Universitat Oberta de Catalunya

uoc.edu

Citation for published version

Rius-Gavidia, À. [Àngels], Santanach, F. [Francesc], Almirall Gil, M. [Magí], Sicilia Urbán, M.[Miguel Ángel] & García Barriocanal, E.[Elena] (2011). An Open and Service-Oriented Architecture to Support the Automation of Learning Scenarios. International Journal of Information Technologies and Systems Approach, 4(1), 38-52. doi: 10.4018/JITSA.2011010103

DOI
http://doi.org/10.4018/JITSA.2011010103

Handle
http://hdl.handle.net/10609/149623

Document Version
This is the Accepted Manuscript version.
The version published on the UOC's O2 Repository may differ from the final published version.

Copyright and Reuse
This manuscript version is made available under the terms of the Creative Commons Attribution license (CC BY) http://creativecommons.org/licenses/by/4.0/, which allows others to reuse it for any purpose, as long as they credit you for the original work.

Enquiries
If you believe this document infringes copyright, please contact the UOC's O2 Repository administrators: repositori@uoc.edu

An Open and Service-Oriented Architecture to Support the Automation of Learning Scenarios

Àngels Rius, Francesc Santanach, Magí Almirall
Computer Science, Multimedia and Telecommunication Studies
Open University of Catalonia
Spain
{mriusg, fsantanach, malmirall}uoc.edu

Miguel-Ángel Sicília, Elena García-Barriocanal
Computer Science Department
University of Alcalá
Spain
{msicilia, elena.garciab}@uah.es

Abstract: The specifications of automated learning scenarios could bring about advantages for virtual learning environments and important benefits for organizations, although research in this e-learning area has not addressed this issue yet. In order to achieve this goal, one of the requirements is to have an infrastructure able to support the execution of specifications of learning scenarios. This paper presents an open service-oriented architecture based on the Open Services Interface Definition (OSID) specifications proposed by the Open Knowledge Initiative (OKI) and other normative specifications. Such an architecture is used as a technological infrastructure in a virtual learning environment with more than 40,000 students enrolled and has been tested as the infrastructure of a tool to automate specifications of learning scenarios. A given case study has been used to test the suitability of such an architecture and to sketch out such a tool for the future.

Keywords: SOA architecture, automation, learning scenarios, web services, OKI OSID interfaces

Introduction

Research in e-learning technology with respect to the concept of reusable learning objects and their standardization has evolved in recent years, however, research on the learning scenario concept and its automation has not been so successful.

The automation of the specifications of business processes has been very successful in the Business Process Management (BPM) area and has brought about important benefits for enterprises and organizations. In a similar way, the automation of specifications of learning processes could provide important advantages to the e-learning area and those organizations that focus its activity on the LMS. Some of these advantages are: 1) the establishment of a valid set of primitive learning scenarios for any organization, 2) the creation of a catalogue of learning scenarios that nowadays does not exist yet and the possibility to customize learning scenarios according to an organization's features, procedures and rules, 3) the possible certification of learning processes that permit the implementation of learning scenarios as the first step to future standardization, 4) the optimization and innovation resulting from a deeper knowledge of learning scenarios and finally, 5) the taking on of a given role by the LMS could free a participant of those tasks that can be mechanized, thus providing time for carrying out others of a more added-value nature.

The IMS Learning Design (IMS Global Learning Consortium, 2003a) specification provides recommendations about the learning process design and some tools related to the execution of learning scenarios, which have been developed to test the IMS LD specification. The RELOAD editor (Bolton et al., 2006) to author learning designs in IMS LD format, the CooperCore Design Engine (Vogten, 2004; Vogten & Martens, 2005) to automate the scenario of the delivery of learning activities and finally, the LAMS (LAMS foundation, 2005) to provide teachers with an intuitive visual authoring environment to create sequences of learning activities. In addition, there are other tools like the TELOS editor (Technologies Copigraph Inc.) to validate and design learning scenarios using an ontologydriven and service-oriented architecture, but tools to obtain specifications of automated learning scenarios have yet to exist.

