

LUISA

*Learning Content Management System Using Innovative Semantic Web
Services Architecture*

IST- FP6 - 027149



Deliverable D4.5

LOMR architectural prototype evaluation report

Miguel-Angel Sicilia
Salvador Sanchez
Elena García Barriocanal

Due date of deliverable: 31/08/2007

Actual submission date: 30/09/2007

Start date of the project: 01/03/2006

Duration: 30 Months

Miguel-Angel Sicilia
University of Alcalá

Version 1.0, dated 30/08/2007

Change History

Version	Date	Status	Author (Partner)	Description
0.1	10.08.2007	Initial Draft	Miguel-Angel Sicilia	First draft
0.1.1	30.08.2007	Initial version	Miguel-Angel Sicilia	First revised version
1.0	30.08.2007	Final version	Miguel-Angel Sicilia	Final version after reviews

EXECUTIVE SUMMARY

The LUISA LOMR is an open-source software framework that provides the functions to store and query for learning objects stored in semantic form. The semantics are provided by using WSML stored in RDF format as the underlying data management mechanism.

The architectural prototype of the LUISA LOMR, described in deliverable 4.3 serves the role of an early assessment of the key architectural issues behind the LOMR. This deliverable reports on the evaluation of that architectural prototype.

Document Information

IST Project Number	FP6 – 027149	Acronym	LUISA
Full title	Learning Content Management System Using Innovative Semantic Web Services Architecture		
Project URL	http://www.luisa-project.eu /		
Document URL			
EU Project officer	Kypros Kyprianou		

Deliverable	Number	4.5	Title	LOMR architectural prototype evaluation report
Work package	Number	4	Title	Learning Object Metadata Repositories and LOMR Integration Development

Date of delivery	Contractual	31.08.07	Actual	30.09.07
Status	Version 1.0, dated 30/08/2007		final <input checked="" type="checkbox"/>	
Nature	Report <input checked="" type="checkbox"/> Demonstrator <input type="checkbox"/> Other <input type="checkbox"/>			
Dissemination Level	Public <input checked="" type="checkbox"/> Consortium <input type="checkbox"/>			
Abstract (for dissemination)	This deliverable describes the evaluation of the LUISA LOMR architectural prototype.			
Keywords	Learning objects, learning object repositories, Semantic Web, Semantic Web Services, ontologies, WSMO, IRS-III, IEEE LOM, IMS LD.			

Authors (Partner)	Miguel-Angel Sicilia (UAH), Salvador Sánchez-Alonso (UAH), Elena García-Barriocanal (UAH)		
Responsible Author	Miguel Angel Sicilia	Email	msicilia@uah.es
	Partner	UAH	Phone

Project Consortium Information




Partner	Acronym	Contact
Atos Origin S.A.E. (Coordinator)	ATOS 	Nuria de Lama Atos Origin S.A.E. c/ Albasanz 12 E-28037 Madrid, Spain Email: nuria.delama@atosorigin.com Tel.: +34 91 214 9321 Fax: +34 91 754 3252
University of Alcalá de Henares	UAH 	Dr. Miguel-Angel Sicilia Information Research Unit University of Alcalá Ctra. De Barcelona, Km 33.6 E-28871Alcalá de Henares (Madrid), Spain Email: msicilia@uah.es Tel.: +34 91 886 6603 Fax: +34 91 885 6646
University of Uppsala	ULL 	Dr. Ambjorn Naeve University of Uppsala Kyrkogårdsgatan 2 C Uppsala Email: amb@nada.kth.se Fax: +46 184-716-294
Open University	OU 	Dr. John Domingue Knowledge Media Institute, The Open University, Walton Hall, Milton Keynes, MK7 6AA, United Kingdom Email: j.b.domingue@open.ac.uk Tel.: +44 1908 655014 Fax: +44 1908-653-169
University Henri Pointcaré	UHP 	Dr. Monique Grandbastien University Henri Pointcaré Vandoeuvre les Nancy 54506, PO Box 239, France. Email: monique.grandbastien@loria.fr Fax: : +33 383-278-319
Giunti Interactive Labs S.r.l.	GIUNTI 	Dr. Fabrizio Giorgini Giunti Interactive Labs S.r.l. Abbazia dell'Annunziata Via Portobello Baia del Silenzio 16039 Sestri Levante (GE), Italy Tel.: +39.0185.42123 Fax: +39.0185.43347
EADS FRANCE – Innovation works	EADS 	Anne Monceaux EADS FRANCE – Innovation works Avenue Didier Daurat - Centreda 1, Toulouse, 31700, France. Email: anne.monceaux@airbus.com Tel.: +33 561-184-725 Fax: +33 561-187-611

TABLE OF CONTENTS

EXECUTIVE SUMMARY	3
TABLE OF CONTENTS.....	6
1 INTRODUCTION.....	7
2 BACKGROUND: OVERALL ARCHITECTURE OF LUISA AND LUISA LOMR.....	7
2.1 Overall LUISA Architecture	7
2.2 LUISA LOMR Architectural prototype architecture.....	8
3 APPROACH FOR THE ARCHITECTURAL PROTOTYPE EVALUATION.....	10
3.1 Context	10
3.2 Methodology	11
4 ARCHITECTURAL VIEWS	11
4.1 Use cases addressed	11
4.2 Architectural view of the design model.....	12
4.3 Architectural view of the deployment model.....	14
5 EVALUATION OUTCOMES.....	15
5.1 Key artefacts and degree of coverage of planned use cases	15
5.2 Status of testing	15
5.3 Report on extension case	18
6 DISCUSSION AND OUTLOOK.....	18
6.1 Pending architectural issues	19
6.2 Plan for the next release	19

1 INTRODUCTION

The mission of LUISA is that of exploiting the advantages of a Semantic Web Service Architecture to make richer and more flexible the processes of query and specification of learning needs in the context of Learning Management Systems and Learning Object Repositories.

