

LUISA

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

IST- FP6 - 027149



Deliverable D4.1 **LOMR overall architecture**

Miguel-Angel Sicilia
Salvador Sanchez
Sinuhe Arroyo
Sergio Martín-Cantero

Due date of deliverable: 31/08/2006

Actual submission date: 19/09/2006

Start date of the project: 01/03/2006

Duration: 30 Months

Miguel-Angel Sicilia
University of Alcalá

Version 1.0, dated 30/07/2006

Change History

Version	Date	Status	Author (Partner)	Description
0.1	31.03.2006	Initial Draft	Sinuhé Arroyo	Table of contents and first ideas
0.2	10.04.2006	Initial Draft	Sinuhé Arroyo	Architectural context
0.3	12.04.2006	Initial Draft	Sinuhé Arroyo	Core technological building blocks
0.4	15.06.2006	Initial Draft	Sinuhé Arroyo	Architectural background
0.5	15.07.2006	Initial Draft	Sinuhé Arroyo	Review
0.6	20.07.2006	Initial Draft	Salvador Sanchez	Object repositories specifications
1.0	30.07.2006	Initial Draft	Miguel Angel Sicilia	Final version for QA

EXECUTIVE SUMMARY

This deliverable describes the overall architecture of the LUISA *Learning Object Metadata Repository* (LOMR), i.e., the high-level components, their interactions, and possible configurations. The goal is to provide an open source reference Service-Oriented Architecture which can be implemented for the flexible learning object discovery, selection, negotiation and composition through the use of Semantic Web Services.

The rationale of our architecture design is to build on existing work, extend it where necessary and be as open as possible. We tried to incorporate as much existing knowledge, frameworks, and architectural proposals as possible to ensure that our architecture is coherent with the ongoing developments in the SWS domain. By these means we look forward to maximize the industrial and academic uptake of LUISA.

ACKNOWLEDGMENTS

The discussions inside the REDAOPA thematic network on learning objects (funded by the Spanish Ministry of Education, action number TS12004-21263-E) were used as a useful input on the technical requirements of what a semantic learning object repository should be.

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	4.1	Title	LOMR overall architecture
Work package	Number	4	Title	Learning Object Metadata Repositories and LOMR Integration Development

Date of delivery	Contractual	01.09.06	Actual	01.09.06
Status	Version X, dated dd/mm/yyyy2006		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 delineates the architecture of the LUISA Learning Object Metadata Repository. The aim of the repository is that of storing and making available through a service-oriented architecture the semantic metadata that will be used for the advanced location, selection and negotiation of learning resources.			
Keywords	Learning objects, learning object repositories, Semantic Web, Semantic Web Services, ontologies, WSMO, IRS-III			

Authors (Partner)	Miguel-Angel Sicilia (UAH), Salvador Sánchez-Alonso (UAH), Sinuhé Arroyo (UAH), Sergio Martín-Cantero (UAH)			
Responsible Author	Miguel Angel Sicilia	Email	msicilia@uah.es	
	Partner	UAH	Phone	+34 91 886 6603

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 Poincaré	UHP 	Dr. Monique Grandbastien University Henri Poincaré 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 – Corporate Research Centre	EADS 	Anne Monceaux EADS – Corporate Research Centre 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
ACKNOWLEDGMENTS	4
TABLE OF CONTENTS	7
1 INTRODUCTION	9
2 ARCHITECTURAL CONTEXT AND MOTIVATION	10
3 BACKGROUND	11
3.1 Semantic Web Service Initiatives	11
3.2 Learning Objects (LO)	11
3.3 Learning object repositories	11
3.4 IMS DRI 1.0	12
Figure 1. IMS DRI function diagram	13
3.5 IEEE LTSA	13
Figure 2. LTSA draft standard	14
3.6 LSAL	15
Figure 3. Learning Service Architecture	15
4 ARCHITECTURAL PRINCIPLES	17
4.1. Goals: Interoperation and Decoupling	18
4.1.1 Interoperation	18
4.1.2 Decoupling	19
4.2 Trade offs	19
5 LOMR MODEL	21
5.1 An IMS DRI view of the LOMR	21
5.1.1 Overview of the IMS DRI specification	21
5.1.2 REQUEST/DELIVER and SUBMIT/STORE in semantic repositories	21
4.1.1. SEARCH/EXPOSE in semantic repositories	22
4.1.2. GATHER/EXPOSE in semantic repositories	22
4.1.3. Resulting architecture of the semantic repository	22
Figure 4. Overall function architecture	23
5.2 Main Architectural elements	23
5.2.1 Principles and commitments for the storage of semantic metadata for LO	23
5.2.2 Functional decomposition	24

5.2.3 LOMR for ontologies	24
Figure 5. LOMR functional view	25
5.2.4 LOMR for instances.....	25
Figure 6. LOMR for instances	26
5.2.5 LOMR for Semantic Web Services.....	27
Figure 7. LOMR for Semantic Web Services	27
6 OVERALL ARCHITECTURAL INSTANTIATION.....	28
6.1 Architecture for LOMR ^o	28
6.2 Architecture for LOMR ^{SWS}	28
6.3 Architecture for LOMR ^{i+o}	28
Figure 8. Architectural instantiation	29
6.3.1 RDF storage.....	29
6.3.2 Interface with the LUISA Annotation tool.....	30
6.3.3 Interface with resource utilizers for ontology search	30
6.3.4 Main Web Service interfaces.....	30
6.3.5 Query resolvers.....	30
6.4 Interfacing with existing repositories	31
CONCLUSION.....	32
REFERENCES	33

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 overall architecture of Learning Object Metadata Repositories (LOMR) as one of the main components of the LUISA architecture. Concretely, this deliverable is the main result of task 4.1 in work package 4 “**LOMR overall architectural specification**: *The overall DRI-compatible specification of the LOMR architecture would incorporate semantics as the key asset in the retrieval functions. This would include the integration of the WSMO technical and conceptual framework*”.

The document is structure as follows. Section 2 presents the architectural context and the motivation that allows justifying the technological paths chosen. Section 3 depicts the background technologies relevant to the objectives of LUISA. Section 4 introduces the main architectural principles driving LUISA. Section 5 carefully depicts the particularities of the LOMR model as far as the IMS DRI and the main architectural components concerns. Section 6 summarizes previous sections by introducing the overall architecture LUISA architecture. Finally, the conclusions of the work and further outlook are detailed.