The proposal of this paper is to present an architecture that is able to support a tool capable of automating teachers' tasks that usually occur in a learning environment from its specifications, and not only those related to the learning process design. The technological infrastructure proposed matches with the results of the Campus Project (Open University of Catalonia, 2006) of the Open University of Catalonia (UOC). The UOC is a pioneer virtual university in Spain with more than *43*,000 students enrolled during the academic course 2008-2009. The Campus Project has the aim of evolving the e-Learning Platform created 10 years ago according to an open and service-oriented approach. Due to the characteristics of the UOC architecture and the technological requirements for a tool able to achieve the specifications of automated learning scenarios, a prototype has been tested in the UOC Campus.

This paper is composed of six sections: introduction, the concept of a learning scenario, the requirements of the infrastructure to support the automation of specifications of learning scenarios, the proposal of a concrete architecture, the case study that illustrates the feasibility of the automation of a learning scenario using the proposed architecture and last but not least, the conclusions and future work are presented.

Learning Scenarios

The learning scenario concept used by IMS LD is that given by Fowler in (Fowler, 2000): "A scenario is a sequence of steps describing interactions between the user and the system (...)" in reference to the kind of scenarios considered which are related to the learning process design only.

From the point of view of computer science engineering, a learning scenario is a flexible tool for the system's design that has no single form or way of using it (Toffolon, 2006). This definition allows for the consideration of learning scenarios as a first draft of the system's behavior and sees them as a link between the LMS functionality and the learning processes or between the architecture and its implementation components.

In this paper, a learning scenario is considered as a usual function that occurs in an LMS and its boundaries with the aim of achieving goals related to learning in a broad sense. From a more detailed point of view, a learning scenario must be understood as a sequence of interactions among participants of a learning environment that collaborate to achieve a goal related to learning, whether it is before learning, during learning or after learning. Thus, a learning scenario can be as simple as an interaction or so complex to being described in terms of sequences of reusable learning scenarios. An example of a complex learning scenario could be the preparation of a course or the preparation of the learning activities for such a course; both include other simpler scenarios like the search for materials in a repository or the sending or receiving of messages for instance. In the case study section, the description of the Preparation of Leaning Activities (LA) of a course scenario is presented in Figure 3 and afterwards described in terms of a sequence of reusable learning scenarios.

Infrastructure Requirements

The requirements of the architecture to support the specification of automated learning scenarios are those outlined as follows:

- Useful for any LMS with platform-independence. This goal implies achieving almost real interoperability between all the possible e-learning platforms and the tool. It involves normative descriptions like ADL SCORM (ADL, 2004) and IMS Common Cartridge specifications (IMS Global Learning Consortium, 2008) to provide portability, the LOM standard for defining metadata of learning resources (IEEE LTSC WG12, 2002) or the OSID-OKI specifications (Open Knowledge Initiative, 2004) to define interfaces for component interaction and so on, in order to guarantee the integration of the tools and systems and the exchange of information independently of the medium of transmission. Thus, the infrastructure that supports such a tool must be based on normative descriptions to assure interoperability.
- Fostering the autonomy of the participants involved in the different learning scenarios. The learning scenarios considered a set of interactions in a distributed environment that implement the communication of autonomous components of the system or other systems in the achievement of a given goal require a Service Oriented Architecture (SOA). The SOA is defined as "a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce the desired effects and which are consistent with measurable preconditions and expectations" (MacKenzie et al., 2006). This seems to be the most suitable reference

architecture model for our purposes due to the distributed nature of the learning environment and the required collaboration between components involved in learning scenarios.

- Promoting the construction of new executable specifications of learning scenarios. This purpose requires the existence of some elemental specifications of learning scenarios which must be catalogued and called primitive scenarios in order to use them in other more complex scenarios. Thus, the learning scenarios as components of other learning scenarios must have a defined pattern of behavior that determine their functionality and facilitate their reuse. Therefore, an architecture to support executable descriptions of learning scenarios must foster the reusability of learning scenarios and must permit the broadening of LMS functionalities even for the system's personalization.