This document describes the outcomes of the work on task 4.3 labelled “*Architectural prototype development*”, and part of the effort of task 4.4 labelled “*Development of integration case specification*”.

The intention of the work reported herein is that of reporting on the evaluation of the implementation of an architectural prototype that exercises the main functionalities that are required in LUISA LOMR (Learning Object Metadata Repository), as defined in deliverable 4.3 labelled “*LOMR Architectural prototype specification*”. These include the storage of learning object metadata in semantic form, the provision of a service-oriented interface and the import of data in non-semantic form, among others.

This document is structured as follows. Section 2 provides some background information on the LUISA LOMR and its positioning in the LUISA architecture. Then, section 3 introduces the criteria used for the evaluation of the LUISA LOMR architectural prototype. Section 4 reports on the results of the evaluation. Finally, Section 5 provides concluding remarks and sketches the directions of the subsequent development work towards the LUISA LOMR reference implementation.

2 BACKGROUND: OVERALL ARCHITECTURE OF LUISA AND LUISA LOMR

This section provides a brief synthesis of the state of the overall LUISA architecture as the big picture of the whole project in which the LOMR architecture discussed plays a role.

2.1 Overall LUISA Architecture

The main components of the LUISA framework architecture are depicted in Figure 1.

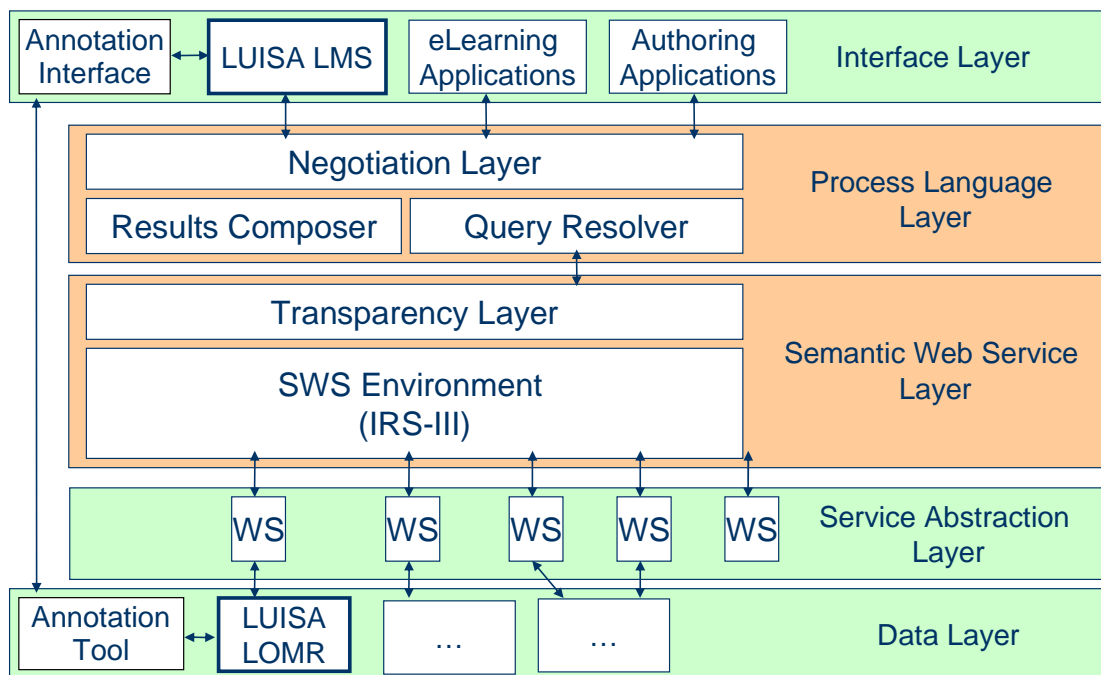


Figure 1. Main components of the LUISA Architecture.

The architectural prototype described here deals with the specific of LUISA LOMR repositories, labelled as *Data Layer* in the picture. LUISA LOMR instances provide a Web Service interface to the *Semantic Web Service Layer*. This enables the use of SWS technology to select the best repository for a given learning need that arises at the interface layer. It also enables that specialized components as custom *Query Resolvers* and *Result Composers* benefit from the availability of different, heterogeneous LOMR instances. Further, LUISA LOMR instances can be accessed through the *Annotation Tool* for the edition of metadata in semantic form.

2.2 LUISA LOMR Architectural prototype architecture

Figure 2 depicts the main components of the LUISA LOMR architectural prototype. The components that were (partially) implemented as part of WP4 are depicted with pointed lines. Additional LUISA software components are depicted in shaded (yellow) colour outside the main boxes, and with a reference to the WP(s) in which the bulk of the work is carried out.

