RESTful Service Oriented Architecture for Querying and Publishing Learning Objects in Repositories

Sergio Mazo, Salvador Otón, Luis de-Marcos, Antonio García, Eva García
Computer Science Department. University of Alcalá.
Technical School of Computer Science Engineering.
Alcalá de Henares. Madrid. Spain.
sergio.mazo@gmail.es, {salvador.oton, luis.demarcos, a.garciac, eva.garcial}@uah.es

Abstract—**This paper presents a service oriented architecture for querying and publishing learning objects in distributed repositories. The architecture is based on RESTful web services and proposes the adaptation of A Simple Query Interface Specification for Learning Repositories (SQI) and The Simple Publishing Interface Specification (SPI) specifications for supporting this technology.**

Keywords-component; Learning object; learning repository; querying; publishing; web service.

I. 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 having previous knowledge about the collection, it is only needed a reference to locate it [1].

Most repositories are usually autonomous, i.e., they work as websites that can be accessed through a Web-based interface, providing a search mechanism and a list of categories for conducting the search. However, the possibility of carrying out federated searches in distributed repositories is becoming increasingly important [2].

II. State of the Art

There are federated repositories engines oriented towards sharing and reusing learning content, e.g., Multimedia Educational Resource for Learning and Online Teaching (MERLOT) [3], but the content stored usually does not contain the associated metadata instances inside the resources. Storing metadata packaged with the learning contents increase flexibility and interoperability between learning systems.

Another federations as Alliance of Remote Instructional Authoring and Distribution Network of Europe (ARIADNE) [4] allow sharing full learning objects with its metadata across heterogeneous repositories, using publishing and query interfaces such as SPI [5] and SQI [6], but the current implementations are based on Simple Object Access Protocol (SOAP) [7].

There are several repositories that do not support SQI because it is SOAP based, so it is necessary to adapt the existing publishing and query mechanisms to a modern RESTful approach, in order to support new bridges for communication between repositories and federations of repositories. The main advantages of this new model are:

- Those repositories, which have discontinued SOAP support, can take advantage of SQI benefits.
- The background technology has already been used in some systems.
- The repositories of the federation store learning objects packages with their metadata.

III. Standards and Specifications

This paper is focused on two main specifications: A Simple Query Interface Specification for Learning Repositories (SQI) and The Simple Publishing Interface Specification (SPI). Both of them have been developed by a public initiative known as the CEN/ISSS Learning Technologies Workshop, whose commitment is to guarantee interoperability between learning object repositories.

A. *Simple Query Interface*

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 1 shows how repository A (source) makes a data request from repository B (target). For this communication being possible it is necessary to use a common query language (based on SQI) which both repositories understand. However, the query language may differ inside the repository. In this case, a layer (an SQI component) is responsible for making the conversions needed.

SQI identifies thirteen methods to be provided by such systems, these methods are classified in three categories: configuration methods, session management methods and query methods (synchronous and asynchronous).

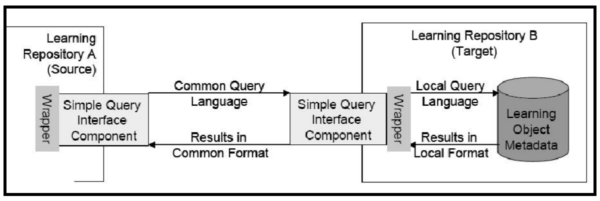

Figure 1. Communication between two SQI repositories

B. *Simple Publishing Interface*

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. In a exchanging data process, semantic interoperability is achieved when data is interpreted in the same way by all the applications involved.

In a typical SPI scenario, two approaches enable data transmission 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 Uniform Resource Locator, 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: data is a resource (e.g., a learning object) while metadata is the description of the resource (e.g., using Learning Object Metadata, LOM [8], to describe the learning object) (Figure 2). 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 matches 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.

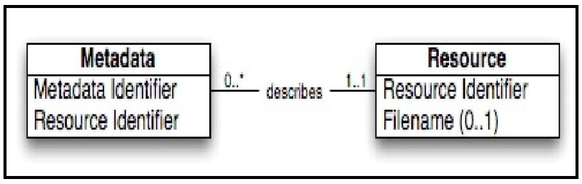

Figure 2. Resources and metadata instances in SPI.