2 ARCHITECTURAL CONTEXT AND MOTIVATION

The aim of this section is to put into context the LUISA architecture. It provides the background knowledge and perspective that justify the directions and decisions taken.

The main architectural concepts to be used in the LUISA project are, namely, Services, Web Services, Semantic Web Services, Service Oriented Architecture (SOA), Semantic Service Oriented Architecture (SSOA), Enterprise Service Bus (ESB) and Semantic Enterprise Service Bus (SESB).

Given the scope of “D3.1: *State of the art – SWS Infrastructure, Annotation, LCMS*”, also delivered in M6, this section and consequently the deliverable will not address Services aspects, since they are covered in that other deliverable. For specific details on the mentioned architectural concepts see D3.1.

3 BACKGROUND

3.1 Semantic Web Service Initiatives

The Web Service Modelling Ontology (WSMO) [4] and the Internet Reasoning Service - IRS [3] are the main initiatives as far as modeling and executing Semantic Web Services concerns. The LUISA project aims at integrating both approaches as part of a reference SOA. For specific details on WSMO and IRS-III see D3.1.

3.2 Learning Objects (LO)

The evolution of Web-based learning has fostered the search for methods and technologies that enable a degree of recycle of learning contents and learning activity designs. Such attempt is intended to allow for the reuse of quality resources and the development of automated resource-search tools, and it may eventually reduce the costs of devising learning activities. The concept of *learning object* is at the center of the new paradigm for instructional design of Web-based learning that emphasizes reuse as a quality characteristic of learning contents and activities. For example, Polsani [9] includes reuse in his definition of learning object as “*an independent and self-standing unit of learning content that is predisposed to reuse in multiple instructional contexts*”, and Wiley [13] also mentions the term in his learning object definition “*any digital resource that can be reused to support learning*”.

In practical terms, a learning object is a piece of Web content of arbitrary type and structure that is described by a metadata record that provides information about the object and its prospective educational usages. Learning object metadata is thus the key to reuse. The LUISA approach to reusability is that of providing formal metadata expressed in terms of ontologies, combined with the capabilities of a semantics-enabled service-oriented architecture that fosters the discovery of LO providers based on such formal expression of metadata.

3.3 Learning object repositories

Learning object repositories are systems that provide access to collections of learning objects. The major reason for their existence is the fact that online web search engines return too many results. Instructional materials like policy guides, assignments, simulations, websites, tutorials, matrices and other kinds of educational materials are easier to find from within a contained collection. For the purposes of this project and for the rest of this document, we will provide the following definition of a learning object repository:

A learning object repository is a software system aimed at storing educational resources and/or metadata for those resources, which provides search interfaces to humans and/or to other software systems.

In most cases, they are not repositories of learning objects in a strict sense but repositories of links to learning objects, as these systems often store metadata about the learning objects but not the resources themselves. Consequently, it is

possible to find the same resource through different repositories, and repositories can be called *Learning Object Metadata Repositories* (LOMR) instead.

The main functionality of a learning object repository is that of searching for learning objects. This search can either be human-oriented or machine-oriented:

1. Human-oriented search services are implemented as interactive search/browsing interfaces.
2. Machine-oriented search services can be consulted by software agents (e.g. through Web services) and are implemented as software interfaces.

On top of the previous definitions, it is possible to define a semantic learning object repository in the following manner:

A **semantic learning object repository** is a learning object repository that makes use of formal representations of knowledge, in the form of ontologies, to provide enhanced search and retrieval mechanisms to its users.

This is a new term coined in a few cutting-edge research works, such as SLOR ([12], [11]) or UREKA [8].

For a complete state of the art on learning object repositories see D3.1.

3.4 IMS DRI 1.0

Regarding standards, the most remarkable effort in the arena of learning object repositories targets their ability to work with similar systems, an issue that lies on the exchange and interoperability of learning objects metadata. The IMS Digital Repositories Interoperability (DRI) has as its main purpose precisely this issue. The architecture of the IMS DRI **Phase 1** specification version 1.0 [7] aims at “*providing recommendations for the interoperation of the most common repository functions*”.

DRI defines the interactions between core functional components (*resource utilizers* and *repositories*) that support interoperability, including:

- SEARCH, GATHER, (ALERT¹)/EXPOSE
- REQUEST/DELIVER
- SUBMIT/STORE

The DRI Project Group is focusing on these core interoperability functions within the functional architecture. The following functional diagram of the IMS DRI specification depict the core interaction addressed (the rest of the elements are blurred since they are not covered by Phase 1).

¹ ALERT is a core function, but is not addressed within this version of the DRI specification.

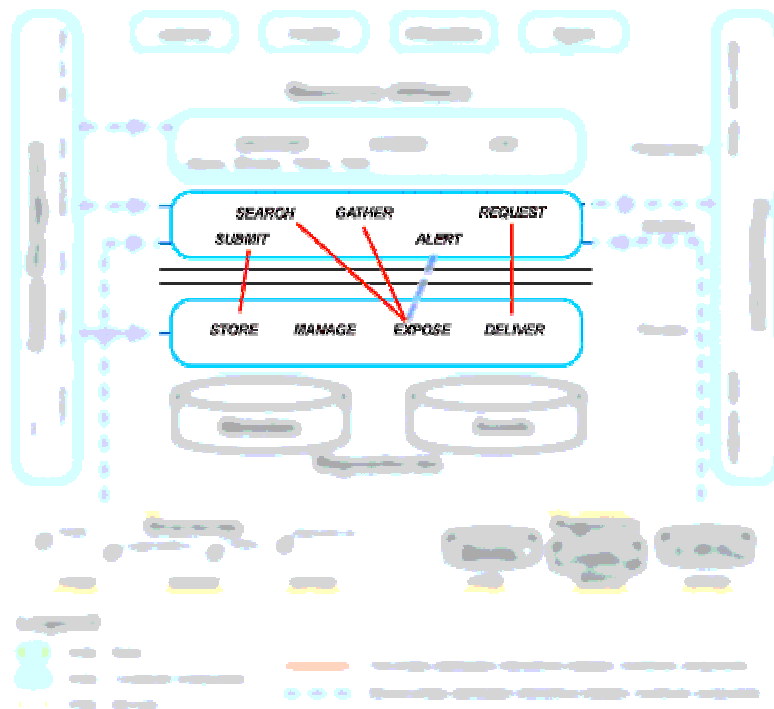


Figure 1. IMS DRI function diagram

3.5 IEEE LTSA

The LTSA draft standard [6] “specifies a high level architecture for information technology-supported learning, education, and training systems that describes the high-level system design and the components of these systems”.