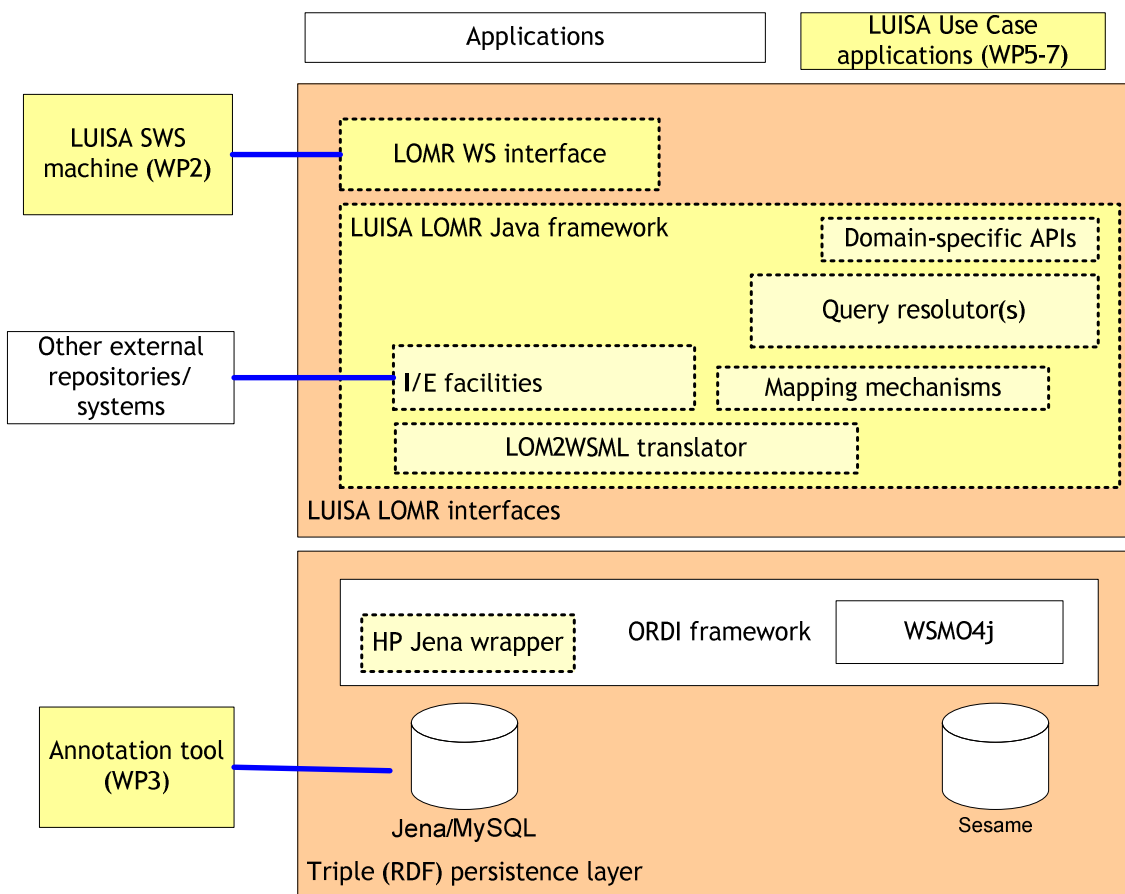


Figure 2. Main elements of the architectural prototype

The *Triple persistence layer* abstracts applications and LOMR components from the RDF storage of WSMML data. Eventually, applications might access the data at the RDF level, either directly or through some kind of adapter. Work in that direction for the prototype is intended to include assessment of existing distributed triple querying interfaces.

The *Luisa LOMR interfaces* feature the interfaces to applications and the core components that will perform queries and composition. The following is a brief description of the elements that will be included in the prototype:

- The LOM2WSML translator libraries (described in a separate document) provide the required translation functionality from IEEE LOM-compliant metadata to the ontology representation in WSMML. The early implementation of these libraries uses the *LOM for Java* libraries for the parsing of XML LOM records.
- Import facilities are based on the translation libraries. Specific import cases will include metadata from the ARIADNE¹ and CAREO² repositories.

¹ <http://www.ariadne-eu.org/>

² <http://careo.netera.ca/>

- The mapping mechanisms feature the translation of non-IEEE LOM compliant metadata (which is covered by the LOM2WSML translator libraries) in the form expressed by domain-specific libraries to the WSML representation.
- Query resolvers or resolvers (including eventually specific composition algorithms) are the central component for advanced retrieval capabilities.
- Domain-specific APIs represent the view for different domain ontologies or special-purpose ontologies that can be combined with the core learning object ontologies for the purpose of semantic search for a given application or domain. These include the competency ontology (described in a separate document) delivered as part of LUISA work.
- The LOMR WS interface is the decoupled interface to other parts of the LUISA selection and composition framework, and also to other applications or systems.

An important issue in the architecture is that the LOMR provides a Web Service interface for applications and the rest of the LUISA components. A part of this interface will be used by the *Semantic Web Service Layer*, in which the semantic annotations – according to the WSMO framework – will be stored.

3 APPROACH FOR THE ARCHITECTURAL PROTOTYPE EVALUATION

This section describes the approach followed to evaluate the outcomes of the architectural prototype.