To sum up, the required architecture must promote interoperability, the loose coupling between the involved components, the reusability of learning scenarios and the broadening of LMS functionalities.

The Architecture of the UOC Campus Project

The Campus Project comes under the Digital University program fostered by the Catalan Government STSI Department. The Universitat Oberta de Catalunya (UOC) is in charge of coordinating and leading the project, which is carried out using the knowledge and experience of each associated university. Each member, therefore, contributes tools and resources to the project, which is organized according to a development community in open source.

The aim of the Campus Project is to develop a technological infrastructure with open source tools to provide online training. The project requirements are: 1) open code and open standards, 2) user-centered design, 3) interoperability between tools and with other systems, 4) scalability of the solution, 5) high concurrence of users and processes, 6) OKI OSIDs specifications (Open Knowledge Initiative, 2004) as a mechanism of interoperability, which can be executed and integrated into Moodle's and Sakai's platforms and is founded on a service-based architecture.

Furthermore, such an architecture is designed to pursue a two-fold objective: 1) to share its e-learning tools with other institutions and 2) to integrate and use e-learning tools from other institutions at the UOC. Therefore, the design of the architecture starts from the assumption that real interoperability relies on adopting a service-oriented approach in order to promote the interoperability of the system.

The Campus project's architecture follows a three-layered architectural model which the IMS-DRI (Digital Repository Interoperability) (IMS Global Learning Consortium, 2003a) specifications recommend as can be seen in Figure 1. On the top layer, the modules corresponding to the tools and applications that extend the LMS functionalities are shown and on the bottom layer, the e-learning platform as the base of such an infrastructure. And just in the middle, the intermediate layer which acts as a bridge between the modules and the e-learning platform to extend the LMS functionalities of the system.

Figure 1: The three-layered architecture of the UOC Campus

The desired interoperability has been considered during all the stages of the Campus project development. First of all, Moodle and Sakai, two open platforms have been included in the architectural model instead of a unique platform as can be seen in Figure 2.

Secondly, the two selected e-learning platforms have evolved in frameworks in order to solve some problems of integration with third party tools. A framework is the development of a platform which offers a set of APIs; mechanisms to incorporate plug-ins and other elements that enhance both the extensibility and personalization of the platform. This approach facilitates a loose coupling degree but is still not enough to achieve fine interoperability between systems and tools.

Thirdly, the service-oriented architectural model was adopted in order to go beyond the achieved interoperability. In SOA, the system is modeled around a set of modules with a public functionality and responsibility and a set of mechanisms that allow interaction between services (MacKenzie et al., 2006). If the services implement a very clear-cut interface, then it is possible to isolate the interaction mechanisms in a unique layer facilitating the control of loose coupling across the systems. It has been achieved through OKI, using the OKI Bus as part of the middleware. The mission of this OKI Bus is to solve all the problems related to the communication protocols, remote communication between applications, performance optimization measures, increasing communication quality and so on.

As each tool has its own architecture and the most appropriate technology to solve its business logic, an OKI Bus and an OKI gateway are needed in order to facilitate the communication between tools and platforms. The OKI gateway is an adapter that translates the requests of the basic services used by tools into calls to the specific API of each platform. Each platform has its own gateway, so in order to integrate a new platform; a new gateway must be used.

Figure 2: Three-layered architecture using OKI in the middleware

This architecture is based on the OKI proposal and the best way to see it is to think about a system of blocks or pieces that fit together as shown in Figure 2. Each piece is a black box that performs an activity within its limits and invisible to the rest. Consequently, each tool has its own internal architecture and the most appropriate technology to resolve its business logic.