The LTSA architecture is not a design but an analysis model, as stated in its introduction: “An architecture isn’t a blue print for designing a single system, but a framework for designing a range of systems over time, and for the analysis and comparison of these systems, i.e., an architecture is used for analysis and communication”

The normative part of the LTSA draft standard is depicted in the following Figure 2.

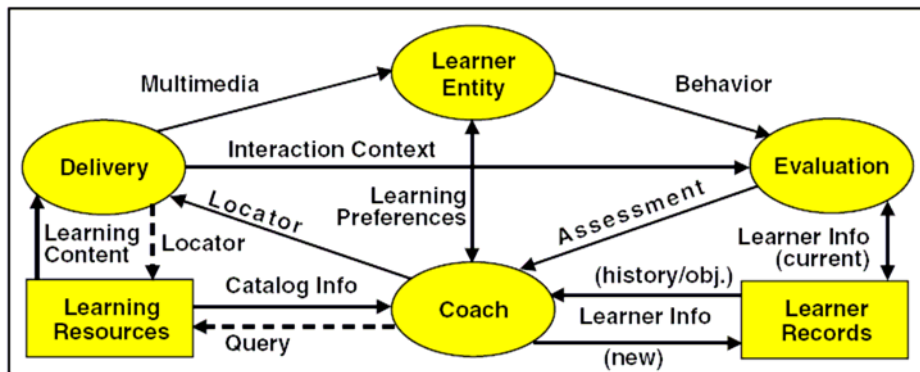


Figure 2. LTSA draft standard

The LTSA system components are:

- Processes: learner entity, evaluation, coach, delivery.
- Stores: learner records, learning resources.
- Flows: learning preferences, behavior, assessment information, learner information (three times), query, catalog info, locator (twice), learning content, multimedia, interaction context.

The overall operation is specified in conceptual terms, stated as follows: “Briefly, the overall operation has the following form: (1) the learning styles, strategies, methods, etc., are negotiated among the learner and other stakeholders and are communicated as learning preferences; (2) the learner is observed and evaluated in the context of multimedia interactions; (3) the evaluation produces assessments and/or learner information; (4) the learner information is stored in the learner history database; (5) the coach reviews the learner’s assessment and learner information, such as preferences, past performance history, and, possibly, future learning objectives; (6) the coach searches the learning resources, via query and catalog info, for appropriate learning content; (7) the coach extracts the locators from the available catalog info and passes the locators to the delivery process, e.g., a lesson plan; and (8) the delivery process extracts the learning content from the learning resources, based on locators, and transforms the learning content to an interactive multimedia presentation to the learner”

The above description can be easily mapped to “learning scenarios” as identified in LUISA. However, they do not entail any commitment to particular technologies or architectural concepts from the viewpoint of *software design*.

The conceptual nature of the LTSA specification can be mapped to elements in learning scenarios, but does not have implications for the technical architecture described in this document.

3.6 LSAL

The LSAL Learning Services Architecture (LSA)² has already provided a layered view of a service-oriented learning technology infrastructure. The following Figure 3 depicts the main layers.

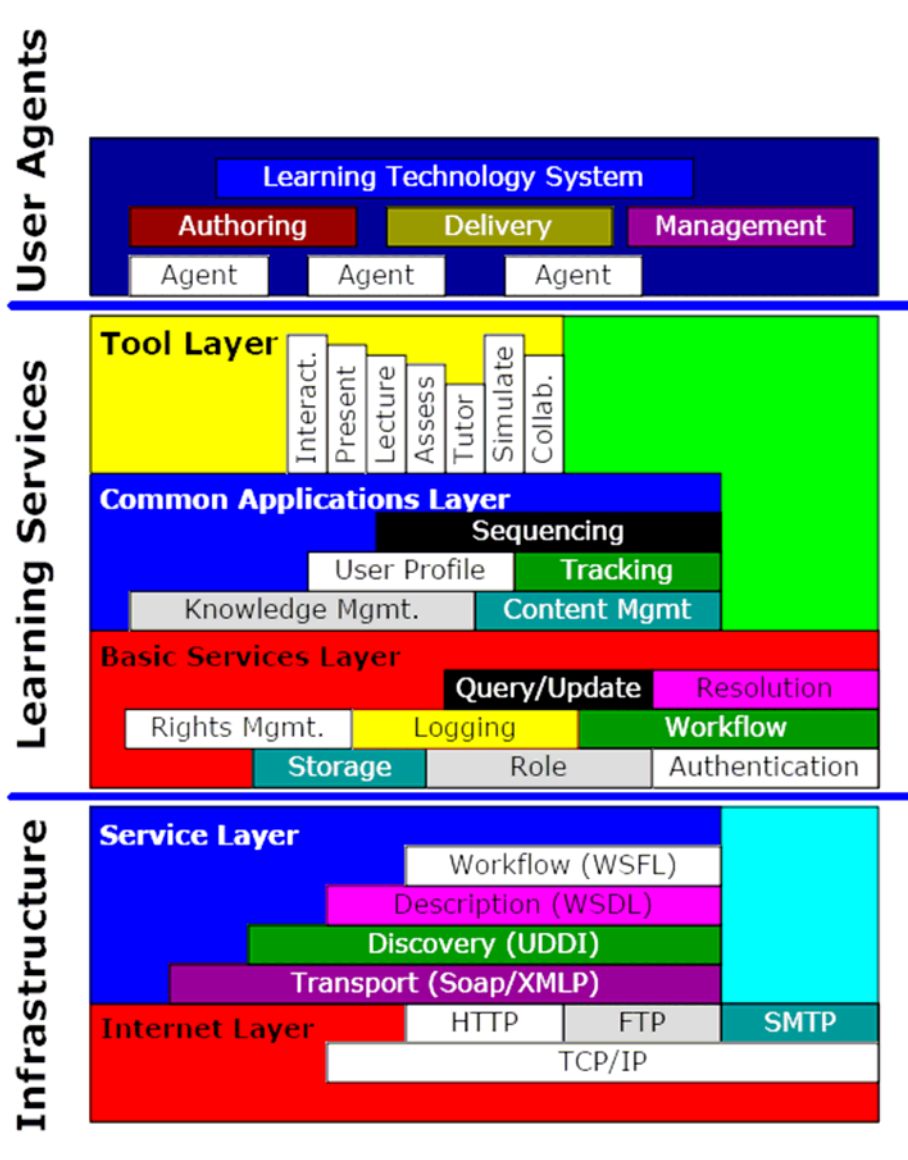


Figure 3. Learning Service Architecture

The “Infrastructure” layer of the LSA includes WSFL as a “workflow” component.