3.1 Context

Software architecture can be defined in broad terms as “*the software components, their external properties, and their relationships with one another.*” The SWEBOK guide³ adopts this definition, which considers architecture as a blueprint, “*a description of the subsystems and components of a software system and the relationships between them*”.

The main objective of the architectural prototype is thus to **early shape the main architectural aspects of the LUISA LOMR reference implementation.**

Following the philosophy of Open architecture definitions, the concrete objectives of the LUISA LOMR Architectural prototype are:

1. To assess the application software architecture.
2. To provide an evolvable baseline that will end up in the software components of the reference implementation.
3. To signify completion of the Initiation phase.

³ <http://www.swebok.org>

3.2 Methodology

The evaluation of the architectural prototype has as its general objective checking that the objectives of the prototype have been achieved. This includes the following:

- Check that the executable code base artefacts have been developed, and that they are functional.
- Check that the use cases planned have been covered by the prototype.
- In the cases of software frameworks as the LUISA LOMR, check that the architecture can be extended.

All of these aspects are reported below.

4 ARCHITECTURAL VIEWS

4.1 Use cases addressed

The LOMR is intended to be a framework rather than an end-used application for a particular domain. As a software framework, it will provide a number of services that would eventually be used in the implementation of learning object metadata repositories in different organizations and/or domains.

The list of cases that were planned to be addressed in the prototype, included in the architectural prototype specification, is provided here as the expanded scenario descriptions that drove the development process.

UC_I	Storage of learning object metadata in semantic form, using conventional data stores (e.g. relational DBMSs).
Scenario UC_I	An existing IEEE LOM metadata record expressed in XML form is stored in the LOMR (and parts of it will be later retrieved as the result of a query).

UC_II	Provision of query services that allow for different query languages, from existing IEEE LOM based approaches to specific queries that exploit the ontologies stored as part of the semantic metadata.
Scenario UC_II	Different learning object references are retrieved using different criteria. One of these criteria should be based on IEEE LOM metadata.

UC_III	Provision of a generic, service-oriented interface to the metadata, which will be used by other components of LUISA.
Scenario UC_III	Some LUISA module invokes a conventional SOAP-based Web service to query for learning objects stored inside the

	LOMR and then retrieves a piece of metadata.
--	--

UC_IV	Storage of composite learning objects, which may reference metadata in other semantic repositories.
Scenario UC_IV	Some new learning object will be created inside the LOMR as a result of an explicit call that uses some criteria for composition.

UC_V	Integration of existing metadata in other repositories.
Scenario UC_V	Metadata from external, conventional repositories will be stored in the LOMR.

UC_VI	Integration with the architectures developed in the rest of LUISA work packages.
Scenario UC_VI	The LUISA Annotation Tool accesses a LOMR instance and edits a piece of learning object metadata ⁴ .

It is important to highlight that the use cases developed do not include the design of an end-user interface, since the main usage of the LOMR inside LUISA is as a data back-end.

4.2 Architectural view of the design model

The overall model of the design relies on the basic separation of concerns of three entities that are abstracted in the following interfaces.

Interface	Responsibility
<code>LOMetadataRepository</code>	Data management and repository creation.
<code>LOMRSession</code>	Session management.
<code>LOMRQueryManager</code>	Query management.

The basic pattern for any interaction with a LOMR instance is first acquiring an instance of `LOMetadataRepository`. For example, the `LOMetadataRepositoryFactory` class provides a basic controlled creational pattern for that purpose. Then, applications need to interact through an instance of `LOMRSession` that is acquired through the `LOMetadataRepository` instance.

⁴ NOTE: UC_III covers the integration with all the modules of the LUISA architecture except the Annotation Tool, so this UC deals with the latter.

Session management provides a place to manage the storage and usage of user profile data. Finally, sessions allow for gaining access to LOMQueryManager instances of different kinds. This basic pattern is sketched in Figure 3.