As the implementation is based on a service-oriented architecture (SOA) the tools interact with the base platform using a set of basic services The criteria for deciding what these basic services should be are as follows: 1) the minimum, 2) they should be defined by OKI, 3) those that are indispensable for the system to work (authentication and authorization) an 4) those that allow the system to be administered and managed as though it were a single product (logging, locale, configuration and messaging). Thus, the tools developed communicate with the base platform using a maximum of six services:

- The authentication service permits a new user to be registered in a system or to find out if a user is connected to it. It is an indispensable service in any computer program with user registration.
- The authorization service permits it to be known if a user is authorized to access to a function or resource. It is indispensable in any system where the users perform different roles.
- The logging service allows data on the activity carried out by the program to be stored. It is very useful for finding out what is happening in a system and how it is working.
- The locale service permits the language of a program to be changed or new languages to be added.
- The configuration service permits the configuration of the parameters of a computer application to be created or changed.
- The messaging service permits communication among users within the system. It is an indispensable service in any system based on a distributed architecture.

From all of this, it is easy to realize that the OKI proposal plays an important role in the architecture of the UOC Campus. Furthermore, the OKI-OSID specification is essential for the automation of specifications of learning scenarios since the definition of recommended web service interfaces are addressed specially to the higher education community which fosters interoperability, integration and is suitable for the SOA architectural model as will be seen in the following sections.

The case study

This section relates how the suitability of the proposed architecture as an infrastructure to support the automation is tested using a specific learning scenario that occurs frequently in every course in the UOC Campus. First of all, the specific learning scenario is described and secondly, some considerations regarding its implementation using OKI service interfaces are presented.

Learning scenario description

This case study scenario consists in preparing learning activities (LA) for a given course for the Computer Science, Multimedia and Telecommunication Departments in the UOC. In Figure 3, the description of such a learning scenario in terms of a sequence of reusable learning scenarios is depicted using a BPMN diagram according to the BPMN (Business Process Modeling Notation Specification) (Object Management group, 2006) As can be seen, the two pools correspond to both participants in this scenario: 1) the LMS assuming the role of the coordinator instructor and 2) the consultant as the teacher being able to interact with students during the course. For those who do not know the UOC learning methodology and the learning organization figures associated to such a pedagogical model, it would be recommended to read (Rodríguez et al., 2006) before going on.

Figure 3: The BPM diagram of the Preparation of Leaning Activities (LA) of a course

The preparation of learning activities for a given course is shared by usually several virtual classes at the UOC Campus. Each group of students enrolled in each virtual class has its own consultant, who is responsible for solving doubts of the students assigned to the group and who guides the learning process. Coordinating the consultancy tasks and taking care of the learning quality is the responsibility of the teacher who is a member of the academic staff that is ultimately responsible for the subject matter in question. In this case, we are interested in those tasks related to the preparation of activities of the course: distribution of tasks, their assignation and validation, among others. As a result of the execution of this learning scenario, all of the activities of the course are going to be saved in a repository with its established format and structure for future use.

The course has four theoretical learning activities and two practical learning activities. Each of them is formed by a set of exercises with a specific learning objective and contributes to the final result of the learning activity where it is included. The LA template is a learning resource that describes the structural content of each LA, its learning objectives and the competences that it develops; the number of exercises that are composed in the LA activity, their type, objectives and their assessment criteria. This information must be annotated using the LOM metadata. Each learning activity has its own solution which is prepared by a consultant in order to estimate the time needed by students to solve it and to check that all the information required is well provided.

The teacher must distribute the tasks related to the preparation of LAs, so a distribution tasks' template is needed. This template describes the tasks to be carried out, the roles that they perform, the condition for their validation, the testing points of the task and the procedure required to carry them out.

The consultants design new exercises, including their solution. When the deadline reaches, they send their proposal to be verified. If it is not validated, then a list of corrections is retuned to the consultant creator in order to improve his/her proposal. When all the exercises of the LA proposal are validated, the next step is the integration of them. It must be repeated for each LA of the course. The learning scenario finishes when all the LA of such a course is prepared and saved in the LA repository.

As it can be seen in this case study, the teacher role in the preparation of the learning activities of the course is done by the system when possible.