² <http://www.lsal.cmu.edu/lsal/expertise/technologies/learningservices/>

However, WSFL has been superseded by BPEL, an orchestration language that does not make use of semantic but syntactic, static matching of the services to be called.

The emphasis of LUISA is on that “workflow” layer, understood as “choreography and orchestration”. The LSA architecture introduces a notion of “global supervision” that can be assimilated to the notion of BPEL engine: “In addition to the localized workflows, there must be an *overall monitor process*. The top level supervisory control is simply a user monitor that waits for user requests and passes them along to the appropriate services for processing. The overall processor must, however, *maintain global state control, handle errors and service failures, manage client interface agents*, etc. The overall processing is also described by a set of control rules, applied to the system”.

The LSA in its current state does not go further in defining learning technology scenarios and specifying how the “Tool service” layer is supported by the Workflow layer.

4 ARCHITECTURAL PRINCIPLES

Based on the ideas, specifications and initiatives presented in previous sections, a number of principles that summarize the goals of the LUISA architecture can be sketched. In the following these principles are enumerated and briefly depicted.

Interoperability. Being a core architectural building block, services have a very dynamic and heterogeneous nature. On the one hand, they are added, enhanced, renovated and replaced in short periods of time. On the other hand, their vendor and functionality are varied. Still, heterogeneous services need to coexist and closely interact, requiring to take all these aspects under close consideration.

Autonomy. The different building blocks of the architecture need to be able to perform in an autonomous fashion. In this sense they are envisioned as black boxes, which require some inputs and provide some outputs, making its boundaries explicit. By achieving autonomy coupling is reduced, as dependencies among components are made explicit. The concept of autonomy can be extended to cover the independent design, implementation, deployment and maintenance of services.

Decoupling. Decoupling implies abstracting dependencies away. It can be envisioned as an integration prerequisite, which supports the dynamic and heterogeneous nature of services. It applies at two different levels. From a fine-grained point of view, the services that define the conceptual framework need to be well decoupled from each other. From a coarse-grained point of view, the whole architecture needs to be decoupled from the general execution environment, where it is going to be integrated. In both cases, decoupling allows picturing the different building blocks as independent interacting pieces of functionality.

Message-based. Message-based interactions are characterized by the exchanging of complete, self-contained business documents over the wire. Scalable and reliable architectural decoupling is based on the asynchronous exchange of messages [2], rather than synchronous RPC-based interactions, which strengthen dependencies among parties. Thus, services need to expose public message interfaces descriptions [1] that allow the document trade in a loosely coupled asynchronous fashion.

Standard-based. Services must interface seamlessly with each other, regardless of their degree of sophistication. Different implementations of a standard diverge from the original specification by extending features or making available only a subset of the functionality. Even though the use of standards does not completely solve the interoperation problem, they help to narrow and favour true interoperation. Standards should be applied at all possible different levels of the conceptual model as long as they do not force a technological choice.

4.1. Goals: Interoperation and Decoupling

Given LUISA will be realized as a Semantic Service Oriented Architectures (SSOA), the architectural challenges can be synthesized into two conflicting aspects: interoperation and decoupling. It is the combination among interoperation and decoupling that introduces the major difficulties in designing SSOA.

4.1.1 Interoperation

Interoperation allows a service to communicate with others independently of its implementation or its implementation technology. These are the most significant requirements subsumed by interoperation:

- **Technical heterogeneity and mediation:** Services are implemented following different technological paradigms. They are usually self-developed or acquired following the “best-of-breed” approach [2]. This implies having a number of heterogeneous services that need to interoperate. To do so, it is required to make minimal assumptions on the capability of each component or its conformance to standards and communication protocols.
- **Semantic heterogeneity:** Services are built to work in isolation. This implies having a number of different services, coming from different vendors and using different conceptual models to express the business process semantics. The meaning of the business processes needs to be mapped in order not to change semantics.
- **Syntactic heterogeneity:** Services interoperate by exchanging data. They make use of different document types, representation formats, standards, communication patterns. The syntactic representation of the data exchanged needs to be mapped in order to allow interoperation. Further, patterns need to be mediated, to successfully conduct communication among heterogeneous parties.
- **Openness:** Deals with the ability of services to work in an open environment. It allows any service to interoperate with a similarly enabled one. The aim is to avoid prior discussions regarding syntactic or semantic details and patterns. Further, the process should be rather automatic, avoiding manually customizing and implementing each party.
- **Autonomy:** Each service change states without consulting other services [2]. As far as it is relevant for the mutual interaction, state changes affecting the external behaviour of services should be observed and communicated in order to satisfactorily conduct the required message exchange.
- **Interface:** Specify the way to access a service. It provides a layer that allows services to interoperate with each other, clearly defining what is “inside” and what is “outside” them. Interfaces provide the means to communicate among services. Furthermore, they allow hiding

implementation details, making technology matters transparent to other services.

4.1.2 Decoupling

Decoupling or loose coupling refers to the degree of mutual dependency among components. It allows services, its message interfaces and message structures to be deployed, modified, or replaced without necessarily affecting all of the other components that communicate or share data with them. Decoupling translates into the following requirements:

- **Indirect binding:** Services are not directly bonded to each other. Instead they bind to the messages that services produce and consume. The only common link among services is the vocabulary they share. This does not necessarily hold in the case of heterogeneous services in an open environment. The absence of predefined types systems increases decoupling.
- **Abstract processes:** Messages are never directly exchanged among services. Instead they are directed to intermediary correlating services [2] which are responsible for overcoming interoperability issues and dispatching the message to the appropriate service.
- **Coarse grained:** Services are spread over a network communicating remotely. In order to reduce the network overhead and eliminate dependencies all the required information to conduct and action is contained in the message.
- **Interfaces:** Services expose message-based interfaces hiding the internal details of the application logic. Thus, any modification in the application logic does not require further modification in the code of the components using the message-exchange interface. In order to favour decoupling, interfaces need to be:
 - Extensible: Permitting messages to evolve over time without forcing both sides into a complex upgrade.
 - Evolvable/Versionable: Allowing for compatible changes in messages. It provides for the understanding of old messages by new receivers and reciprocally, the consumption of new messages by old receivers.

