13

Developing Distributed Repositories of Learning Objects

Salvador Otón, Antonio Ortiz, Luis de-Marcos, Sergio Mazo de Dios,
Antonio García, Eva García, José R. Hilera and Roberto Barchino
Computer Science Department, University of Alcala,
Technical School of Computer Science Engineering,
Alcalá de Henares, Madrid
Spain

1. Introduction

A repository or digital storage of educative elements is a collection of resources (learning objects) accessible through a communication network. It is not necessary to have a previous knowledge about the collection containing the structure of the resources or only the metadata describing them, together with a reference to locate it (IMS DRI, 2003).

The aim of a repository is to facilitate the reusability of educational resources, providing access to the stored resources to learning management systems (LMS); learning content management system (LCMS); content portals (for an instance: searching systems of digital libraries, World Wide Web searching, etc.); or any application/software developed to access to learning objects.

Digital repositories, in the broadest sense, are used to store any sort of digital material. However, digital repositories for learning objects are much more complex in terms of what to store and how to do it. The purpose of a digital learning object repository is not just to store and distribute learning objects, but to allow them to be shared by different users and, above all, to make it easier to reuse them in different training activities (figure 1). From the users' point of view, these repositories have the advantage of having access to the content stored in them. To make this possible, the content must be gathered through certain procedures, rules and standards whose implementation is intended to promote the reusability of learning objects. Moreover, the repository itself must follow a series of specifications and standards which enable to browse the content it stores and facilitate interoperability with other repositories.

Most repositories are usually autonomous, that is, they work as portals that can be accessed through a Web-based interface, providing a search mechanism and a list of categories to conduct the search. However, it is becoming increasingly common the possibility of making federated searches in distributed repositories from the original repository.

Some examples of this kind of search can be found in repositories such as those set out in table 1. In this way, it is possible to access to different repositories from the learning management platform, preview their contents and even download and incorporate them into a course.

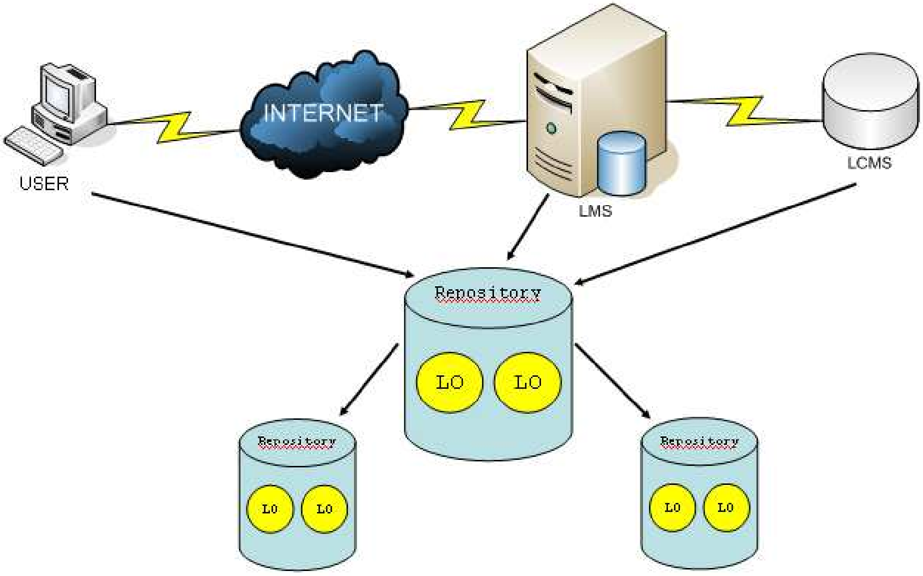


Fig. 1. Learning Objects Repositories.

REPOSITORY	*URL*
Agrega	*http://contenidos.proyectoagrega.es*
Ariadne	*http://www.ariadne-eu.org*
EdNA Online	*http://www.edna.edu.au*
EducaNext	*http://www.educanext.org*
LORNET	*http://www.lornet.org*
MACE	*http://portal.mace-project.eu/*
Merlot	*http://www.merlot.org/*
OER Commons	*http://www.oercommons.org*

Table 1. Public Learning objects repositories.

Therefore, it is now essential that repositories are developed according to standards and specifications that guarantee interoperability with any other repository or search system. In this chapter we present (1) the most important standards and specifications in this area, (2) a service-oriented architecture of a federated repository based on these standards and specifications, and (3) how to develop a federated repository using this SOA architecture. The chapter is structured as follows: section 2 presents the main specifications and standards in the scope of e-learning with an emphasis on those related to construction and management of learning object repositories. Section 3 presents the design of a service-oriented architecture for building a distributed repository of learning objects. Section 4 details how to develop a federated repository using SOAP Web services while section 5 details how to develop a federated repository using RESTful Web services. Finally, section 6 presents our conclusions.

2. Standards and specifications

Nowadays, one of the main problems with e-learning systems is the accessibility, interoperability, durability and reusability of the educational materials. It is necessary that the platforms of e-learning and the author tools are based on standards and specifications and, in addition, all the learning objects are "described" using the same "language". Moreover, it is sure that a learning object will be re-usable, if it is created according to the definition of some standard, and so, the object content must be described through metadata. However, the standards do not provide any guideline of how a learning object can be discovered.

Therefore, the educational content should be:

- Developed as standards.
- Universally discovered.
- Easy to find.
- Independent of metadata format.
- Reusable.
- Independent of the storage platform.
- Integrated in other learning systems.

E-learning systems have made satisfactory progress towards communication through the Internet, either to communicate with users or to intercommunicate with others systems. The main feature of the new e-learning environment is the use of the Web as a single distribution platform. Therefore, e-learning systems provide features which were unknown in the traditional education system, making it possible to solve some deficiencies as the high production and development time costs. However, there is no reusability of the learning objects and software components that compose an e-learning tool.

These problems are summarized by (Koper, 2000) on the following points:

- The increase of the heterogeneous products and the interaction between people and systems, only between people or only between systems.
- The spectacular increase of available information and its dispersion in different systems and applications, which implies the need to communicate the different software products and platforms.
- The distributed e-learning process organization, due to geographic dispersion of the course members.

Therefore, there are different solutions to solve these problems; solutions that try to unify the educational way of creation and the way of integrating platforms and educational repositories, and also the communication of those contents. The solution to these problems is called interoperability.

The IEEE defines interoperability as the ability of two or more systems or components to exchange information and to use the information exchanged. Within the world of e-learning exists a set of standards and specifications (hereinafter “norms”) to ensure interoperability between different learning systems and more specifically those related with learning object repositories (Otón S. et al, 2009). These norms enable the exchange of the teaching contents they store and consequently achieve the reuse of those contents in different training projects. These norms may be classified as follows:

1. Norms geared towards building and defining the learning object itself, that is to say, its content and metadata.

2. Norms geared towards the publication and search for learning objects by making it easier to locate resources in different repositories.
3. Norms designed to assist in the design of repositories whose aim is interoperability and which therefore specify software architecture for their construction.

The majority of the norms belonging to the second and third groups norms are based on Web services and service-oriented architecture (SOA) (W3C, 2004). Web services constitute a reusability mechanism of distributed software components. They can be registered and published in the Web, and they make use of open and standard protocols of Internet, like HTTP, XML, UDDI, SOAP, WSDL and REST, that deal with the interoperability problem among the different technologies and software platforms.

The first group of norms are aimed at the generation, documentation and packaging of learning objects. Its main feature is the description of the object using metadata. The main norm to comply in the description of the metadata of a learning object is LOM (IEEE, 2002). With respect to packaging, we outline the two most used norms today that are SCORM (ADL, 2002) and IMS Common Cartridge (IMS CC, 2008) belonging to a broader standard called Digital Learning Services Standards.

The IMS Common Cartridge (CC) specification proposes a standard way of packaging learning objects and its metadata (making use of the LOM specification). This standard describes the structure of a package, disposing learning resources as a folder tree. Assessments, Question Banks, Discussion Topics, Web content, and many others can be packaged inside a CC cartridge.