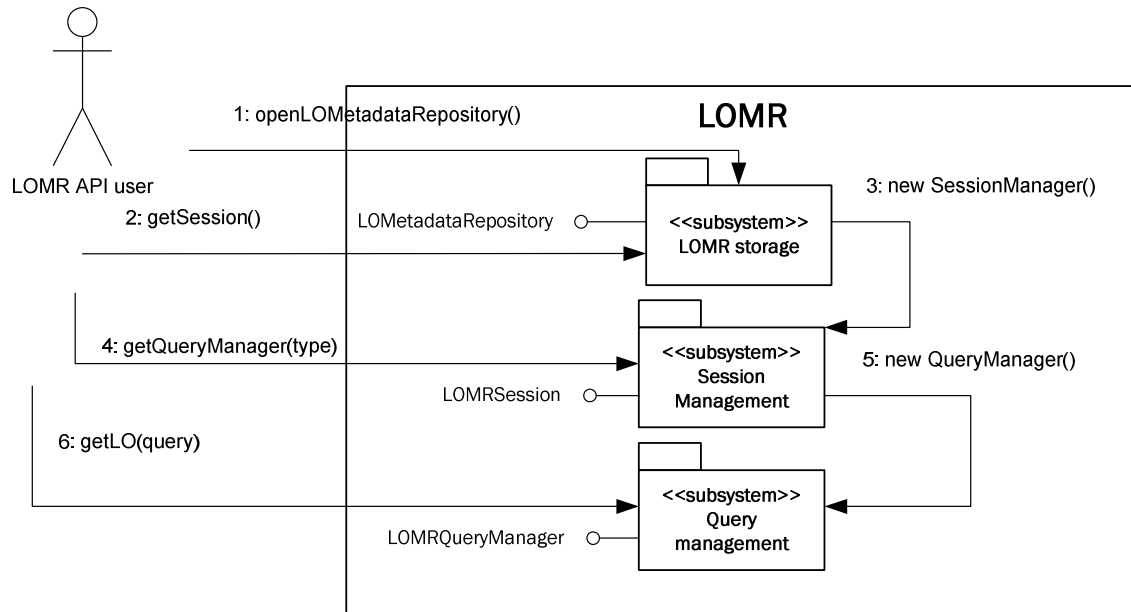


Figure 3. Subsystems collaborating in query resolution in the LOMR

The structure of the code base is centred on different implementations of the LOMR, which encapsulate the specifics of different ontological models. In consequence, there is a *family of repositories* which share the functionality of a core learning object metadata repository featuring the ontologies of IEEE LOM and IMS LD (Level A). The Java package naming scheme follows this organization. Figure 4 provides a view of dependencies among ontologies that serves to illustrate that structure.

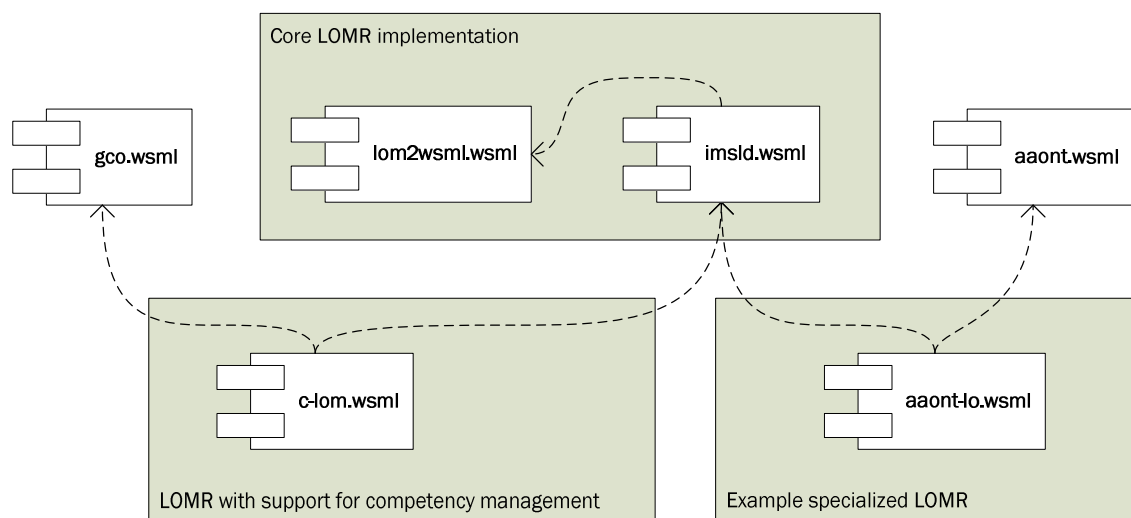


Figure 4. Overview of the ontology structure as a family of repositories.

Figure 4 shows that the IEEE LOM and IMS LD ontologies together serve as the data model for the LOMR core implementation. From it, specialized LOMR implementations can be derived. For example, the generic competency ontology (*gco*) represents a particular view of learning resource selection based on the notion of competency as “observable person behaviour”. Other accounts of learning theories or instructional approaches could eventually be implemented as separate repositories. Figure 4 also provides as example the use of specialized functions related to the field of Art & Architecture (*aaont*). In this case, it is the domain ontology which provides support for specific queries that exploit the knowledge represented in a particular ontology.

There is also a significant pattern of interaction that affects `LOMRQueryManager` structure. The idea is that once a user application has access to a query manager instance, the dialogue should be structured in two phases:

- **LO selection phase.** In this phase, the client application uses a method to query for learning resources matching some given criteria. The expression of those criteria depends on the query manager type. The results of that query will be one or several matching “learning object references” (*LORef*). Some query managers might return a kind of “explanation” of the selection returned, but this is not mandatory.
- **Metadata retrieval phase.** Once the client application has the *LORefs*, it can use them to get any kind of metadata related to the learning object. This includes the URL where the actual learning content (files) resides. The retrieval of metadata for the learning objects returned as query results typically depends on the layout of the end user interface.

The implementation for the second phase is based only on IEEE LOM metadata in the current implementation.

4.3 Architectural view of the deployment model

An arbitrary number of LOMR instances of different kinds can be created for different purposes. There is a requirement that every instance of `LOMetadataRepository` is self-conscious of its access point (URL) to access the WS interface for querying. This is accomplished by the notion of Learning Object reference, which contains the information for that purpose.