Learning scenario implementation

As interoperability and reusability are key factors and a requirement for the tool that automate the specification of learning scenarios, the OKI service interface definition specification and the recognition of behavioral patterns have been used in this prototype.

On the one hand, the achievement of interoperability is possible due to the OKI-OSID specification. Such web service interfaces have been very useful in order to automate the mapping from the participant interactions to calls to web services due to the SOA architecture of the e-learning platform.

On the other hand, the reusability of leaning scenarios promotes the description of a learning scenario in terms of sequences of reusable learning scenarios, which are defined once and reused many times. In this case, the Preparation of the learning activities' scenario is defined as a sequence of the following reusable learning scenarios: Searching_LA_template, Searching_distribution_tasks_templates, Assigning_tasks_to_consultants, Preparing_exercises_and_solutions, Validating_tasks_performed, Integrating_exercises_and solutions_into_LAs and Saving_LAs.

Due to the limited extension of this paper, it is not possible to present a detailed analysis of the OKI-OSID services used in the implementation of the scenario. Instead of it, a brief description of each reusable learning scenario is done in terms of the OKI service interfaces used.

The OKI services used in this implementation are those referred in the previous section (authentication, authorization, logging, locale and user messaging) in addition to the repository service. The reason is that learning activities and the templates used in the Preparation of learning activities' scenario are considered to be stored in a

Dspace repository and described with IEEE LOM metadata. Other OKI services like Scheduling Service and Assessment Service will be added to the architecture in the future to achieve a more automated and easy-to-use tool.

Before presenting tables 1 to 7 to describe the OKI services used in each reusable learning scenario, it is important to point out common OKI services in all of them. This part refers to the **Agent Service** to take user profile information like the preferred language of the user, the **Logging Service** to notify tracking and monitoring information, the **Locale Service** to configure the user interface using the user's preferred language and the **Id Service** to get and to create unique identifiers

Searching_LAs_template
A user with the **Teacher** role executes this application functionality. 1. Verify if the user is authenticated: **Authentication Service**. 2. Verify if the user is granted permission to execute this application functionality: **Authorization Service**. 3. The user fills in a search form. The form's fields are the metadata used to describe the Learning Activity templates. The form is generated using the RecordStructure and PartStructure OKI OSIDs. **Repository Service**. 4. The system searches all the assets that its metadata matches with the search form fields. **Repository Service**. 5. A result form is showed to the user and the user selects the appropriate learning activity template. 6. The selected template is returned.

Table 1: The OKI services used in the Searching_LAs_template process

Searching_Distribution_Tasks_Template
A user with the **Teacher** role executes this application functionality. 1. Verify if the user is granted permission to execute this application functionality: **Authorization Service**. 2. The user fills in a search form. The form's fields are the metadata used to describe the distribution task templates for the consultancy of the course. The form is generated using the RecordStructure and PartStructure OKI OSIDs. **Repository Service**. 3. The system searches all the assets that its metadata matches with the search form fields. **Repository Service**. 4. A result form is showed to the user and the user selects the appropriate distribution task's template. 5. The selected template is returned.

Table 2: The OKI services used in the Searching_Distribution_Tasks_template process

Assigning_Tasks_To_Consultants
A user with the Teacher role executes this application functionality. 1. Verify if the user is granted permission to execute this application functionality: Authorization Service. 2. The user introduces the planning schedule of the subject and the system returns all the consultants that can be course instructors of this subject. **Agent and Group Service**. 3. The subject is a Group and its members are the different courses of the subject. The course is also a Group and has its own members, with different roles. 4. The user sends a message to each consultant describing the task to be done according to the learning activity template and the distribution task's template. Messaging/e-mail Service.

Table 3: The OKI services used in the Assigning_Tasks_To_Consultant process

Preparing_Exercises_And_Solutions
A user with the **Consultant** role executes this application functionality. 1. Verify if the user is authenticated: **Authentication Service**. 2. Verify if the user is granted permission to execute this application functionality: **Authorization Service**. 3. The user reads the message sent by the Teacher and creates the exercises and the solutions according to the

message indications.
4. The user sends the exercises and the solutions indicating the corresponding subject and the semester. **Messaging/e-mail Service.**
5. The systems send a message to the Teacher notifying that the consultant task is done. **Messaging/e-mail Service**.