CC considers four categories of resources inside a package:

- imsmanifest.xml: A file describing the content of the package. It must be placed at the root directory in the package.
- Web Content: A folder tree which contains digital content resources (Web pages, documents, media files, etc.), Web links and links to other resources inside the package. All resources inside this category must be placed at the Web Content root folder or at one of its subfolders. Only one single Web Content folder can exist inside a package.
- Learning Application Object (LAO): A folder tree with the files (or file references) needed for a learning system to distribute learning object contents. There can be multiple LAOs folders inside a package and each one can contain subfolders.
- Folder: A package may contain folders to group other kinds of resources not considered in the categories defined above.

References between the resources packaged in a cartridge can exist with the following restrictions:

- A Web content resource can reference any other Web content resource, but it can’t reference resources outside the Web content folder.
- A LAO resource can reference Web content resources and any other LAO resource inside its folder tree, but it can’t reference LAOs outside its folder tree.

One of the basic pillars of interoperability between learning object repositories is the ability to search their contents. Recently, search systems have evolved from only working in one repository to working simultaneously in various distributed repositories; this is known as “federated search”. In this kind of search is widespread to use of Web services, so that these services act as intermediaries between different learning objects repository. A proof of this is found in the specification SQI (CEN, 2005). Examples of repositories that implement federated search through SQI can be found in Merlot, Ariadne or GLOBE.

SQI was defined by the CEN (European Committee for Standardization). It forms part of a public initiative known as the CEN/ISSS Learning Technologies Workshop, whose commitment it is to guarantee interoperability between learning object repositories. SQI uses XML as the language for receiving information requests and for returning the results.

SQI specification consists in a definition of a set of methods that a repository should provide, so that remote systems (clients) can query for learning objects stored within the repository. Figure 2 shows how repository A (origin) makes a data request from repository B (destination). For this communication to be possible, it is necessary to use a common query language (based on SQI) which both repositories understand. However, the internal query language of each repository may differ. In this case, a layer (an SQI component) is responsible for making the necessary conversions.

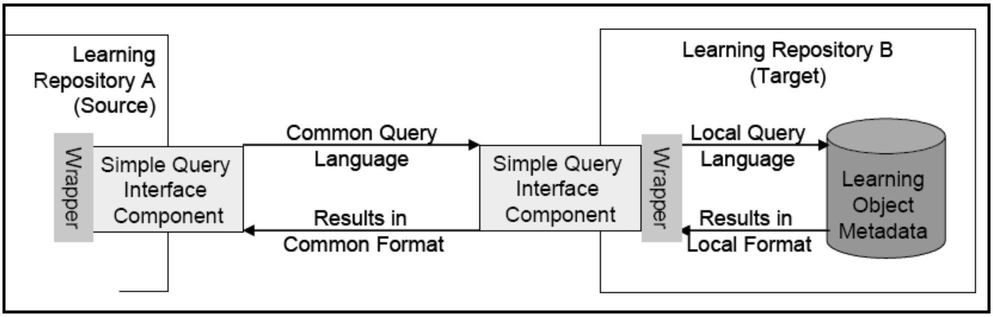


Fig. 2. Communication between two SQI repositories.

SQI identifies thirteen methods to be provided by such systems, these methods are embedded in a Web service that is exposed by the target repository and defined in WSDL files. They are classified in four categories: configuration methods, session management methods and query methods (synchronous and asynchronous).

SPI (Simple Publishing Interface) specification (CEN, 2010), also devised by the CEN, is a protocol for publishing digital objects or their metadata in repositories. It provides a simple protocol which is easy to implement and integrate in already existing systems. His objective is to develop a practical approach towards interoperability between repositories.

SPI makes a distinction between semantic and syntactic interoperability. Syntactic interoperability is the ability of applications to deal with the structure and format of data (for example XML documents). Semantic interoperability refers to the ability of two parties to agree on the meaning of data or methods. When exchanging data, semantic interoperability is achieved when data is interpreted in the same way by all the applications involved.

In a typical SPI scenario, two approaches allow for passing data from a source to a target: “by value” or “by reference”. “By value” publishing embeds a learning object, after encoding, into the message that is sent to a target. “By reference” publishing embeds a reference (e.g., a URL) to a learning object to publish into the message that is sent to a target.

The model for SPI builds on a separation between data and metadata, the data is a resource (e.g., a learning object) and the metadata is the description of the resource (e.g., using LOM to describe the learning object) (figure 3). Every resource can be described by zero, one, or more metadata instances. A metadata instance must have a metadata identifier that identifies the metadata instance itself, and must have a resource identifier that is equal to the identifier of the resource. The metadata identifier enables distinguishing between multiple metadata instances referring to the same resource. In this model, a metadata instance must be connected to a resource; however the resource may be hosted externally.

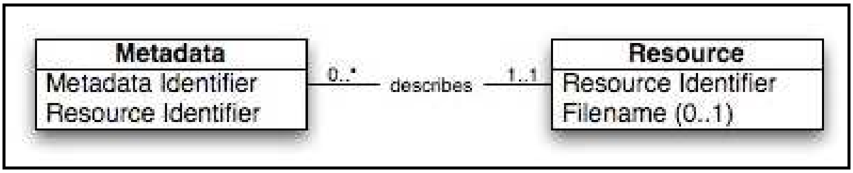


Fig. 3. Resource and Metadata instances in SPI.

The SPI model defines several classes of messages and functional units in a publishing architecture. When binding the specification to a given technology, these concepts are mapped into a concrete specification that can be implemented in a repository and for which conformance can be tested.

SPI defines the following methods:

- Submit/Delete Metadata Record: These are methods for inserting or deleting object descriptions respectively.
- Submit/Delete Resource by value / by reference: These are methods for inserting or deleting resources respectively.
- The SPI model does not include explicit methods for updating resources or metadata instances.

The IMS has also specified a set of services addressed to exchange of information that describes people, groups, memberships, courses and outcomes within the context of learning. That is the IMS Learning Information Services (LIS) specification (IMS LIS, 2010). LIS consists of six services that can either be used individually or in various combinations.

To achieve interoperability across Service Oriented Architectures, IMS has developed the General Web Services (GWS) specification (IMS GWS, 2005). The specification is formed by different profiles focused on addressing, security, attachments and a base profile describing most common problems when Web Services are used. The specification offers solutions to improve those considered by WS-I specifications. LIS is aimed at classic Web Services, i.e., SOAP based Web Services. Other kinds of Web Services such as RESTful Web Services are out of scope.

There is a necessity to create, manage, organize and exchange resource lists (collections of resources and their metadata instances) in distributed learning architectures. IMS Resource Lists Interoperability (RLI) exposes a flexible model (IMS RLI, 2004) that enables resource lists handling between systems that comply with the specification.

In order to exchange vocabulary between the parts of a learning system, the IMS Vocabulary Definitions Exchange (VDEX) defines a grammar for the exchange vocabulary lists (IMS VDEX, 2004). These lists are formed by domain values that are understood by systems involved in the learning architecture.

The IMS Learning Object Discovery and Exchange (LODE) charter (IMS LODE, 2008) is a proposal to achieve easily discoverable and exchangeable learning objects stored in digital repositories. The draft, rather than specify a model, compiles a set of use cases and scenarios which are inside the scope of the research. LODE is focused on Search, Publication, Query and Metadata. Many of the scenarios considered by LODE are solved in existing specifications, so the final specification of LODE could look like a framework combining e-learning standards to perform discovery and exchange between heterogeneous repositories.

Finally it shall be noted that service-oriented architectures (SOA) are starting to be used massively for building e-learning systems and learning object repositories. We may find a number of interesting norms for the design of these systems to ensure interoperability and to be defined as the architectures of computer systems that support them. The most interesting is IMS Abstract Framework (AF), a framework that covers the entire range of possible e-learning architectures that could be constructed from a set of services based on SOA. It focuses on support for distributed training systems and one of its principles is interoperability. It is also interesting to read "Adoption of Service Oriented Architecture for Enterprise Systems in Education: Recommended Practices" from IMS.