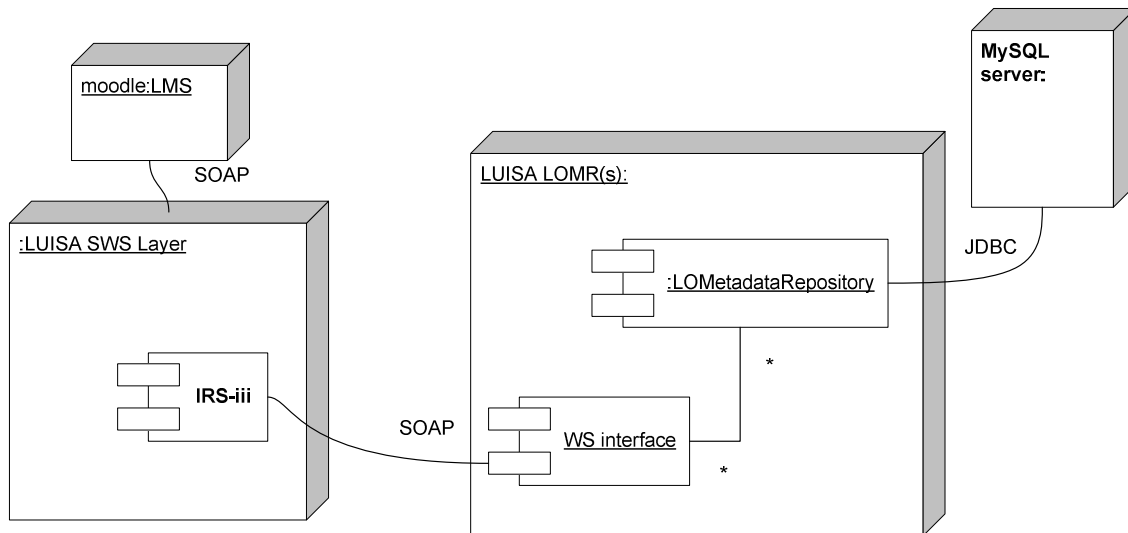


Figure 5. Typical deployment configuration for the LUISA LOMR

Figure 5 depicts the typical physical configuration components, emphasizing the LOMR as a black-box to the rest of the architecture that communicates through a pure Web Service interface. This enables configurations in which SWS facilities combine calls to LOMR instances with calls to other, non-semantic repositories.

5 EVALUATION OUTCOMES

5.1 Key artefacts and degree of coverage of planned use cases

The following Table summarizes the status of the key deliverables of the architectural prototype.

Deliverable	Description	Status
Repository code base	The code base for the evolving reference implementation.	Published and tested against the basic use cases identified above.
User manual	Manual for API users of the LOMR	Initial version developed.
Instances of LOMR for LUISA use cases	Instances of the LOMR for testing purposes.	Developed and running.

5.2 Status of testing

Testing on the architectural prototype has been done at the following levels:

- Unit testing using JUnit tests.
- A preliminary load test.

JUnit tests are spread across the different modules. The following Table gives an overview of the distribution of unit tests across packages.

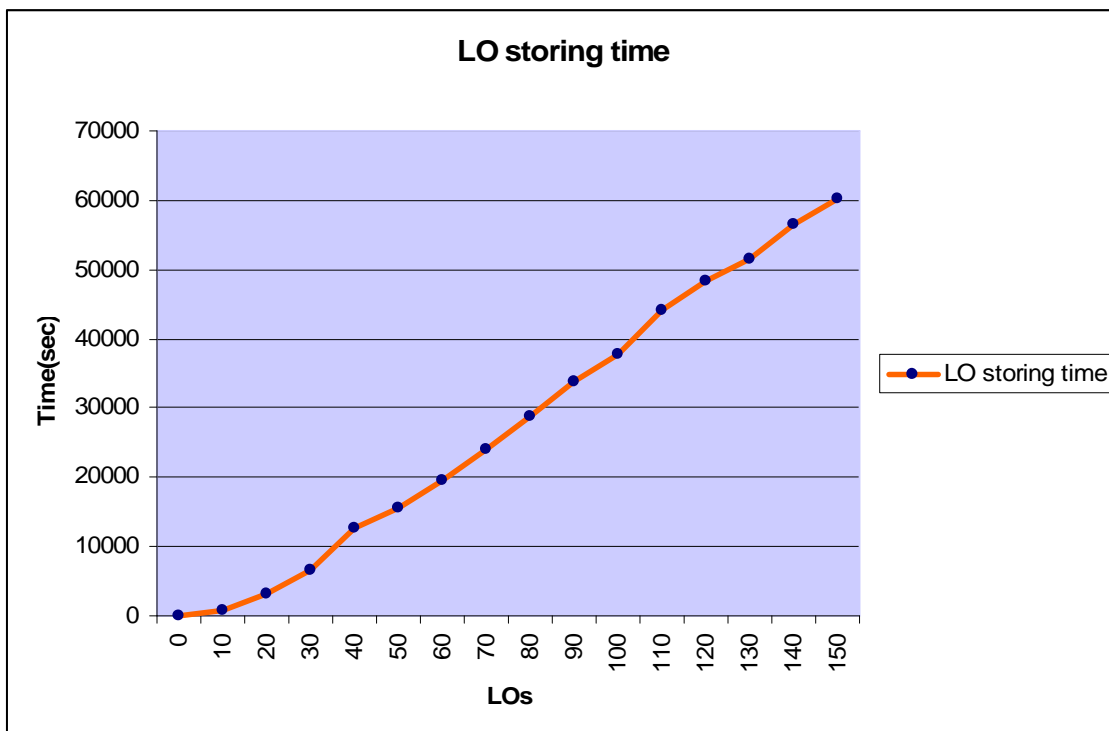
Package	Number of classes/interfaces	Number of test classes
lomr	70	20
lomr+competencies	63 (additional)	15 (additional)

The load test aimed at observing the effect of two variables in the response time of a server with an instance of the LOMR.

