Using the Learning Tools Interoperability Framework for LMS Integration in Service Oriented Architectures

José Paulo Leal[1] and Ricardo Queirós[2],

[1] CRACS/INESC-Porto & DCC/FCUP, University of Porto, Portugal
zp@dcc.fc.up.pt

[2] CRACS/INESC-Porto & DI/ESEIG/IPP, Porto, Portugal
ricardo.queiros@eu.ipp.pt

Abstract. The LMS plays an indisputable role in the majority of the eLearning environments. This eLearning system type is often used for presenting, solving and grading simple exercises. However, exercises from complex domains, such as computer programming, require heterogeneous systems such as evaluation engines, learning objects repositories and exercise resolution environments. The coordination of networks of such disparate systems is rather complex. This work presents a standard approach for the coordination of a network of eLearning systems supporting the resolution of exercises. The proposed approach use a pivot component embedded in the LMS with two roles: provide an exercise resolution environment and coordinate the communication between the LMS and other systems exposing their functions as web services. The integration of the pivot component with the LMS relies on the Learning Tools Interoperability. The validation of this approach is made through the integration of the component with LMSs from two vendors.

Keywords: eLearning, Interoperability, SOA, LMS, LTI.

1 Introduction

This work aims to manage and coordinate a network of eLearning systems where students can solve exercises in complex domain such as computer programming. Networks of this kind include systems such as evaluation engines, repositories of learning objects and exercise resolution environments. The Learning Management System (LMS) has also an important role in this network of systems since it is the natural place to assign exercises to students and to collect grades. However, the coordination of a network of such disparate systems is rather complex. Some of these systems expose their functions as web services, such as the repository of learning objects or the evaluation engine, but others have their own web interfaces for students and teachers, such as the LMS and the exercise resolution environment.

The objective of the research described in this paper is to explore the possibility of embedding a pivot component in an LMS that acts as exercise resolution environment and coordinates the communications between the LMS and the web services. The integration of this component with the LMS is supported by the IMS Basic LTI. An integration component developed with LTI can be used in any LMS that supports this standard.

In the remainder of this paper we state the interoperability efforts in the eLearning realm. In the following section we present an architecture for a network of eLearning systems supporting the resolution of programming exercise. Finally we validate the proposed solution by integrating the pivot component in two major LMS - Moodle and Sakai.

2 State of Art

In the last few years there have been initiatives [1] to adapt Service Oriented Architectures (SOA) to eLearning. These initiatives, commonly named eLearning frameworks, had the same goal: to provide flexible learning environments for learners worldwide. Other eLearning interoperability initiatives (e.g. NSDL, POOL, OKI, EduSource, IMS DRI, IMS LTI) appeared in the last decade.

While eLearning frameworks are general approaches for eLearning system integration, several authors proposed service oriented approaches specifically targeted to the LMS. In fact, there are several references in the literature to middleware components for LMSs integration in SOA based eLearning systems. Apostolopoulos [2] proposes a middleware component to address the lack of integration of eLearning services. In this approach the eLearning components are implemented as agents maintained in a local management information base, and can communicate with the agent manager through the SNMP protocol. Costagliola [3] develop an architecture based on a middleware component and use Web Services to integrate different software components and improve interoperability among different systems. The middleware component enables the student learning process traceability since it has been developed to be compliant with SCORM. Al-Smadi [4] presents a service-oriented architecture for a generic and flexible assessment system with cross-domain use cases. All these approaches have in common the need of a modification of LMS for each specific vendor, with the implementation of a new module or building block. To the best of the authors' knowledge there are no references in the literature to the use of a common standards supported by the major LMS vendors as a means to integrate the LMS in a service oriented network of learning environments.

A common interoperability standard that is increasingly supported by major LMS vendors is the IMS Learning Tools Interoperability (IMS LTI) specification. It provides a uniform standards-based extension point in LMS allowing remote tools and content to be integrated into LMSs. The main goal of the LTI is to standardize the process for building links between learning tools and the LMS. The IMS launched also a subset of the full LTI v1.0 specification called IMS Basic LTI. This subset exposes a unidirectional link between the LMS and the application. For instance, there is no provision for accessing run-time services in the LMS and only one security policy is supported [5].

3 Network Architecture

In this section we present the overall architecture of a network of eLearning systems participating in the resolution of programming exercises based on a pivot component embedded in the LMS. Then, we exemplify the integration of these services in a pedagogical learning process. The proposed architecture is described by the UML components diagram shown in Figure 1. The architecture is divided in four components: **Learning Objects Repository (LOR)** to store the exercises and to retrieve those suited to a particular learner profile; **Evaluation Engine (EE)** to automatic evaluate and grade the students attempt to solve the exercises; **Programming Exercises Resolution Environment (PERE)** to recover the exercises descriptions from the repository and provide an exercise resolution environment allowing students to solve the exercises and submit them to the evaluator; **Learning Management System (LMS)** to sequence the presentation of exercises to learners and to collect the students' grades.