On the other hand we find CORDRA which is one of the most detailed architectures. As an open, standard-based model, it allows to design software systems which are intended for the discovery, sharing and reuse of teaching material through interoperable repositories. Since Ariadne, an architecture has been proposed for repositories which implement SQI-based federated searches.

3. Designing a repository with Service Oriented Architecture

In order to solve interoperability problems it is necessary to define a software architecture which is able to assume certain functions, mainly e-learning systems interoperability (integration) and the reusability of learning objects (Otón S. et al, 2010).

Any developed e-learning architecture should have the following characteristics:

- Open: the goal is to easily create interoperated and connected applications, thus commercial tools from different companies could be assembled into a single global system. To achieve this feature it is necessary to define an architecture following a standard model.
- Scalable: Regardless of the size originally designed of the system, the architecture must be defined to grow in the future. For example: as the educational repositories number increases, the applications in charge of the management must have enough capacity and shall not overload.
- Global: To allow the linguistic and cultural diversity. This is one of the most difficult aims to reach, because most of the applications are designed for an Anglo-Saxon audience: nowadays there are several projects which try to show the same content in different languages depending on the final user.
- Integrated: Not only among the components of the system but among other applications which are not directly related to learning (for example: human resources, financial, knowledge management systems). The final objective is to get the interoperability among all of them.
- Flexible: It is important the ability to implement new solutions without making big changes in the system architecture.

Bellow we present an architecture that solves these problems. From the perspective of interoperability it is based on the main standards and specifications that should be used to guarantee interoperability and reusability of learning objects in repositories. These standards are shown in table 2, arranged by category.

CATEGORY	*SUB-CATEGORY*	*STANDARD*
LEARNING OBJECTS		
	Metadata	*IEEE LOM*
	Packaging	*IMS CC*
	Resources	*IMS RLI*
	Vocabulary	*IMS VDEX*
SEARCHES		
	Interoperability Repositories	*IMS DRI*
	Discovery and Exchange	*IMS LODE*
	Query interoperability	*CEN SQI*
	Publication interoperability	*CEN SPI*
ARCHITECTURES		
	Framework	*IMS AF*
SERVICES		
	Data exchange	*IMS LIS*
	Interoperability	*IMS GWS*

Table 2. Classification of the main standards and specifications related to Learning Objects Repositories.

Web services allow the companies to wrap their business processes publishing them as services; to subscribe to other services and to interchange information among different organizations. Thanks to the maturity achieved by this technology, together with the great expansion it has suffered, its use has gone from being an advantage to be a primary market need.

Web services and SOA architecture constitute a very suitable technology for the implementation of repositories that manage learning objects which are located in different stores of didactic resources, since it offers a clear access to those distributed objects which are in repositories based on different storage and metadata technologies through a single interface.

We suggest a multilayer (or multilevel) architecture, called LORA (Learning Object Reusability Architecture), comprising 3 levels (figure 4). Each of these levels has a fixed role for the correct working of the universal distributed repositories which can be implemented together. These levels are basically fixed by the use of the SQI specification. It is an interface that establishes the search methods that any search system should have (not just in e-learning environments), so that the interoperability of the resulting search system can be guaranteed.

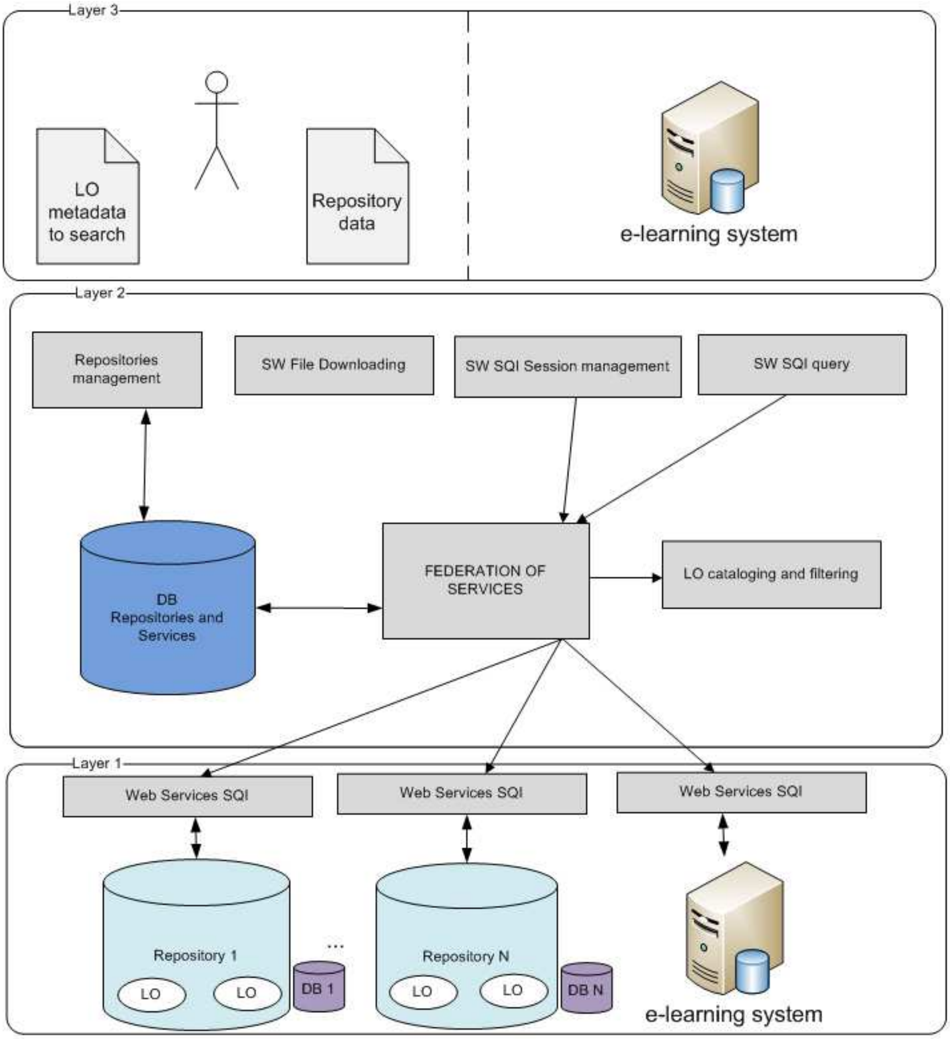


Fig. 4. LORA: A Service Oriented Architecture.

The main features of the architecture are:

- It is a fully interoperable with all other systems and repositories. This not only means that the system can launch searches over other SQI-compliant systems or repositories, but these repositories or systems may also launch searches over LORA-associated repositories. This enables to make content accessibility and reusability as wider and complex as desired.
- It expedites the process of searching learning content. It presents two kinds of searches (which are associated with the implemented PLQL levels-- ProLearn Query Language). This enables a more accurate search level, because the system allows key field-based searches (PLQL level 0) and metadata-based searches (PLQL level 1).

- It is also possible to adapt it to any present or future set of metadata. LORA will only require changing a configuration to virtually adapt the system to any set or subset of metadata. This feature is not available in any other system and it is critical for the system interoperability.

The layers of the LORA system were presented in figure 3. The rest of this section is devoted to describe each of these layers. Standards and/or recommendations employed on each layer are specially highlighted. A summary of them is presented in Table 3.

STANDARD	LAYER 1	LAYER 2	LAYER 3
IEEE LOM	X		
IMS CC	X		
IMS RLI	X		
IMS VDEX	X		
IMS DRI	X	X	X
CEN SQI	X	X	
CEN SPI	X		
IMS AF	X	X	X
IMS LIS	X	X	
IMS GWS	X	X	

Table 3. Standards used by LORA.