4.2 Trade offs

From what we listed above, it is quite clear that both interoperability and decoupling pose significant difficulties if requested singly, even more when the combination of both is required. However, satisfactory results of the software architecture, EAI and Semantic Web research solve some of the problems related to the technological, syntactical and semantically heterogeneity respectively. Similarly, as far as decoupling is concerned, message-based techniques have been developed to cope with reliable service communications. It should be also clear that interoperability and decoupling conflicting features. In fact, techniques that optimize the interoperation of heterogeneous services

usually strengthen the dependencies among them, not being suitable for open environments. On the other hand, highly decoupled services with rich interfaces and messages exchanges are better realized with homogeneous services where semantic and syntactic differences are minimal, placing them, in any case at the message and document processing level. The real challenge for the architectural design of LUISA is to offer an expressive interface while assuring openness and autonomy at a syntactic and semantic level, especially given that clients need to interact with number heterogeneous services.

5 LOMR MODEL

In the following the particularities of the LOMR model are depicted.

5.1 An IMS DRI view of the LOMR

This section provides an overview of the relationships of the IMS DRI specification with the Semantic view of the LUISA project.

5.1.1 Overview of the IMS DRI specification

The IMS DRI **Phase 1** specification version 1.0 defines the interactions between core functional components (*resource utilizers* and *repositories*) that support interoperability.

The IMS DRI 1.0 functional architecture and recommendations are mainly oriented to the interchange and storage of digital contents in IMS formats as mandated by other specifications like the *IMS Content Packaging* specification³. However, the specifics of Semantic Web technologies emphasize on formal metadata and semantic discovery, which is the key point that is analyzed in this document. In what follows, the main DRI functions are examined in that direction.

5.1.2 REQUEST/DELIVER and SUBMIT/STORE in semantic repositories

REQUEST/DELIVER is defined in IMS DRI in the following fashion: “*The Request functional component allows a resource utilizer that has located a meta-data record via the Search (and possibly via the Alert) function to request access to the learning object or other resource described by the meta-data. Deliver refers to the response from the repository which provides access to the resource*”.

For SUBMIT/STORE, the IMS DRI specifies that “*In the case of a learning object repository that is compliant with the DRI specification, the Submit function shall be satisfied through the transmission of an IMS-compliant Content Package using SOAP Messages with attachments*”. The Storage function is a simple representation of “*the ability of the repository to present an IMS Content Package at some level of its operation*”.

Since our model for semantic learning object repositories focuses on the storage of (semantic) metadata, REQUEST/DELIVER and SUBMIT/STORE will be a normal IMS DRI function, and it is thus out of the scope of LUISA. Note that IMS DRI 1.0 allows for the definition of metadata-only repositories: “*Repositories may hold actual assets or the meta-data that describe assets*”.

However, the SUBMIT function is actually a metadata-transfer function, since IMS packages usually contain metadata descriptions for learning objects. This raises the need for two elements in the architecture:

³ <http://www.imsproject.org/content/packaging/>

- An **import/export facility** from/to IMS metadata to the semantic representation of metadata.
- An extended function specific to the storage of semantic metadata. A basic function **ASSERT** is defined for this purpose.

4.1.1. SEARCH/EXPOSE in semantic repositories

“The Search reference model defines the searching of the meta-data associated with content exposed by repositories. Compatibility of SEARCH/EXPOSE in semantic repositories must be provided by some kind of mediation layer. This raises the need for additional elements:

- A **query mediator**, which takes as input either Z39.50 or XQuery queries and transform it to a search in the internal format of the semantic repository.
- A **SEMANTIC-SEARCH** function to directly search in semantic terms.

4.1.2. GATHER/EXPOSE in semantic repositories

The IMS DRI states: “*The Gather reference model defines the soliciting of meta-data exposed by repositories and the aggregation of the meta-data for use in subsequent searches, and the aggregations of the meta-data to create a new meta-data repository*”. The aggregations of metadata are to be handled by the same import/export facility described above. However, the soliciting in “Pull” mode requires an additional consideration. Concretely:

- A **gather engine** with the added functionality of invoking the corresponding import/export facility and using the above mentioned import/export facility to bridge the semantic gap.

Semantic gather engines with functionality extended from the recommended OAI functions could be devised, but they are left for a future investigation.

The “Push” gather does not require any further functional element in the architecture.

4.1.3. Resulting architecture of the semantic repository

The following diagram depicts the overall functional architecture resulting from the analysis of the IMS DRI specification (only the relevant portions of the DRI 1.0 functions are depicted).

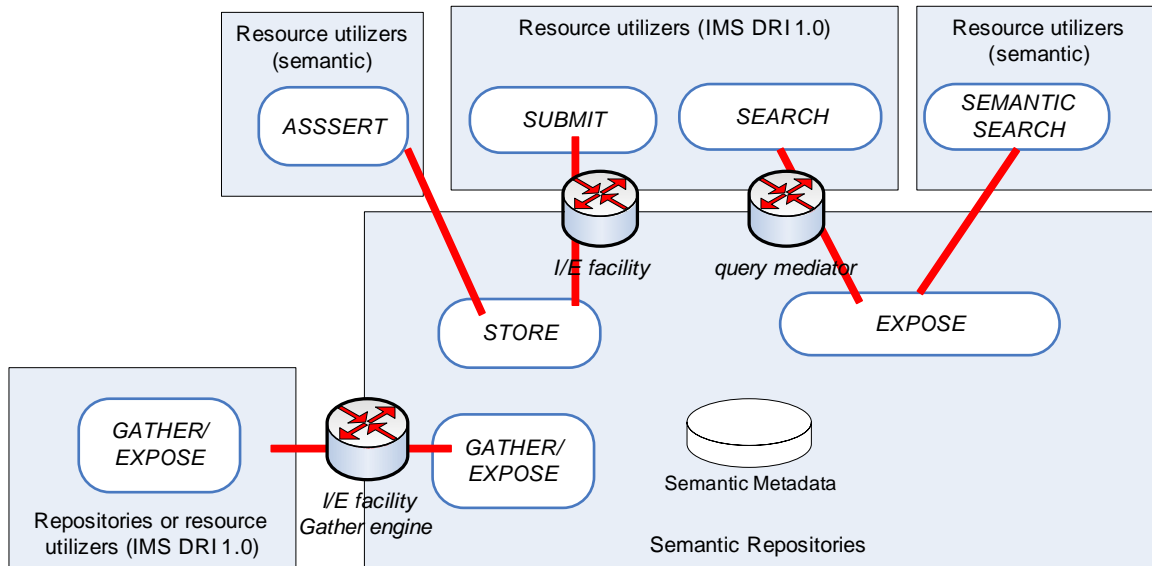


Figure 4. Overall function architecture

The conventional search, gather and submit functions require specific components to bridge from the non-semantic to the semantic representation. However, the current recommendations of the IMS DRI for query do not represent mandatory requirements and are not specific of learning objects. This entails that focusing on the I/E facilities is the key aspect to be considered, and this will be addressed by building translators from learning object metadata standards to the internal ontology-based representation language of the LUISA LOMR.