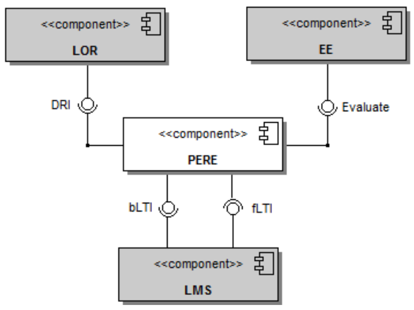


Figure 1: UML components diagram.

The integration of PERE with the other systems relies on the following communications standards: **IMS DRI** to allow the access to the programming exercises stored in repositories, for instance the crimsonHex repository [6], implementing the IMS DRI functions; **Evaluate service** [7] to allow the submission and evaluation of students' attempts to Evaluation Engines implementing the Evaluate service - recently submitted to the e-Framework; **IMS Basic LTI (bLTI)** to embed the PERE in the LMS (e.g. Moodle, Sakai, Blackboard) by assigning information in the launch process such as user identity, course information and role information; **IMS Full LTI (fLTI)** to allow the report of a evaluation result back to the students' LMS grade book.

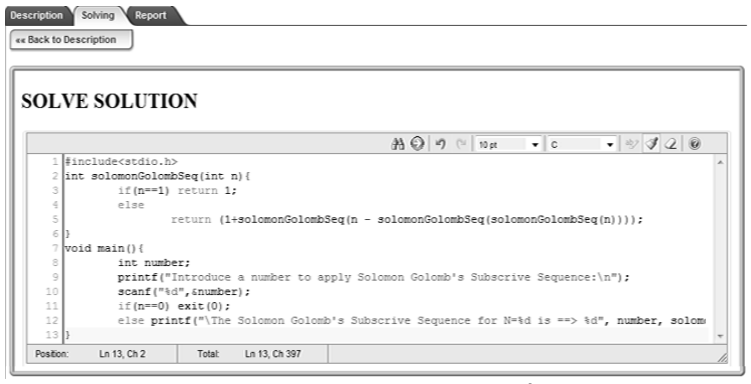
This diagram could be applied to a typical pedagogical learning process such as a classroom assignment in a computer science course. In this scenario, the teacher sets a number of activities (exercises) in the LMS. To select the relevant programming exercises the teacher searches for them in the repository. Then, the learner tries to solve the exercises assigned by the teacher using the PERE (launched by the LMS). The PERE recovers the exercise description from the repository and shows it to the student. After coding the program the student send an attempt to the Evaluation Engine. The student may submit repeatedly, integrating the feedback received from the Evaluation Engine. In the end, the Evaluation Engine sends a grade to the PERE and consequently for the LMS that records it and reports the Learning Object usage data back to the repository. This last task will provide data for future adaptability services that will adjust the presentation order in accordance with the effective difficulty of programming exercises (not the difficulty stated on the LO) and the needs of a particular student.

4 Programming Exercises Resolution Environment

The PERE is a pivot component that is embedded in the LMS and has two main roles: 1) to provide an exercise resolution environment, and 2) to coordinate the communication between the LMS and the other systems (e.g. learning objects repositories and evaluators) exposing their functions as web services.

The PERE component is organised in two main packages: the back-end (used by the teacher) and the front-end (used by the student). In the back-end the teacher configures a work assignment by searching for programming exercises in the repository and associating the most relevant. In the front-end, the student reads the exercise description, solves it in the exercise resolution environment and gets the evaluation report that will help him to refine the exercise and, if necessary, resubmit it. In order to model the front-end three classes were defined: **Description** recovers the exercise description from the repository; **Solver** provides an exercise resolution environment allowing the students to solve the exercise and submit it to the evaluator; **Report** presents the final evaluation report of the students' attempt to solve the exercise.

The graphical interface of the front-end is composed by three panels reflecting the classes previously explained. The next figure depicts the Solver panel.



```
#include<stdio.h>
int solomonGolombSeq(int n){
        if(n==1) return 1;
        else
                return (1+solomonGolombSeq(n - solomonGolombSeq(solomonGolombSeq(n))));
}
void main(){
        int number;
        printf("Introduce a number to apply Solomon Golomb's Subscrive Sequence:\n");
        scanf("%d",&number);
        if(n==0) exit(0);
        else printf("\The Solomon Golomb's Subscrive Sequence for N=%d is ==> %d", number, solom
}
```

Position: Ln 13, Ch 2 Total: Ln 13, Ch 397

Figure 2: The PERE graphical interface.

The integration of the pivot component in the LMS relies on the LTI specification. The basic workflow for using Basic LTI starts when the Teacher (or LMS administrator) adds the tool (PERE) as a Basic LTI tool into their course structure as a resource link using the LMS control panel. The Teacher sets the URL, secret, and key as metadata for the resource link. When the students select the tool, the LMS uses the

URL, secret, and key information to launch the student into the PERE in an iframe or new browser window. The PERE receives a launch request that includes user identity, course information, role information, and the key and signature. The launch information is sent using an HTTP form generated in the user's browser with the Basic LTI data elements in hidden form fields and automatically submitted to the external tool using JavaScript.

5 Integration and Validation

In this section we validate the proposed approach by creating such a network integrated with LMS from two different vendors: Moodle and Sakai.