The main characteristics of the layers are:

- **Layer 1** comprises all Web services associated to each repository. The IMS GWS (General Web Services) (IMS GWS, 2005) recommendation was employed to build this layer. It determines the way in which Web services are developed and the way in which they interact with each other. These Web services enable the access to learning objects described by the IEEE LOM (Learning Object Metadata) and packaged according to IMS CC (Common Cartridge). In order to access to this data, specifications SQI (Simple Query Interface) and SPI (Simple Publishing Interface) are employed. SQI defines the Web methods to be developed. These methods allow the system to access to the stored learning data. SPI establishes the set of methods to define the way to publish the learning objects in a repository. Moreover, semantic techniques are used to search data. The IMS VDEX (Vocabulary Definition Exchange) (IMS VDEX, 2004) specification has been adopted to tag this metadata and create vocabulary categories. IMS RLI (Resource List Interoperability) (IMS RLI, 2004) is used to arrange, describe and interchange lists of resources included by each learning object. Finally the IMS LIS (Learning Information Services) (IMS LIS, 2010) has been used to make effective every data interchange.

- **Layer 2** comprises the federated search services. They use the services provided by layer 1 to get and handle the data provided by them. Consequently, they also use the SQI specification. Since they are also Web services, they were also developed according to IMS GWS. This layer will also handle all information provided by the repositories, so the use of the IMS LIS specification is also required. In addition to the services requiring SQI methods, another set of services have been developed in order to classify and filter information, and to manage the repositories associated to the search engine. It should also be noted that this layer may be called by other search systems, so it can also be the target of outside calls.
- **Layer 3** corresponds to the presentation layer of the system. All the interfaces that offer access to the system will be found here. Besides, all the aforementioned specifications, IMS DRI (Digital Repositories Interoperability) (IMS DRI, 2003) and IMS AF (Abstract Framework) (IMS AF, 2003) have also been adopted for this layer. IMS DRI defines the features that a repository must offer in order to be interoperable and allow access to its hosted content. IMS AF (Abstract Framework) presents an abstract representation of the set of services that should be used to build an e-learning system in its broader sense, so this specification was the main source to determine the required service.

4. Developing a federated repository with SOAP web services

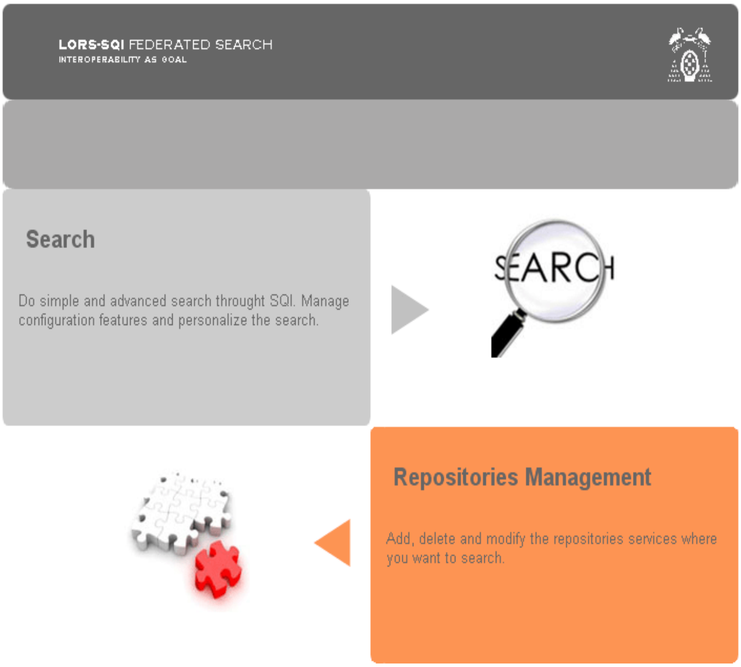
We have presented an architecture with a clear objective: to enable the universal reusability of learning objects. Besides providing an efficient solution to this problem, we also present an implementation that allows showing all the explained theoretical content. The system developed from the above mentioned architecture has been called LORS (Learning Object Reusability System) and represents a federated repository.

The Java platform has been chosen for the development of the system due to the fact that it is considered as the most currently used. Besides, it is perfectly bounded to the Web development. When SOA architecture is used, the presented system will be completely accessible from any other programming language, platform or system that uses any communication protocol. Also it enables to incorporate new services without affecting the already developed system.

MySQL has been chosen as a database manager for the implementation of this architecture. Also the system led by Glassfish Server, has been chosen as an application server.

In the main user interface of the application (figure 5) they can be seen the two possible actions that will be carried out: federated search and catalogue repositories. It is possible to add new repositories within the repositories management regard for the searches to take effect on them. This is one of the new and most important characteristics this system has, as the interoperability among them is guaranteed when all repositories follow the same search interface. Figure 4 shows how it would be the process of adding new repositories.

The other option offered by the initial website is “search”. As it was stated above, the searches done in this system follow the SQI specification; so it is necessary to create a session anonymous or with credentials before starting to look up. There is no user management device in this system, but a SQI management device, so the database table will be directly modified when adding new users.



LORS-SQI FEDERATED SEARCH
INTEROPERABILITY AS GOAL

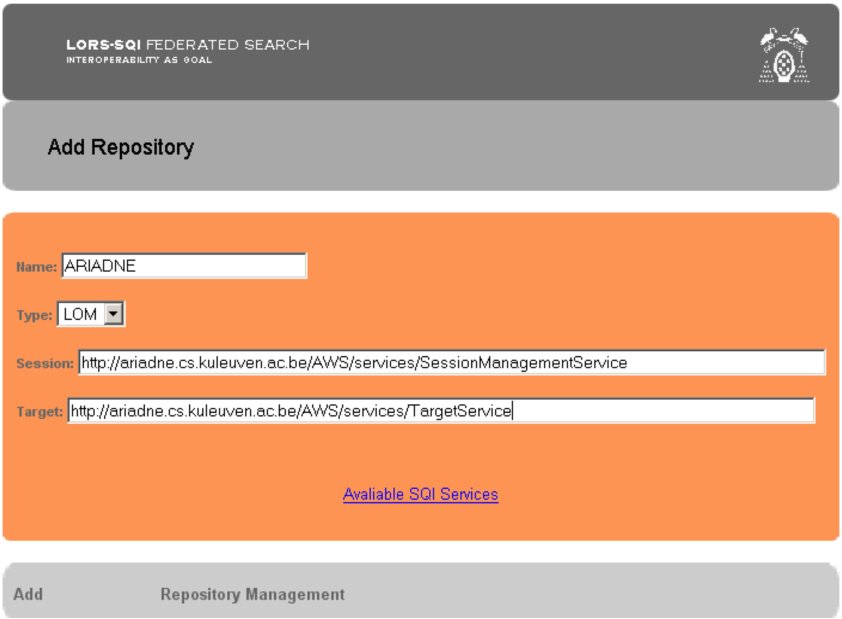
Search

Do simple and advanced search throught SQI. Manage configuration features and personalize the search.

Repositories Management

Add, delete and modify the repositories services where you want to search.

Fig. 5. Main user interface.