The storage test has been performed using a special example class in luisa-lomr distribution, *BulkMetadataLoadTest* class. This class translates a XML learning object to an ontology representation of this learning object in WSML, and stores it in a database.

The graph represents the time that takes the program to translate and store the LOs. The translation and storage of 10 learning objects takes 825 seconds meanwhile the translation and storage of 150 learning objects takes 60173 seconds. The storing time indicates the total time that the program requires translating and storing the number of the learning objects indicated.

The computer used for the tests is an Intel Core 2 CPU 2.13GHz, 3.50 GB of RAM and Microsoft Windows Xp Pro. with SP2 installed.

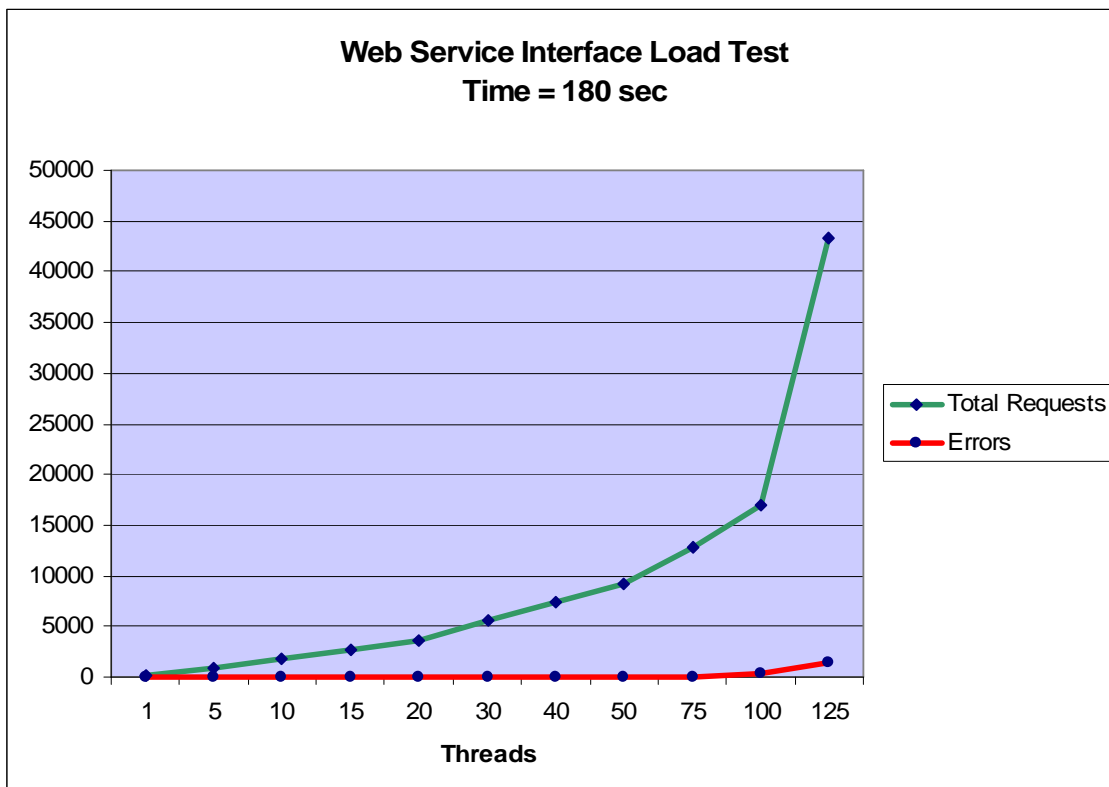


The response time behaves approximately linear, and it can be regarded as appropriate, provided that the conversion from IEEE LOM XML to WSMML is the main driver of response time.