To recreate the proposed network using Moodle (version 1.9.9) it was necessary to install the XAMPP package that includes the Moodle basic requirements such as the Web server (Apache), the database (MySQL) and PHP. This Moodle distribution needs the further installation of an IMS bLTI consumer.

In relation to Sakai (version 2.7.1) it was necessary to install the Java 1.6 and a Servlet container (Apache Tomcat 5.5). Like the previous case, the Sakai distribution needs the further installation of an IMS bLTI consumer.

After the LMS installation, the PERE tool must be configured as a basic LTI tool in the LMS Control panel. Then, the PERE tool can be included as an activity in the course structure. In the next figure, an activity associated with the PERE tool is defined in a Moodle course.

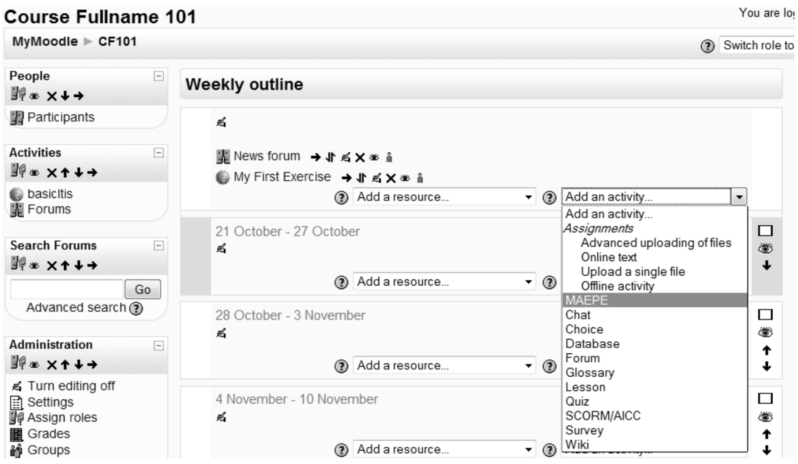


Figure 3: Assign of a PERE activity in a Moodle course.

In both cases it was possible to configure the PERE as an IMS Basic LTI tool in the LMS and allow its standard launch by the students. The main advantage was the fact that there weren't significant differences between both LMS in the configuration and launching of the PERE tool. The main disadvantage was the fact that both LMS still do not support the full LTI. For this reason it was impossible to implement the report of an evaluation of a student's attempt back to the LMS grade book.

6 Conclusions

In this paper we presented an approach to integrate the LMS in a network of services that participates in the automatic evaluation of programming exercises.

We concluded that a pivot component integrated in the LMS is a promising approach to the task of coordinating a heterogeneous network of eLearning systems. The pivot component can have its own user interface for interaction with students, as is required for the resolution environment, that is embed in the LMS user interface. It can also control the invocation of remote web services, such as those exposed by the repository of learning objects and evaluation engine. Finally, it can summarize the activity of the student as a grade and report it back to the grade book of the LMS.

Unfortunately, most LMS vendors, and in particular those we tested, support only the Basic LTI. A full and stable support of LTI in major LMS vendors will encourage us to implement a more sophisticated version of the approach described in this paper. Instead of embedding the resolution environment on the LMS the student should be able to use and Integrated Development Environment (IDE) such as Eclipse. We plan to develop an IDE plug-in to create a programming exercise resolutions environment. It will complement the standard code programming features of an IDE with reading exercise descriptions from the repository, submitting code them to the evaluation engine and displaying feedback to the student. In this future work we will split the coordination task among the pivot component integrated in the LMS and the plug-in on the IDE. Also, the LMS must communicate with a local service on the student's machine (hosted on the IDE) rather than on the cloud.

References

1. Leal, J.P. and Queirós, R.: eLearning Frameworks: a survey. Proceedings of International Technology, Education and Development Conference - Valencia, Spain (2010)
2. Apostolopoulos, T. K., & Kefala, A.: An e-learning service management architecture. In Proceedings of the 3rd IEEE International Conference on Advanced Learning Technologies (pp. 140-144). Athens, Greece (2003)
3. Casella, G., Costagliola, G.,Ferrucci, F., Polese, G., Scanniello, G.: A SCORM Thin Client Architecture for e-learning Systems based on Web Services. In International Journal of Distance Education Technologies, Vol. 5, No. 1, January-March, IDEA Group Publishing, pp.: 13-30 (2007)
4. Al-Smadi & Gutl: SOA-based Architecture for a Generic and Flexible E-assessment System. In EDUCON'10 (2010)
5. IMS Basic Learning Tools Interoperability Specification - Version 1.0 Final Specification, http://www.imsglobal.org/lti/blti/bltiv1p0/ltiBLTIimgv1p0.html
6. Leal, J.P., Queirós, R.: CrimsonHex: a Service Oriented Repository of Specialized Learning Objects. In: ICEIS: 11th International Conference on Enterprise Information Systems, Milan (2009)
7. Leal, J. P. and Queirós, R.: Specifying a programming exercises evaluation service on the e-Framework", in "ICWL - 9th International Conference on Web-based Learning", Springer Lecture Notes in Computer Science (LNCS) (2010)