LORS-SQI FEDERATED SEARCH
INTEROPERABILITY AS GOAL

Add Repository

Name: ARIADNE

Type: LOM

Session: http://ariadne.cs.kuleuven.ac.be/AWS/services/SessionManagementService

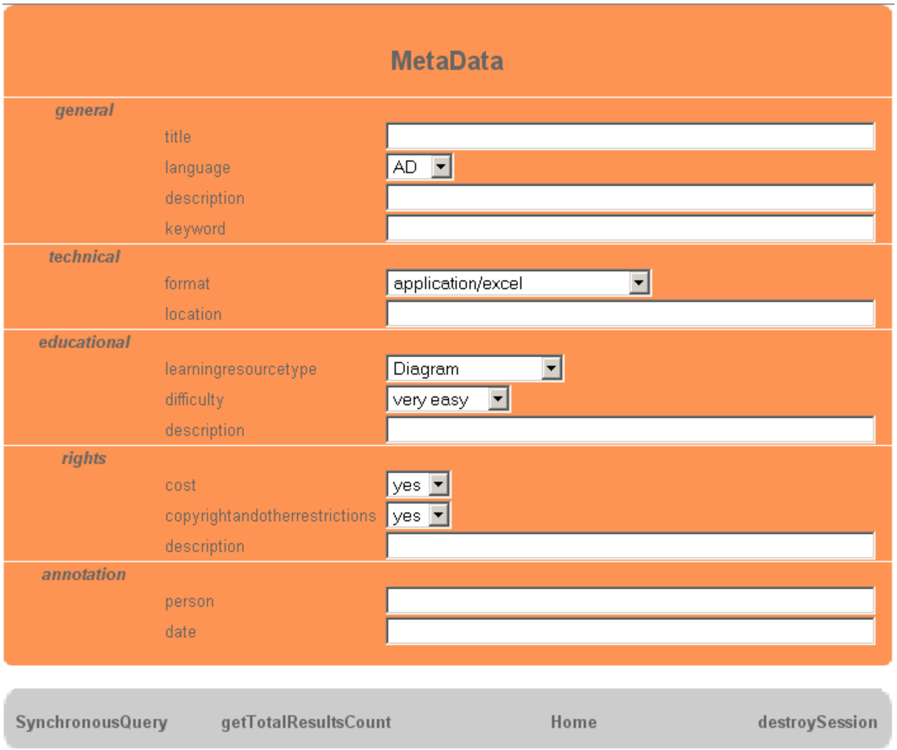
Target: http://ariadne.cs.kuleuven.ac.be/AWS/services/TargetService

Avaliable SQI Services

Fig. 6. Adding a new repository to LORS.

Once validated, the same process will be followed for the validation of all repositories registered in LORS; so if we choose to create an anonymous session, it will be created similarly with all repositories LORS has access. Once created the session, the system offers two types of search: simple and advanced.

The advanced search, also called accurate search, offers the possibility of configuring several SQI parameters as the maximum number of results, the size of the result set, or the initial result. As it is an accurate search, it is done based on educational fields. Figure 7 shows how it would be the search interface.



MetaData

general
title
language: AD
description
keyword

technical
format: application/excel
location

educational
learningresourcetype: Diagram
difficulty: very easy
description

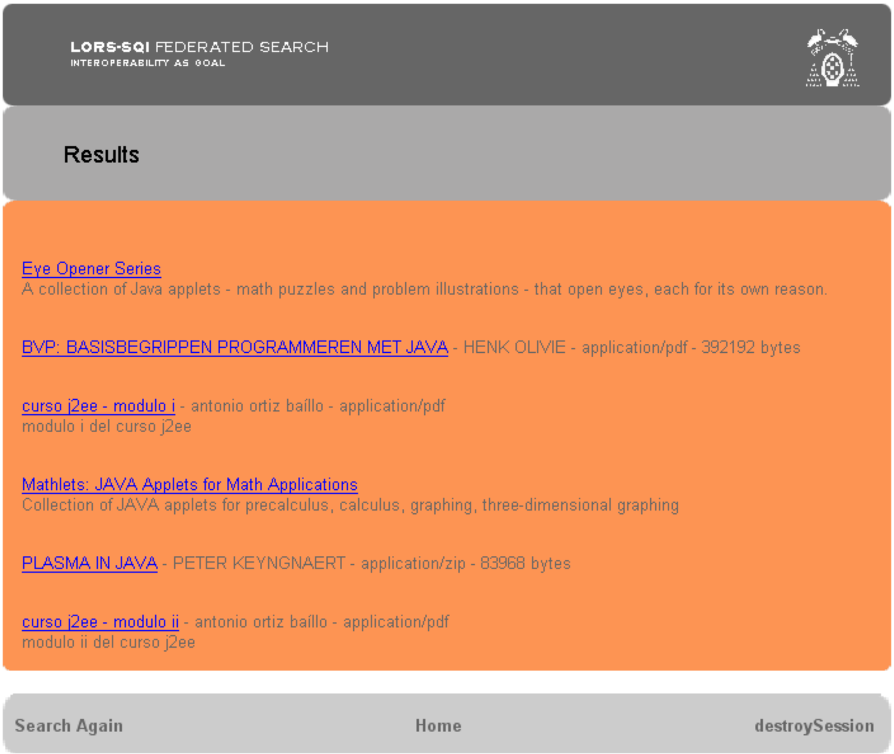
rights
cost: yes
copyrightandotherrestrictions: yes
description

annotation
person
date

Fig. 7. Advanced search interface.

In the case of the simple search or inaccurate search, searches are done on the basis of how many times they appear and how relevant they are the terms given in the search statement in the learning objects metadata. As in the previous case, SQI parameters can be established: maximum number of results, size of the result set and the initial result.

Once the search has been done the system will offer an ordered list with all those learning objects fulfilling the specifications set by the user. Figure 8 shows the offered list. Then, the user can download it by clicking the link. One of the main differences of this system is that the repository associated to LORS has an extra service that does not belong to the SQI specification. This way, it is possible to directly download educational material, instead of link to the author's page as other repositories like Ariadne do.



Results

Eye Opener Series
A collection of Java applets - math puzzles and problem illustrations - that open eyes, each for its own reason.

BVP: BASISBEGRIPPEN PROGRAMMEREN MET JAVA - HENK OLIVIE - application/pdf - 392192 bytes

curso j2ee - modulo i - antonio ortiz baíllo - application/pdf
modulo i del curso j2ee

Mathlets: JAVA Applets for Math Applications
Collection of JAVA applets for precalculus, calculus, graphing, three-dimensional graphing

PLASMA IN JAVA - PETER KEYNGNAERT - application/zip - 83968 bytes

curso j2ee - modulo ii - antonio ortiz baíllo - application/pdf
modulo ii del curso j2ee

Fig. 8. Obtained results interface.

5. Developing a federated repository with RESTful web services

In the construction of distributed information systems the Web services were widely used but today we can see that the traditional Web services that used technologies based on WSDL and SOAP are being replaced by RESTful Web services. Examples of these transitions can be seeing in Google, Amazon, Flickr, Yahoo and YouTube. All this Web sites had WSDL files to describe their Web services and now they have changed to RESTful Web services.

Representational State Transfer (REST) is an architectural style that induces desirable properties, such as performance, scalability, and modifiability that enable Web services to work best on the Web. In the REST architectural style, data and functionality are considered resources and are accessed using Uniform Resource Identifiers (URIs), typically links on the Web. The resources are acted upon by using a set of simple, well-defined operations. The REST architectural style constrains any architecture to a client/server architecture, and is designed to use a stateless communication protocol, typically HTTP. In the REST architecture style, clients and servers exchange representations of resources by using a standardized interface and protocol. RESTful provides four basic methods for interaction with resources, which are: GET (obtain resources) PUT (introduce a new resource), POST (modify a resource) and DELETE (delete a resource).