Table 4: The OKI services used in the Preparing_Exercises_And_Solutions process

Validating_Tasks_Performed
A user with the **Teacher** role executes this application functionality. 1. Verify if the user is granted permission to execute this application functionality: **Authorization Service**. 2. The user validates the exercises and solutions sent by the consultants. 3. The user can send a message to a specific consultant to request their assignments or can send a message asking questions. **Messaging/e-mail Service.** 4. If the user rejects a consultant's assignment, then a message asking for a new assignment is sent. **Messaging/e-mail Service.** 5. Once consultants' assignments are accepted by the Teacher, then the **Integrating_Exercises_Into_LAs** process is called using the list of exercises and solutions as input parameters.

Table 5: The OKI services used in the Validating_Tasks_Performed process

Integrating_Exercises_Into_LAs
A user with the **Teacher** role executes this application functionality. 1. Verify if the user is granted permission to execute this application functionality: **Authorization Service**. 2. The user takes the list of exercises and solutions and creates learning activities according to the learning activity template and the distribution task's template. 3. The system calls the **saving_LAs** process using the Learning Activities list as an input parameter.

Table 6: The OKI services used in the Integrating_Exercises_Into_LA process

Saving_LAs

A user with the **Teacher** role executes this application functionality.
1. Verify if the user is granted permission to execute this application functionality: **Authorization Service**.
2. For each learning activity in the list:
    1. The user fills in a saving form. The form's fields are the metadata used to describe the Learning Activities. The form is generated using the RecordStructure and PartStructure OKI OSIDs. **Repository Service**.
    2. AN OKI Asset is created using the learning activity as content and the form's metadata as an OKI OSIDs Records and Parts. **Repository Service**.
    3. The new Asset is validated before being saved in the repository. **Repository Service**.
3. All the learning activities are in the repository described with metadata, available to be used by other processes like the course scheduling for the students and others.

Table 7: The OKI services used in the Saving_LAs process

It is interesting to note that at the beginning of the process, it is vital to check that the participant that is going to execute an operation has been granted permission. This verification is required for each process implementing a new primitive scenario and consequently this part of the code could be considered as a reusable scenario given its transversal manner. Therefore, an authorization role of each participant is needed, so the Authorization service and the Authentication service will be used at the beginning of each learning scenario, independently of the learning scenario complexity.

In order to show a part of the implementation code, two typical processes related to the repository have been selected: The Searching_LAs_Template and the Saving_LAs_Template. They are so generic to represent the use of OKI web service interfaces related to any repository, independently of the asset to search or save.

```
private void searching_LAs_template() {
    try {
      if (authZ.isUserAuthorized(idC.getId("ActivityPlanification"), idC.getId("instance"))) {
         log.trace("searching_LAs_template");

         //step2
         org.osid.repository.Repository repository = repoComp.getRepository(idC.getId("LAs_template_repository"));
         log.trace("Repository: " + repository.getDisplayName());
         // This implementation has only one RecordStructure for Type
         RecordStructure rs = (repository.getRecordStructuresByType(this.LOM_STRUCTURE_TYPE)).nextRecordStructure();
                  // create a form using record structure
                  LOMForm form = new LOMForm(rs);
                  form.setTitle(locale.getEntry("LOM_FORM_TITLE") + ". " + repository.getDisplayName());
                  form.setHelp(locale.getEntry("LOM_FORM_HELP"));
                  // show the LOM form
                  log.trace("LOM Metadata search form.");
                  if (form.show()) {
                         // repository search
                           log.trace("search by LOM Metadata.");
                           AssetIterator assetIterator =
                  repository.getAssetsBySearch(form.getLOMAsset(),this.SEARCH_BY_LOM_TYPE,new SharedProperties());
                           // show the assets - result screen
                           AssetListForm listForm = new AssetListForm(assetIterator);
                           listForm.setTitle(locale.getEntry("LIST_FORM_TITLE") + ". "+ repository.getDisplayName());
                           listForm.setHelp(locale.getEntry("LIST_FORM_HELP"));
                           log.trace("Asset List Form.");
                           if(listForm.show()) {
                                    // take the selected Asset.
                                    log.trace("LAs Template selected.");
                                    this.LAsTemplateDocument = listForm.getSelectedAsset().getContent();
                                    }
                           }
                  }
    } catch (Throwable t) {
         t.printStackTrace();
    }
}
```