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 so that conformance can also 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.

IV. RESTFUL WEB SERVICES

Roy Fielding introduces in his doctoral dissertation [9] the concept of Representational State Transfer (REST) defined as an abstraction of the architectural elements within a distributed hypermedia system.

REST takes advantage of Hypertext Transfer Protocol (HTTP) [10] [11] for building distributed applications. Components that participate in REST architecture are called resources. These resources are identified by an URL, so they can be accessed and identified in an easy way. The architecture also defines a resource representation as the data found behind the resource URL, e.g., an image file or a web page.

Once these concepts are established, a new web service-based architecture, different of Simple Object Access Protocol (SOAP) web services, has been developed. RESTful web services are those web services based on REST architecture. The operations of the web services are implemented as resources called using HTTP protocol and the results returned by services are resources representations. Operation parameters can be passed using HTTP request parameter mechanism.

This architecture is faster and simpler than SOAP because is not necessary to build heavy clients from Web Service Definition Language (WSDL) files for accessing to the services anymore, just URLs identifying resources are required. Network traffic is also reduced as is not necessary to transmit loud SOAP messages, but lightweight HTTP requests and responses are used instead.

Accessing Resources

HTTP protocol defines an operation set. Some of these operations are used to provide access to RESTful web services:

- GET: Typically used to invoke query operations.
- PUT: This operation is normally used to create new data elements.
- POST: This message should be used for edit data elements.
- DELETE: The common operation for deleting data elements.

Data elements are represented as REST resources. For instance, a web service for managing songs data could be define as a REST resource with an URL to identify it, as http://service.org/songs. If clients need to access to a concrete song, e.g., the song with id 576, they can do that using the URL http://service.org/songs/576. Once the resource is defined, users could perform Create, Read, Update and Delete (CRUD) operations with song data, as the following:

- Create a new song (with ID 576): POST request at http://service.org/songs/576.
- Read song data: GET request at http://service.org/songs/576.
- Update song data: PUT request at http://service.org/songs/576.
- Delete song data: DELETE request http://service.org/songs/576.

Also, if clients need to get a list of data items, they can retrieve it by sending a GET request to http://service.org/songs. So data can be accessed easily with RESTful web services.

In learning repositories context, the CRUD operations are needed for publishing and querying learning objects and its metadata. POST and PUT requests solve the publishing problem and GET requests enable querying operations. Removing objects and its metadata can be done with DELETE requests. Publishing and querying issues can be adapted to RESTful architectures taking all its advantages, i.e., simplicity, performance and an easy way to build clients that comply with the interfaces.

V. PROPOSED ARCHITECTURE

With the idea of adapting the SQI and SPI specifications to RESTful a new architecture is proposed (Figure 3). It is composed by layers, whose main characteristics are:

- Layer 1 comprises all RESTful web services associated to each repository. RESTful web services enable accessing learning objects described by their metadata. With the GET method the RESTful web service allows the system to access to the stored learning data (by its metadata). This method replaces the query SQI methods. 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. For carrying out 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: searching learning objects in the federated repositories and adding a new repository. If we want comply with the SPI specification we must add two new operations, namely 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. For transfering 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 so that 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 queries a repository, the search is extended to the repositories are 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.