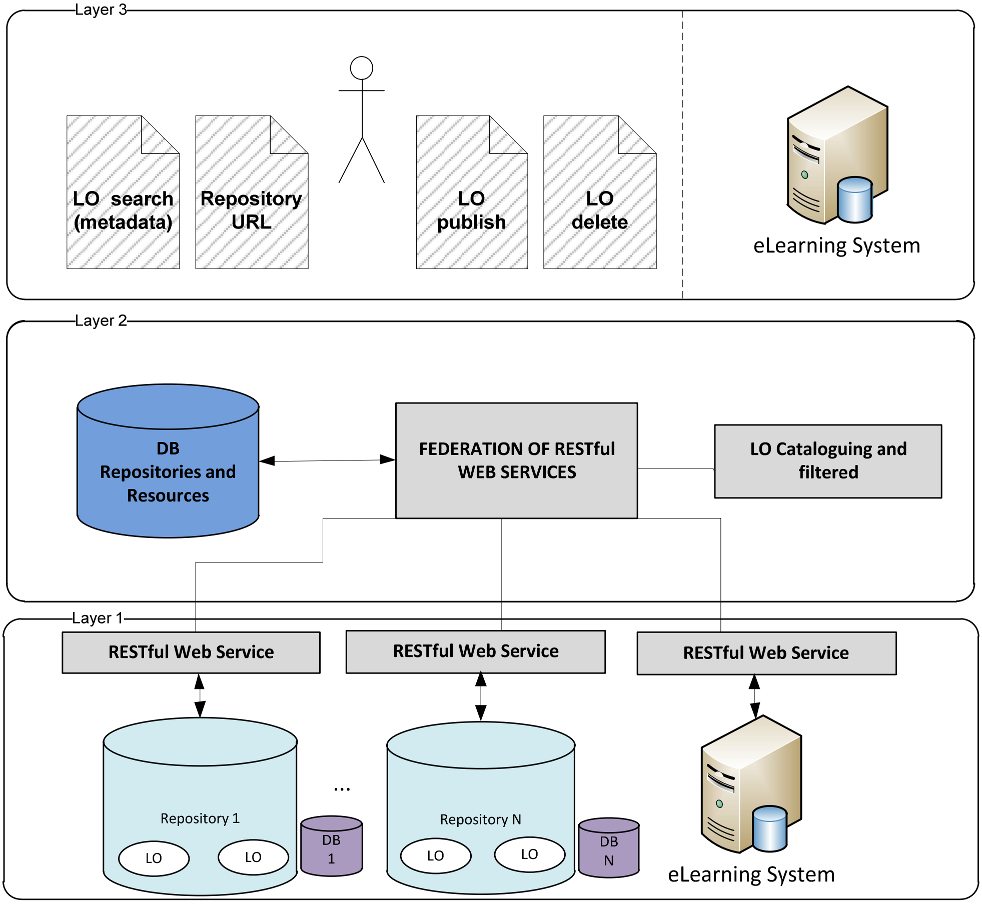


Fig. 9. LORA-REST: A RESTful Service Oriented Architecture.

With the idea of adapting the SQI and SPI specifications to RESTful we have changed and simplify our architecture (figure 9). The main characteristics of the layers are:

- **Layer 1** comprises all RESTful Web services associated to each repository. This RESTful Web services enable access to learning objects described by his metadata. With the GET method, the RESTful Web service enables the system to access to the stored learning data (by its metadata), this method replaces the query SQI methods. With the PUT method, the RESTful Web service enables the system to publish new learning objects in the repository. With the DELETE method, the RESTful Web service enables the system to delete a learning object or his metadata. With the PUT and DELETE methods we complain with the SPI specification.
- **Layer 2** comprises the federated search services. They use the RESTful Web services provided by layer 1 to get and handle the data provided by them. To make a federated search we only need the URL of each repository, and then we use the GET method (with the search criteria of metadata) in the RESTful Web service in each repository and

handle all information provided by the repositories. With the results of each repository the system must classify and filter this information.

- **Layer 3** corresponds to the presentation layer of the system. All the interfaces that offer access to the system will be found here. The system has two primary functions: search learning objects in the federated repositories and add a new repository. If we want comply with the SPI specification we must add two new operations, publish and delete a learning object.

The first step in the design of a system that uses RESTful is to define the data it handles. In the case of repositories of learning objects, data will be as follows. Learning objects are composed of internal elements (resources according to SPI) and metadata describing them. We recommend that the metadata is described using the LOM standard. According to the SPI specification a resource may have 0 or n metadata that describe it. To transfer a resource by reference, its URL must be specified, so the repository will access the address where the resource is in order to include it to the repository.

To adapt the SQI specification to RESTful can perform federated searches in various repositories of learning objects, we only need the URL of the repository in which we want to launch queries. When the user makes a query to one repository it expands this search to the repositories that it has associated. Because we use RESTful, our repository has the role of a client that searches in other repositories. This operation is transparent to the user.

With these requirements we have make a prototype to illustrate how we can develop a RESTful repository. We have designed a database with four tables that represent resources, its metadata, external repositories and external resources. We have transformed these tables in persistent Java classes and from these classes we have developed the RESTful Web services that give access to them. In figure 10 we can see these four persistent classes with his attributes.

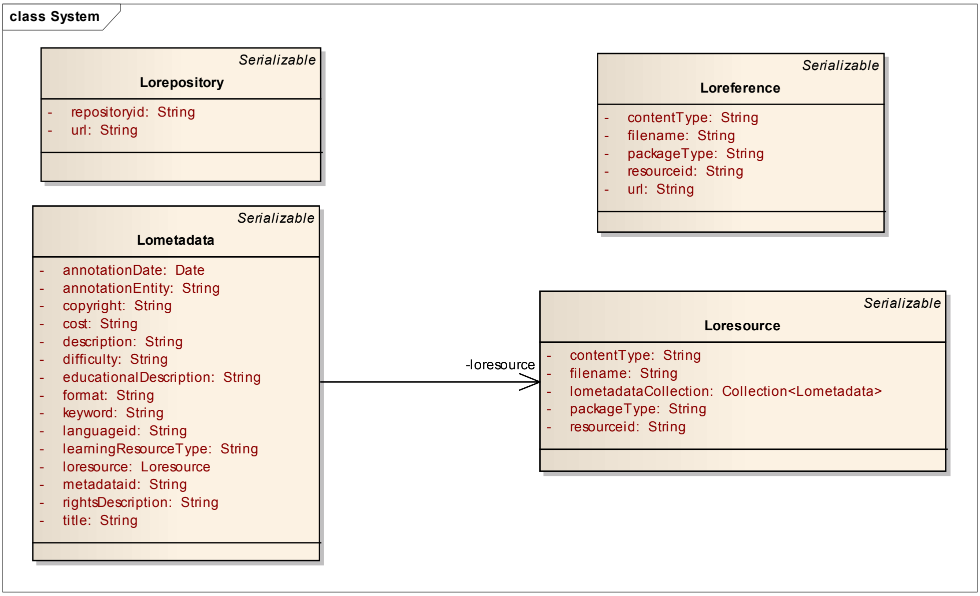
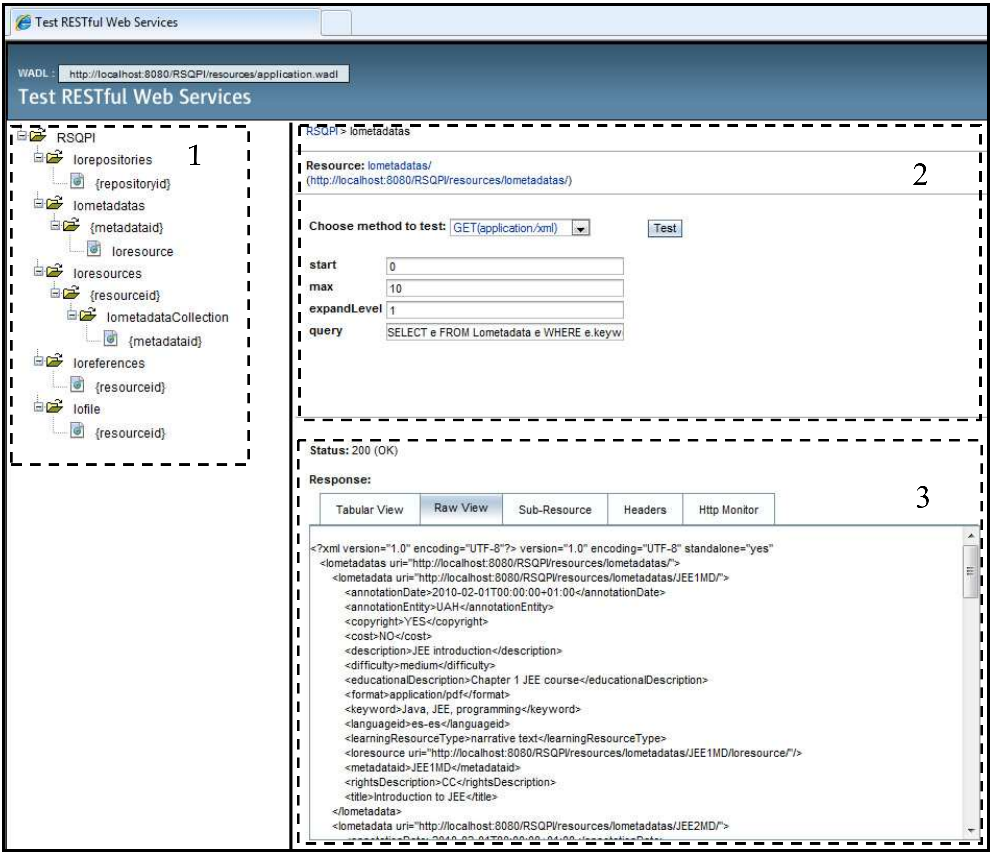


Fig. 10. RESTful prototype data model.