5.2 Main Architectural elements

In the following the main architectural elements are carefully depicted.

5.2.1 Principles and commitments for the storage of semantic metadata for LO

Semantic repositories of LO are only justifiable as an improvement over existing representational mechanisms. This entails a number of principles for design that have resulted in essential commitments for the architecture of the LUISA LOMR. The following Table summarizes them.

#	Principle	Commitment
1	The semantic LOMR must provide advanced search capabilities based on computational semantics	The LO metadata will be stored in WSML to preserve the semantic capabilities that will be used for the process of selecting LO based on learning needs.
2	The semantic LOMR must be able to work with existing metadata.	A translation approach as the one described in the previous section will be provided, in which external metadata can be gathered and converted to a semantic

		be gathered and converted to a semantic representation without loss of information.
3	The LUISA architecture must work with a variety of possible LOMR capabilities.	There will not be a single definition of the query interface for the LOMR, but only general directions that are compatible with different query styles.
4	The LUISA architecture is service-oriented	The query interfaces for LO will be implemented as Web Services.

It should be noted that principle #1 do not prevent existing non-semantic metadata exposing their contents through Web Services to be integrated into LUISA, but the focus of this document is on delivering LO according to semantic descriptions.

5.2.2 Functional decomposition

The LOMR inside LUISA can be considered to have three different essential functions of different importance in the framework of the LUISA architecture. We will call the three as follows:

- LOMR^o: LOMR for ontologies. It is responsible for the storage of ontologies that describe the learning resources, be them domain ontologies or ontologies that describe learning technology standards or pedagogical properties.
- LOMRⁱ: LOMR for instances. It is responsible for the storage of ontology instances representing the LO and any associated information.
- LOMR^{SWS}: LOMR for Semantic Web Services. It is responsible for the storage of the SWS descriptions of LOMR repositories.

It should be noted that the distinction does not entail different physical entities, but one of them could implement the three in the same process. However, the differentiation enables providing separate implementations.

The difference between “ontologies” and “instances” used in this document does not refer strictly to the formal distinction between “concepts” and “individuals” in ontology languages, but to the differentiation between the representation of concrete LO and their descriptions (*instances*) and the elements that describe them (e.g. an ontology that represents the IEEE LOM schema).

5.2.3 LOMR for ontologies

This part of the LOMR is required for end user applications to expose parts of the ontology(-es) that are used to formulate learning goals of a diverse type. The interfaces that can be provided for that purpose are of two kinds:

- Generic interfaces for querying ontology structures. These could be implemented in any general-purpose ontology access or query language.

- Domain or pedagogy-specific interfaces. These require a specific design that is application or domain-specific.

Figure 5 depicts the functional view of this part of the LOMR.

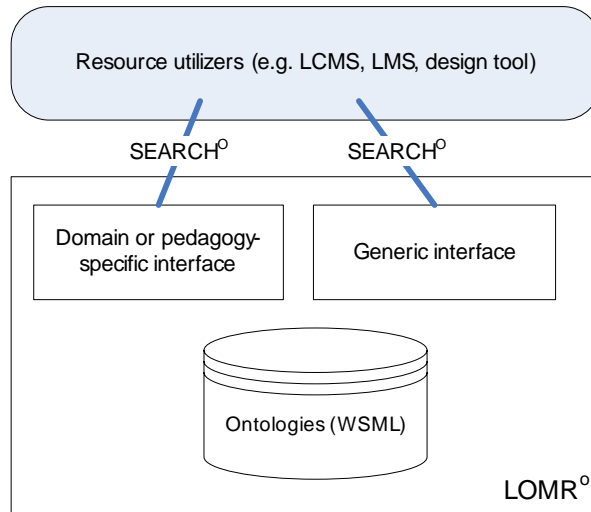


Figure 5. LOMR functional view

In addition to resource utilizers, other parts of the LUISA architecture might require the retrieval of these ontologies for internal use. Notably, the SWS engine of LUISA will require them for their internal working. However, a simple import of the whole ontology is enough for that functionality, or even ontologies stored simply in persistent text file formats can be used.

In any case, this part of the LOMR does not require a Web Service interface, since the advanced SWS capabilities of LUISA are only applied to services providing LO. The interfaces SEARCH[°] for the LUISA project will be decided depending on the requirements of the case studies.

5.2.4 LOMR for instances

This part of the LOMR provides the core function of maintaining the representations of the LO that are the material for the SEARCH functions that are the basis for the LUISA functionality of location, composition and negotiation of LO.

There are several aspects that need to be considered in this part of the repository. Starting with the interface to other systems or to LUISA resource utilizers, the following elements appear:

- Resource utilizers of any kind may invoke the WS interface of the LOMR to retrieve appropriate LO. According to principle #3 above, there will not be a single semantic query interface, since the expression of learning needs is subject to many approaches and styles.
- The metadata in external repositories will be imported through a flexible translation approach. This will be discussed later in this document.

- The Translator/Federator/Aggregator patterns described in the IMS DRI can be implemented in the level of the LOMR. However, these will not be part of the reference implementation since they are covered architecturally by the SWS framework that is the focus of LUISA. Concretely, Translation can be provided through mediation, and the Federation/Aggregation services can be provided through the flexible decoupled discovery mechanisms of WSMO.