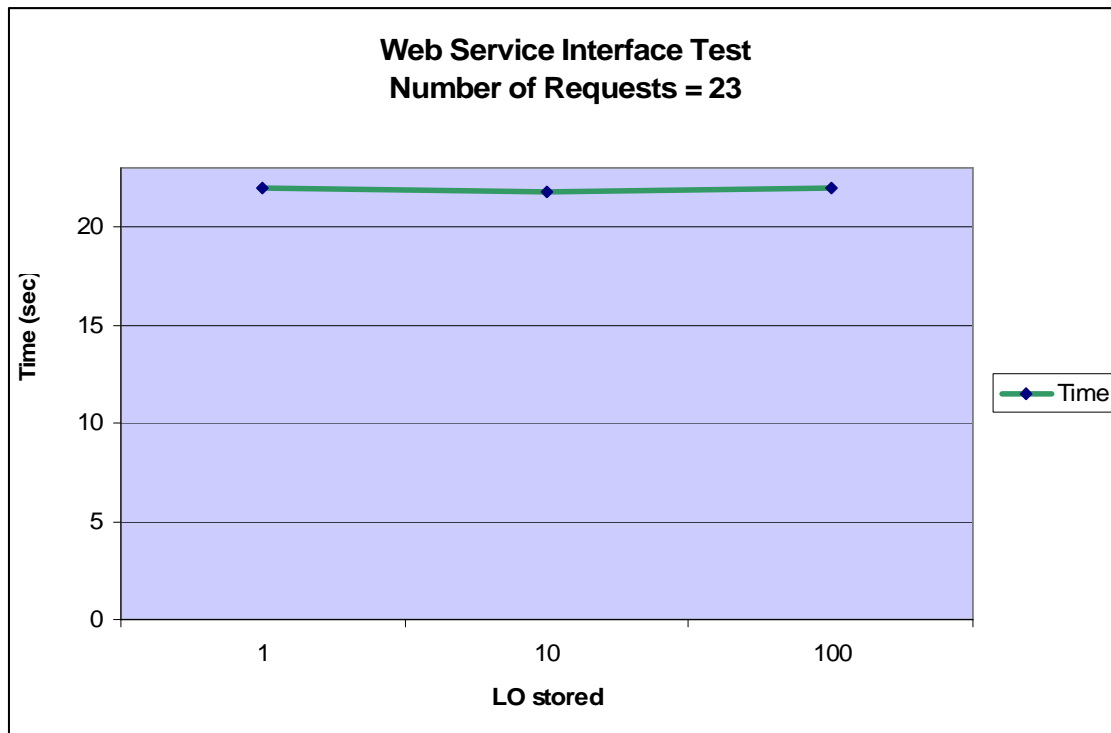
The load test has been performed using the `soapUI` application that is a free and open source desktop application for inspecting, invoking, developing, simulating/mock and functional/load/compliance testing of web services over HTTP. Every test is made up of a concrete number of Java Threads that make requests at all the Web Service operations with preconfigured parameters as the learning object language and the learning object identifier of existing instances in the lom-r repository. The time used in all the test cases is 180 milliseconds and the number of Threads changes in each case from 1 to 125. The computer used for the tests is an Intel Pentium D at 3.00GHz, 2.00 GB of RAM and Microsoft Windows Xp Pro. with SP2 installed.

Every test is made up of a concrete number of Java Threads that make requests at all the Web Service operations with preconfigured parameters as the learning object language and the learning object identifier of existing instances in the lom-r repository. The time used in all the test cases is 180 seconds and the number of Threads changes in each case from 1 to 125.

The computer used for the tests is an Intel Pentium D at 3.00GHz, 2.00 GB of RAM and Microsoft Windows Xp Pro. with SP2 installed.



This graph is the result of combining the two tests exposed above. The resulting time corresponds to the execution of the Web Service tests using different databases. On each database the number of learning object stored fluctuates from 1 to 100.



5.3 Report on extension case

The code skeleton for a hypothetical extension of the LOMR for a particular domain was developed. Concretely, the creation of interfaces for a case on organic agriculture that follow the initial ideas of specializing types of scientific studies and kinds of predication⁵. This was an advance for the foreseen adaptation of the LOMR inside the Organic.Edunet e-content+ project (starting October 1st 2007).

The code skeleton helped in discovering dependency issues in the design of specialized repositories and/or query resolvers that will be included in the final version.

6 DISCUSSION AND OUTLOOK

As a conclusion of the report above, the LOMR architectural prototype has reached an adequate status of maturity to become an initial code base. Further,

⁵ Sánchez-Alonso, S. and Sicilia, M. A.(2007). Using an AGROVOC-based ontology for the description of learning resources on organic agriculture. In Special Session on Agricultural Metadata & Semantics in MTSR'07 - 2nd International Conference on Metadata and Semantics Research.

the use cases planned in the specification of the prototype have been exercised, and additional non-planned functionality has been also implemented.

The analysis of the work so far has led to a plan for the next internal release planned for M22, and also to a list of architectural issues that have still not been exercised and will be a priority for the coming implementation work.

6.1 Pending architectural issues

The following table summarizes pending architectural issues identified during the evaluation of the architectural prototype:

Issue	Description
Management of composed LO that span across several LOMR instances.	The LORef concept in the architecture allows for the management of learning objects composed of parts with metadata residing in different LOMRs. A case of this kind of functionality has not been integrated in the current version of the LOMR.
Integration of WSML reasoning	Integrating WSML reasoning inside the LOMR. The WSMO reasoning framework ⁶ has already been tested, but it is still not integrated inside the code base.

6.2 Plan for the next release

The next release will focus on two main aspects:

1. Extending existing functionality.
2. Adding experimental features.

The extension of existing functionality will cover the following:

- Refinement of IEEE LOM query-by-example functionality.
- Extension of the navigational search capabilities.
- User management integrated in the ontologies.
- Extension of the Web Service interfaces as the requirements of the LUISA use case implementation proceeds.

Experimental features to be explored include:

- Integration with open source collaborative filtering (CF) frameworks, thus allowing search that combined personalized recommendations with classical CF algorithms.
- Integration with tag systems as del.icio.us, based on the experience gained in the prototype developed.
- Integration of WSML-based reasoners.

⁶ <http://tools.deri.org/wsml2reasoner/releases/>