The RESTful Web services for these classes include the methods GET, PUT, POST and DELETE to keep up to date the information in the repository and to comply with the SQI and SPI specifications. We have added a RESTful Web service that enables downloading, inserting and deleting a file associated with a learning object. We have developed the prototype using the NetBeans IDE for the Java programming language as it has useful tools to program and test RESTful Web services.

Figure 11 shows a query to search for metadata records that contain the text "Java" in the "keyword" field of LOM. In this figure we can identify three important areas that are: area 1 is a navigable tree that represents the data of our repository; area 2 allows making queries to the metadata stored in the repository; area 3 shows the results in XML of the query we have made.



Resource: lometadatas/
(http://localhost:8080/RSQPI/resources/lometadatas/)

Choose method to test: GET(application/xml)

start 0
max 10
expandLevel 1
query SELECT e FROM Lometadata e WHERE e.keyw

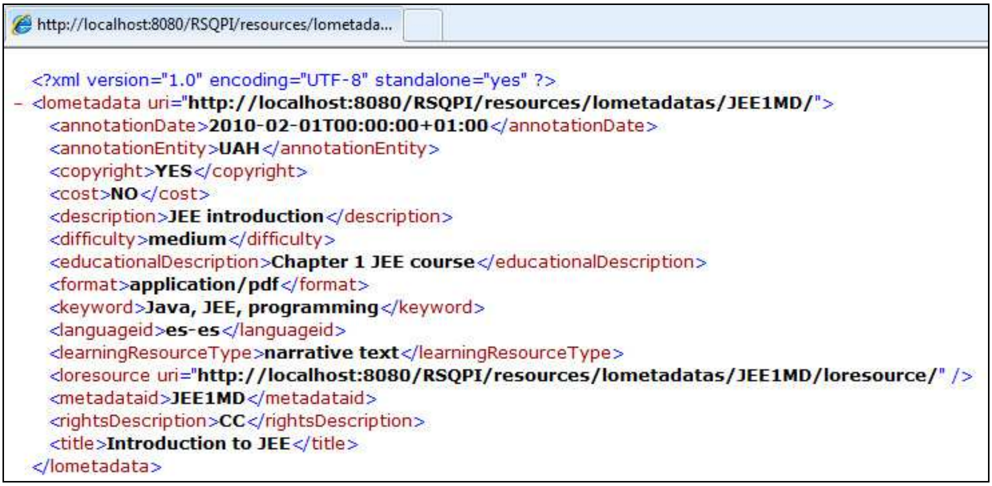
Status: 200 (OK)

Response:

```
<?xml version="1.0" encoding="UTF-8"?> version="1.0" encoding="UTF-8" standalone="yes"
  <lometadatas uri="http://localhost:8080/RSQPI/resources/lometadatas/">
    <lometadata uri="http://localhost:8080/RSQPI/resources/lometadatas/JEE1MD/">
      <annotationDate>2010-02-01T00:00:00+01:00</annotationDate>
      <annotationEntity>UAH</annotationEntity>
      <copyright>YES</copyright>
      <cost>NO</cost>
      <description>JEE introduction</description>
      <difficulty>medium</difficulty>
      <educationalDescription>Chapter 1 JEE course</educationalDescription>
      <format>application/pdf</format>
      <keyword>Java, JEE, programming</keyword>
      <languageid>es-es</languageid>
      <learningResourceType>narrative text</learningResourceType>
      <loresource uri="http://localhost:8080/RSQPI/resources/lometadatas/JEE1MD/loresource/"/>
      <metadataid>JEE1MD</metadataid>
      <rightsDescription>CC</rightsDescription>
      <title>Introduction to JEE</title>
    </lometadata>
    <lometadata uri="http://localhost:8080/RSQPI/resources/lometadatas/JEE2MD/">
```

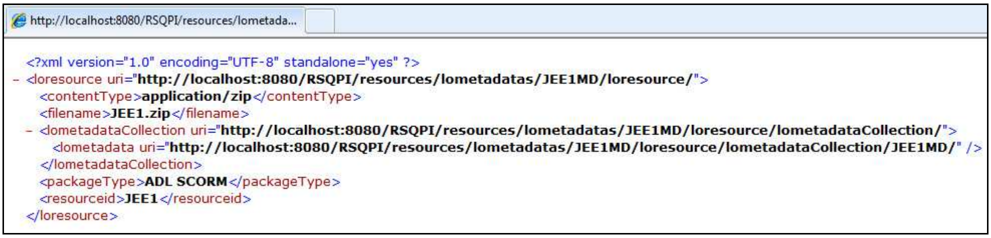
Fig. 11. RESTful metadata query and results.

One important thing to highlight is the possibility of the navigation between our data, for example if we are browsing metadata of one learning object we can access to the resource to which it is associated with. Figures 12 and 13 show this, in figure 12 we are accessing a concrete metadata with the identifier "JEE1MD" and with the URI: "http://localhost:8080/RSQPI/resources/lometadatas/JEE1MD/"; and if we want to access to the resource associated with this metadata (figure 13) we only need its URI, that is: "http://localhost:8080/RSQPI/resources/lometadatas/JEE1MD/loresource/".



```
  <?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <lometadata uri="http://localhost:8080/RSQPI/resources/lometadatas/JEE1MD/">
    <annotationDate>2010-02-01T00:00:00+01:00</annotationDate>
    <annotationEntity>UAH</annotationEntity>
    <copyright>YES</copyright>
    <cost>NO</cost>
    <description>JEE introduction</description>
    <difficulty>medium</difficulty>
    <educationalDescription>Chapter 1 JEE course</educationalDescription>
    <format>application/pdf</format>
    <keyword>Java, JEE, programming</keyword>
    <languageid>es-es</languageid>
    <learningResourceType>narrative text</learningResourceType>
    <loresource uri="http://localhost:8080/RSQPI/resources/lometadatas/JEE1MD/loresource/" />
    <metadataid>JEE1MD</metadataid>
    <rightsDescription>CC</rightsDescription>
    <title>Introduction to JEE</title>
  </lometadata>
```

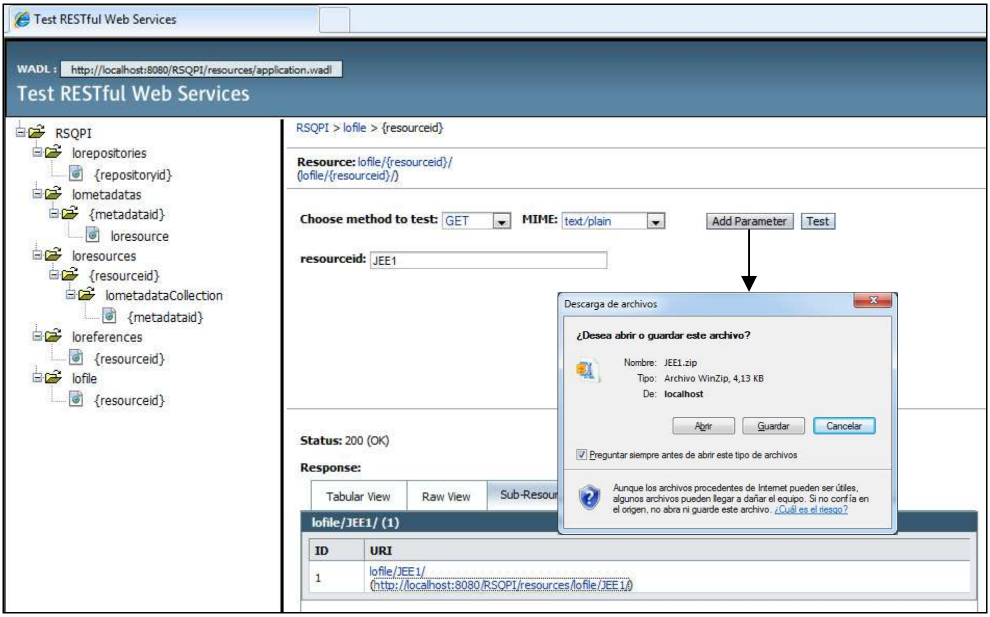
Fig. 12. XML metadata representation.



```
  <?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <loresource uri="http://localhost:8080/RSQPI/resources/lometadatas/JEE1MD/loresource/">
    <contentType>application/zip</contentType>
    <filename>JEE1.zip</filename>
  - <lometadataCollection uri="http://localhost:8080/RSQPI/resources/lometadatas/JEE1MD/loresource/lometadataCollection/">
      <lometadata uri="http://localhost:8080/RSQPI/resources/lometadatas/JEE1MD/loresource/lometadataCollection/JEE1MD/" />
    </lometadataCollection>
    <packageType>ADL SCORM</packageType>
    <resourceid>JEE1</resourceid>
  </loresource>
```