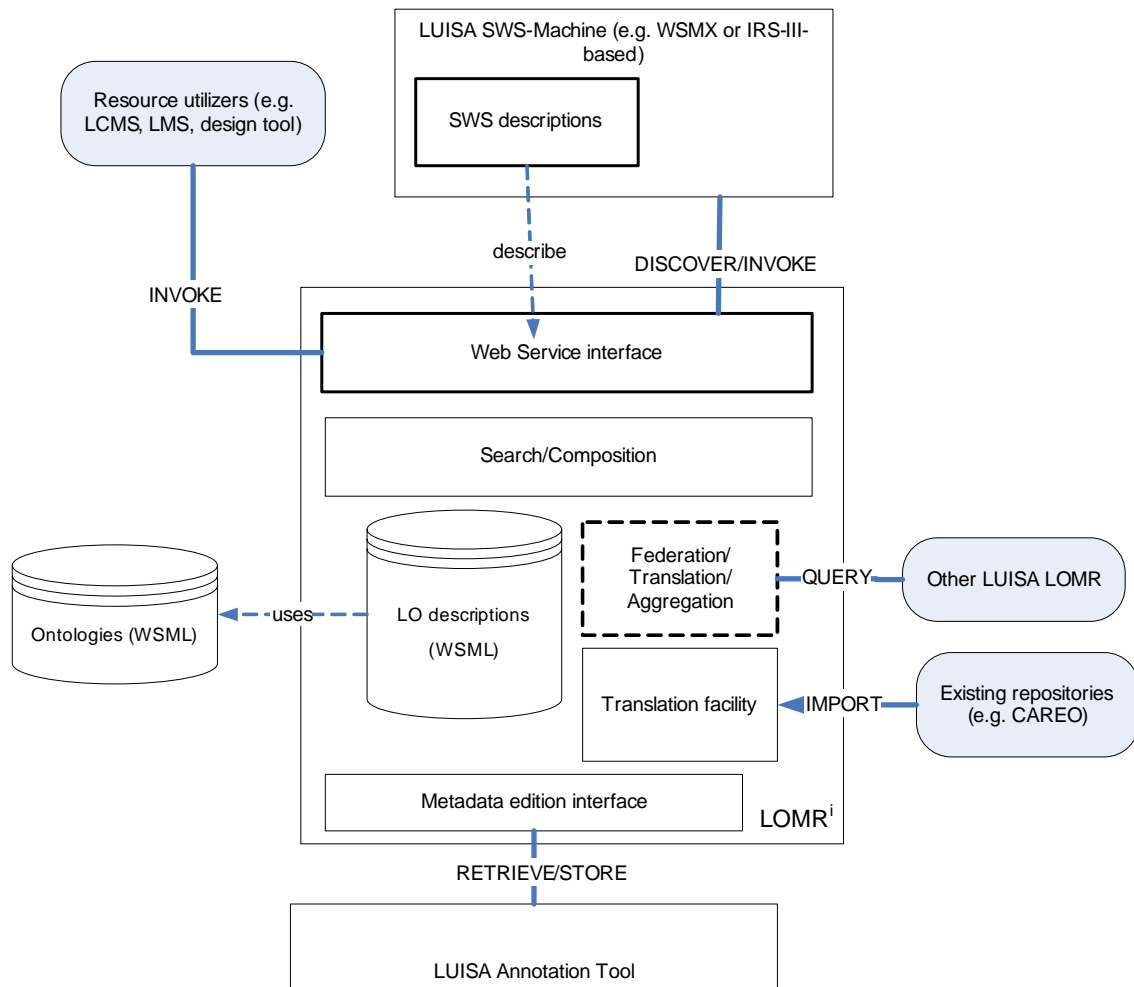


Figure 6. LOMR for instances

The LOMR will provide a Web Service interface for the search of LO according to some expression of given needs. The search in the “local” base of LO metadata could in some cases be complemented by composition mechanisms. However, composition can also be handled in a decoupled way by combining semantic searches in the SWS framework, which makes the composition facilities eventually provided by a LOMR transparent to the SWS functioning.

The Web Service interfaces of the LOMR will be described semantically, and these descriptions are the basis for their discovery by the SWS-Machine of

LUISA. Once the appropriate LOMR Web Services are found, resource utilizers will typically query them, or in some cases the SWS Machine will do the query to provide some additional processing on the results – e.g. when aggregating the results of LO queries.

Finally, the LOMR requires the provision of a specialized interface for metadata edition, focusing on the efficiency of the retrieval and storage of metadata.

5.2.5 LOMR for Semantic Web Services

This part of the LOMR deals with the storage of the descriptions of the SWS, which are dependant on the ontologies describing the learning resources. The rationale for the independent storage of these descriptions is that of providing persistence to SWS before they are instantiated inside SWS frameworks as IRS-III or WSMX. However, this facility is not linked to the core LUISA functionality, which deals with search, composition and retrieval of the LO through these services.

In any case, the development of LOMR for SWS instances as an intermediary between the SWS engines and the resource providers could be considered an interesting extension to the LUISA architecture. The following Figure depicts the main functional elements of the LOMR for SWS, in which a given SWS engine retrieves SWS definitions for the purpose of having their descriptions available for the discovery of WS. Further, some kind of SWS facility could be provided in the future so that some SWS engines could select the kind of services they will install for their functioning.

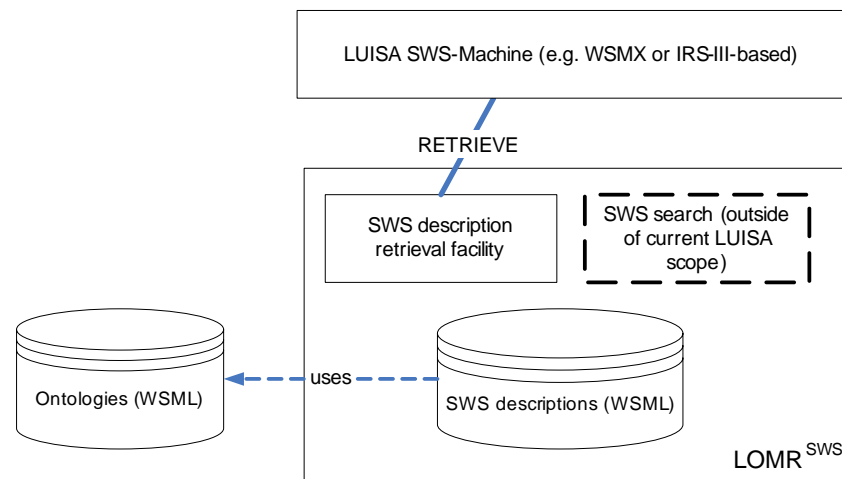


Figure 7. LOMR for Semantic Web Services

6 OVERALL ARCHITECTURAL INSTANTIATION

This section provides the description of the proposed LUISA LOMR overall technical architecture, according to the functional perspective provided above.