Figure 4: The implementation of Searching_LAs_template process using OKI services

In the code presented in Figure 4, it is possible to observe that the Authorization Service is the first OSID Service needed to check that the teacher grants permission to search the templates related to the LA that is to be created. Then, once the search condition is formulated, the system presents a form to be filled in with all that information so that the corresponding metadata could search those templates and return them. Thus, it implies the need to use the Repository Service and use metadata according to the LOM standard.

Furthermore, in the Saving_LAs process, after using the Authorization Service, the LA is described by means of its metadata and such metadata and the LA together make up an asset which is saved in the repository as can be seen in Figure 5.

```
private void saving_LAs() {
    try {
      if (authZ.isUserAuthorized(idC.getId("ActivityPlanification"), idC.getId("instance"))) {
```

```
        log.trace("saving_LAs");

        //step8
        org.osid.repository.Repository repository = repoComp.getRepository(idC.getId("LAs_repository"));
        log.trace("Repository: " + repository.getDisplayName());
        // This implementation has only one RecordStructure for Type
        RecordStructure rs =
                        (repository.getRecordStructuresByType(this.LOM_STRUCTURE_TYPE)).nextRecordStructure();
        // save all the learning activities into the repository
        while (learningActivities.hasNext()) {
                java.io.Serializable document = learningActivities.next();
                // create a form using record structure
                        LOMForm form = new LOMForm(rs);
                        form.setTitle(locale.getEntry("LEARNING_ACTIVITY_METADATA_FORM_TITLE") + ". " +
                                repository.getDisplayName());
                        form.setHelp(locale.getEntry("LEARNING_ACTIVITY_METADATA_HELP"));
                        form.addDocumentLink(locale.getEntry("LEARNING_ACTIVITY_TO_DESCRIVE"), document);
                        // show the LOM form
                        log.trace("Learning Activity Metadata form.");
                        if (form.show()) {
                                // repository save
                                log.trace("Save the new Asset.");
                                Asset newLA = repository.createAsset(form.getDisplayName(),
form.getDescription(), this.LOM_ASSET_TYPE);
                                newLA.updateContent(document);
                                newLA.copyRecordStructure(form.getLOMAsset().getId(), rs.getId());
                                // set the records.
                                if(repository.validateAsset(newLA.getId())) {
                                        log.trace("Asset saved.");
                                        }
                                }
                        }
                }
    } catch (Throwable t) {
        t.printStackTrace();
    }
}
```

Figure 6: The implementation of Saving_LAs process using OKI services

Conclusion and future work

This paper proposes an architecture for the future construction of a tool for automating specifications of learning scenarios. It focuses on the required architecture to support it as well as its implementation through OKI service interfaces. The OKI proposal seems suitable as an interface mechanism to implement interaction between the different components that take part in the scenario within a distributed environment and furthermore, it provides interoperability and loose coupling thereby satisfying the tools' requirements.

Such a test has been done in the UOC learning-environment, so the architecture of the Campus has been proved as suitable as a technological infrastructure for a tool able to support the automation of a learning scenario based on the mapping of the primitive scenarios to calls to OSID services. A more automated and friendly tool could be possible by adding other services and standards to the LMS. Scheduling, workflow and other services, as well as QTI and SCORM specifications for the exercises can be very useful for this purpose.

