were given tutorials for implementing web applications and documentation of CELINE. With weekly dedication of 12 hours, the system was delivered in 45 days.

5. Conclusions

From experiments with the environments developed using the component, where two of them were experienced by students, we could see the feasibility of implementing, with Java, Intelligent Learning Environments that manage SCORM packages. We used instructional content made by the tools CourseLab and MOS Solo, confirming the independence of the material to the environment.

The last environment, ASTRO, was fully implemented by two developers with low experience in web development with Java and without knowledge about CELINE component. At the end of implementation, yet they did not have a clear idea of the SCORM specification, but in approximately 500 hours the environment was ready, which demonstrates the easiness in adding SCORM packages resource management in LMS provided by CELINE.

The tests available on the ADL web site to ensure compliance of the system implemented with the specification was used as a guide in the development, and also ensured that the LMS that uses the developed component are in accordance with SCORM specification.

As future work we aim to incorporate more resources for generation of management reports showing the use of content by students, for example, listing the ways of learning, the sequence and time the student accessed each unit. Furthermore, the architecture adopted for the component, dividing into two distinct layers, one for the SCORM specification, and another for the resources to use the LMS allow for future researchers to explore the potential of the specification in environments with graphical interface and mobile devices.

References