Fig. 13. XML resource representation.

In the figure 14 we present a direct access to a file that represents a packaged learning object. In this case the way of accessing the file is via the identifier of a resource, because in the information of a resource we have the "filename" field, so the Web service uses this filename to retrieve the physical file. In this example we are using the URI: "http://localhost:8080/RSQPI/resources/lofile/JEE1/".



Test RESTful Web Services

WADL : http://localhost:8080/RSQPI/resources/application.wadl

Resource: lofile/{resourceid}/
(lofile/{resourceid}/)

Choose method to test: GET MIME: text/plain

resourceid: JEE1

Descarga de archivos

¿Desea abrir o guardar este archivo?

Nombre: JEE1.zip
Tipo: Archivo WinZip, 4,13 KB
De: localhost

☑ Preguntar siempre antes de abrir este tipo de archivos

Aunque los archivos procedentes de Internet pueden ser útiles, algunos archivos pueden llegar a dañar el equipo. Si no confía en el origen, no abra ni guarde este archivo. ¿Cuál es el riesgo?

Status: 200 (OK)

Response:

lofile/JEE1/ (1)

ID	URI
1	lofile/JEE1/ (http://localhost:8080/RSQPI/resources/lofile/JEE1/)

Fig. 14. Retrieving the file of a packaging learning object.

6. Conclusions

When we want to develop an interoperable repository of learning objects we must know the most important specifications and standards in this filed. The vast majority of these norms are based on Web services and SOA, so their knowledge is essential.

As we have seen in the preceding sections, the proliferation of these standards and specifications focused on the interoperability of learning object repositories is very wide-ranging. As a conclusion we can say that the building of a totally interoperable repository of learning objects, which will consequently allow the learning objects it contains to be reused, must comply with a series of very clear norms. In line with the norms set out above, the steps to be followed in order to achieve an interoperable repository may be summarized as follows:

1. A service oriented architecture should be used when analysing and designing the system software that will house it.
2. In order to integrate the repository in a federated search system, the SQI and SPI specifications should be adopted.
3. Steps 1 and 2 complement each other since, as they both use an SOA architecture, it is extremely easy to integrate the services offered by SQI or SPI.
4. Packaging learning objects using specifications like IMS Common Cartridge and describing their metadata will make those objects reusable.

7. References

ADL Advanced Distributed Learning Initiative (2002). *SCORM*. Available from http://www.adlnet.gov/Technologies/scorm/default.aspx

European Committee for Standardization (2005). *SQI: Simple Query Interface*. Available from ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/WS-LT/cwa15454-00-2005-Nov.pdf

European Committee for Standardization (2010). SPI: Simple Publishing Interface. Available from ftp://ftp.cen.eu/CEN/Sectors/TCandWorkshops/Workshops/CWA16097.pdf

IEEE Standards Association (2002). *IEEE 1484.12.1: Learning Object Metadata (LOM)*.

IMS Global Learning Consortium (2003). *IMS Digital Repositories Interoperability*. Available from http://www.imsglobal.org/digitalrepositories/

IMS Global Learning Consortium (2003). *IMS Abstract Framework*. Available from http://www.imsglobal.org/af

IMS Global Learning Consortium (2004). *IMS Resource List Interoperability*. Available from http://www.imsglobal.org/rli

IMS Global Learning Consortium (2004). *IMS Vocabulary Definition Exchange*. Available from http://www.imsglobal.org/vdex

IMS Global Learning Consortium (2005). *IMS General Web Services*. Available from http://www.imsglobal.org/gws

IMS Global Learning Consortium (2007). *IMS Learning Object Discovery and Exchange*. Available from http://www.imsglobal.org/lode.html

IMS Global Learning Consortium (2008). *IMS Common Cartridge*. Available from http://www.imsglobal.org/cc

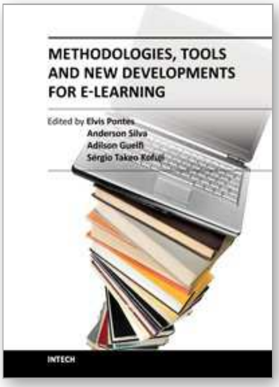
IMS Global Learning Consortium (2010). *IMS Learning Information Services*. Available from http://www.imsglobal.org/lis

Koper, E. (2000). From change to renewal: Educational technology foundations of electronic learning environments, Open University of Netherlands.

Otón, S., Ortiz, A., Hilera, J.R., Barchino, R., Gutiérrez, J.M., De Marcos, L., Martínez, J.J. & Gutiérrez, J.A. (2009). Requirements to ensure interoperability between learning object repositories, *Proceedings of the EEE2009,* pp. 391 – 396, ISBN: 1-60132-100-7, Las Vegas, Nevada, USA, Jul 13-16, 2009.

Otón, S., Ortiz, A., Hilera, J.R., Barchino, R., Gutiérrez, J.M., Martínez, J.J., Gutiérrez, J.A., de Marcos, L. & Jiménez, M.L. (2010). Service Oriented Architecture for the Implementation of Distributed Repositories of Learning Objects. *International Journal of Innovative Computing Information and Control*, Vol.6, No.3, (March 2010), pp. 843-854, ISSN 1349-4198.

World Wide Web Consortium (2004). *Web Services Architecture*. Available from http://www.w3.org/TR/ws-arch/



Methodologies, Tools and New Developments for E-Learning
Edited by Dr. Elvis Pontes

ISBN 978-953-51-0029-4
Hard cover, 332 pages
Publisher InTech
Published online 03, February, 2012
Published in print edition February, 2012

With the resources provided by communication technologies, E-learning has been employed in multiple universities, as well as in wide range of training centers and schools. This book presents a structured collection of chapters, dealing with the subject and stressing the importance of E-learning. It shows the evolution of E-learning, with discussion about tools, methodologies, improvements and new possibilities for long-distance learning. The book is divided into three sections and their respective chapters refer to three macro areas. The first section of the book covers methodologies and tools applied for E-learning, considering collaborative methodologies and specific environments. The second section is about E-learning assessment, highlighting studies about E-learning features and evaluations for different methodologies. The last section deals with the new developments in E-learning, emphasizing subjects like knowledge building in virtual environments, new proposals for architectures in tutoring systems, and case studies.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Salvador Otón, Antonio Ortiz, Luis de-Marcos, Sergio Mazo de Dios, Antonio García, Eva García, José R. Hilera and Roberto Barchino (2012). Developing Distributed Repositories of Learning Objects, Methodologies, Tools and New Developments for E-Learning, Dr. Elvis Pontes (Ed.), ISBN: 978-953-51-0029-4, InTech, Available from: http://www.intechopen.com/books/methodologies-tools-and-new-developments-for-e-learning/developing-distributed-repositories-of-learning-objects

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821