Therefore, as part of our future work and in order to solve the lack of automated mapping, there is the need to develop an ontology to specify learning scenarios formally and obtain from them the calls to OSID services that could execute such specifications.

References

ADL (2004). *Sharable Content Object Reference Model Version 1.3.1.* Advanced Distributed Learning. Retrieved 28/01/2007 from http://www.adlnet.org

Bolton, JISC , Strathclyde. (2006) RELOAD. Reusable eLearning Object Authoring & Delivery. Retrieved 19/10/2009 from the Web Site of the RELOAD Project: http://www.reload.ac.uk/

Fowler, M. (2000).*UML Distilled: A Brief Guide to the Standard Object Modeling Language (2ond Edition).* Addison-Wesley. pp. 39-40

IEEE LTSC WG12. (2002). *Draft standard for learning object metadata.* IEEE Learning Technology Committee. Retrieved 25/10/2007 from http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf

IMS Global Learning Consortium (2003a). *IMS digital repositories interoperability-core functions information model Version 1*. Final Specification. IMS Global Consortium. Retrieved 19/10/2009 from http://www.imsglobal.org/digitalrepositories/driv1p0/imsdri_infov1p0.html

IMS Global Learning Consortium (2003b). *IMS Learning Design Information Model. Version 1.0 Final Specification.* Retrieved 19/10/2009 from http://www.imsglobal.org/learningdesign/ldv1p0/imsld_infov1p0.html (Last update 2003).

IMS Global Learning Consortium (2008). *IMS Common Cartridge Specification.* Version 1.0 Final Retrieved 19/10/2009 from http://www.imsglobal.org/cc/

LAMS foundation (n.d.). Learning Activities Management System (2005). Retrieved 19/10/2009 from http://www.lamsfoundation.org/

MacKenzie, C., Laskey, K., McCabe, F., Brown, P. & Metz, R.(2006) *OASIS reference model for service oriented architecture v.1.0. Technical report, OASIS Official Committee Specification*.

Object Management Group (2006). Business Process Modeling Notation Specification. Final adopted specification.retrieved 16/10/2009 from http://www.bpmn.org/Documents/OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf

Open Knowledge Inititative (2004). *Open Service Interface Definitions.* Version 2.0.0. Retrieved from 20/09/2009 from http://www.okiproject.org/view/html/site/oki/node/1478

Open University of Catalonia (2006). The Campus Project. Retrieved 19/10/2009 from http://www.campusproject.org/en/index.php

Rodíguez, M. E.; Serra, M.; Cabot, J.; Guitart, I. (2006). *Evolution of the Teacher Roles and Figures in E-learning Environments* . In: , . The 6th IEEE International Conference on Advanced Learning Technologies. IEEE Computer Society. Page. 512- 514. ISBN: 0-7695-2632-4.

Rodríguez, M. E.; Serra, M.; Cabot, J.; Guitart, I. (2006). *Evolution of the Teacher Roles and Figures in E-learning Environments.* In: , . The 6th IEEE International Conference on Advanced Learning Technologies. IEEE Computer Society. Page. 512- 514. ISBN: 0-7695-2632-4.

Technologies Copigraph Inc. (n.d.). *IMS-LD Scenario editors*. Retrieved 19/10/2009 from http://www.cogigraph.com/Produits/IMSLDScenarioEditors/tabid/1099/language/en-US/Default.aspx

Toffolon C. (2006). Learning management system scenario-based engineering. *Proceedings of the 5th European Conference on e-Learning (ECEL 2006),* Winchester, UK. 397-406.

Vogten, H. (2004). CopperCore, an Open Source IMS Learning Design Engine. Retrieved 19/10/2009 from http://hdl.handle.net/1820/286

Vogten, H, Martens, H. (2005) . CopperCore 3.0. Retrieved 19/10/2009 from http:// www.coppercore.org

Acknowledgements

This work has been partially supported by the Spanish Government Project PERSONAL (ONTO) ref TIN2006-15107-C02 and by the Spanish MCYT and the FEDER funds ref. TSI-020301-2008-9 (SUMA).