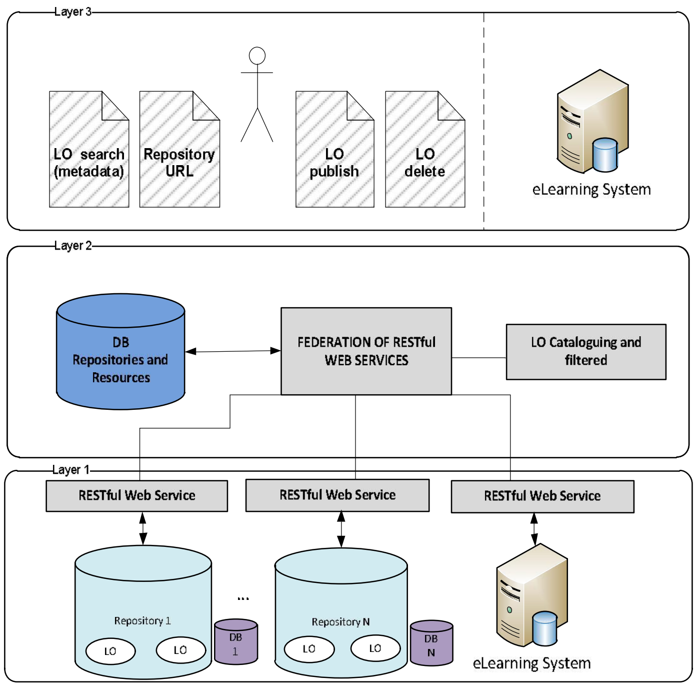

Figure 3. A RESTful Service Oriented Architecture

If we want to build a federated repository of learning objects the SQI and SPI specifications can provide a considerably interoperability because it is no longer necessary to implement different search / publish mechanisms for different repositories.

The problem with the SQI specification is that a lot of repositories that previously supported it are no longer doing so [12]. We believe that this occurred because the SQI specification is based on out-dated technology (SOAP and WSDL) and therefore the main conclusion is that SQI must upgrade and evolve from traditional web services to RESTful.

VI. Conclusions

RESTful web services are especially suitable for the implementation of SQI and SPI specifications because both are based on the management of the data stored in repositories of learning objects, so their use to define these specifications would ease their interoperability via HTTP.

The architecture proposed can help developers of federated repositories of learning objects in their developments using two of the most important specifications to ensure interoperability.

References

[1] IMS Global Learning Consortium, "Digital Repositories Interoperability". 2003. Available from http://www.imsglobal.org/digitalrepositories/.

[2] D.R. Rehak, P. Dodds, L. Lannom, "A Model and Infrastructure for Federated Learning Content Repositories". 2005. Available from: http://www.mendeley.com/research/a-model-and-infrastructure-for-federated-learning-content-repositories/#page-1.

[3] Multimedia Educational Resource for Learning and Online Teaching (MERLOT). 2007. Available from http://www.merlot.org/merlot/index.htm.

[4] Alliance of Remote Instructional Authoring and Distribution Network for Europe (ARIADNE) – Knowledge Pool System (KPS). 2007. Available from: http://ariadne.cs.kuleuven.be/AriadneFinder/.

[5] European Committee for Standarization: "SPI: Simple Publishing Interface". 2010. Available from: ftp://ftp.cen.eu/CEN/Sectors/TCandWorkshops/Workshops/CWA16097.pdf.

[6] European Committee for Standardization. "SQI: Simple Quey Interface" 2005. Available from ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/WS-LT/cwa15454-00-2005-Nov.pdf.

[7] World Wide Web Consortium: "SOAP Version 1.2 Part 1: Messaging Framework". 2007. Available from: http://www.w3.org/TR/soap12-part1/.

[8] IMS Learning Object Metadata, "IMS Global Learning Consortium". 2002. Available from: http://www.imsglobal.org/metadata/index.html.

[9] R. Fielding, "Architectural Styles and the Design of Network-based Sortware Architectures". University of California, Irvine. 2000. Available from: http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

[10] T. Berners-Lee, R. Fielding, H. Frystyk, "Hypertext Transfer Protocol 1.0". RFC-1945. 1996. Available from http://tools.ietf.org/html/rfc1945.

[11] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, "Hypertext Transfer Protocol 1.1". RFC-2616. 1999. Available from http://tools.ietf.org/html/rfc2616.

[12] S. Otón, L. de-Marcos, J.R. Hilera, R. Barchino, J.M. Gutiérrez, A. García, and E. García, "Adapting SQI and SPI specifications to conform RESTful". Proc. of the Fourth International Conference on Internet Technologies and Applications (ITA 11), Centre for Applied Internet Research (CAIR), Sept. 2011, pp. 86-93.