6.1 Architecture for LOMR^o

The LOM^o for the LUISA project will be combined with the instance repository, since the LO description instances are tied to the ontology descriptions. Domain-specific interfaces could be delivered for each Use Case, as a convenience facility for the implementation of LCMS extensions.

6.2 Architecture for LOMR^{SWS}

This part of the LOMR will not be provided as is in the LOMR implementation since it is redundant with the capabilities of current WSMO implementations as IRS-III and WSMX. Eventually and for convenience reasons in the publication of SWS, the provision of a straightforward SWS definition import facility will be provided.

6.3 Architecture for LOMR^{i+o}

The architectural instantiation for the LOMR combining the ontology and instance aspects is summarized in the following diagram⁴. The following subsections detail each of the aspects of this overall picture.

⁴ It should be noted that this architectural instantiation is tentative, since progress in the specification of other elements of the LUISA architecture may lead to a different decision in terms of interfaces.

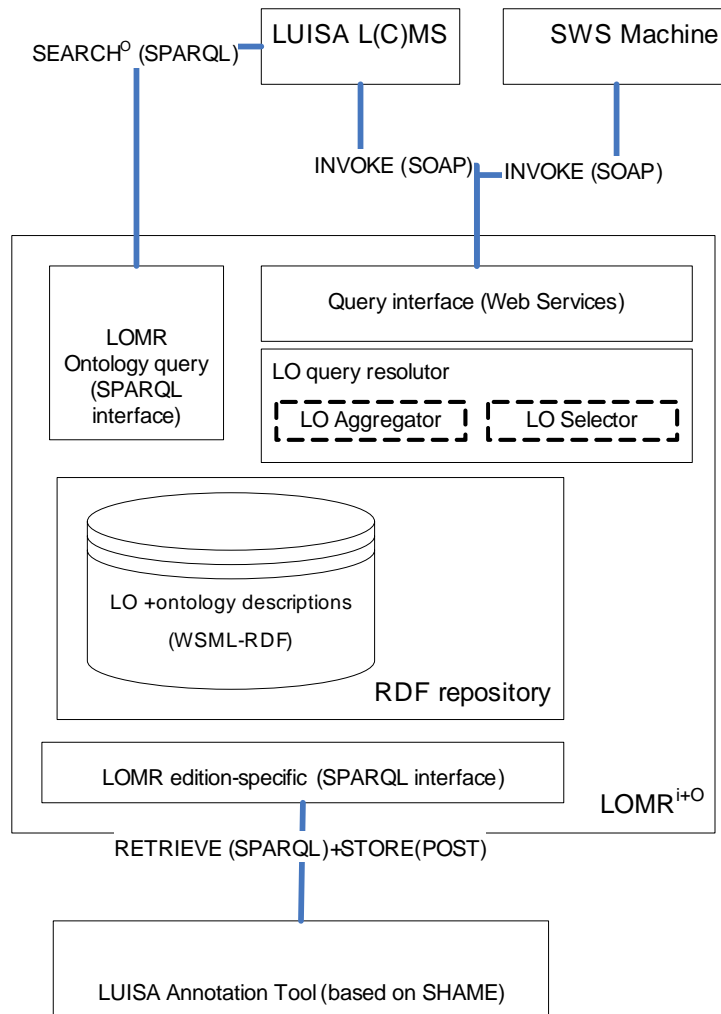


Figure 8. Architectural instantiation

6.3.1 RDF storage

The actual metadata and ontologies will be stored in RDF format, thus enabling the use of mature RDF servers that have been used in several projects to deal with ontology data. The two main technological options being considered are HP Jena⁵ and Sesame⁶. Both comply with the constraints of the architecture and provide several query languages in addition to programmatic interfaces to perform navigational search in the ontologies and their instances.

The storage of WSML in RDF format can be achieved by using the parsing and translation facilities available in the WSMO for Java project⁷, which have been yet tested in LUISA with both Jena and Sesame.

⁵ <http://jena.sourceforge.net/>

⁶ <http://www.openrdf.org/>

⁷ <http://wsmo4j.sourceforge.net/>

6.3.2 Interface with the LUISA Annotation tool

An interface specific to the LUISA Annotation tool (which is based on the SHAME open source tool developed by ULL) is required to deal with the specific setting of human edition. The interface needs to be decoupled to allow for the location of the LOMR and the Annotation tool in different nodes.

The SPARQL⁸ proposed W3C standard complies with the basic constraints of the architecture. The Jena RDF repository can be queried through SPARQL through the Joseki⁹ server, and there is an ongoing implementation of SPARQL compatible with Sesame¹⁰. Dealing with metadata at the RDF level is coherent with the technology inside SHAME, which is essentially a configurable and flexible RDF editing tool.

6.3.3 Interface with resource utilizers for ontology search

Resource utilizers will need to query the ontologies to help users with user interfaces in formulating the learning needs. A SPARQL interface can be used as a generic query facility, and specific interfaces on top of such generic interface will be used if required by the use case implementations.

6.3.4 Main Web Service interfaces

The principal LO retrieval interfaces will be provided in the form of standard SOAP-based Web Services that will be described semantically and invoked either by resource utilizers or by the LUISA SWS Machine. Since the LOMR in LUISA only stores metadata, the query results of those Web Services must return URIs that could be used for the retrieval of the actual resources.

6.3.5 Query resolvers

Just as the LUISA approach does not constraint the kind of LO query services provided by different repositories, a variety of *query resolvers* with different capabilities can be provided. A straightforward query resolver could use SPARQL or other RDF-querying language for the retrieval of the appropriate LO, but more complexity can be implemented in resolvers. Concretely, some of them could have some built-in LO aggregation capabilities, thus eventually creating new instances of LO inside the RDF repository. Further, some other could be specific to some kind of pedagogy or provide some decision procedures beyond the mere matching of RDF descriptions – e.g. some could consider specific parts of the ontologies or use reasoning services in their internal working.

In any case, this component does not have an overall, universally applicable form, since a broad variety of advanced approaches could be implemented.

⁸ <http://www.w3.org/TR/rdf-sparql-query/>

⁹ <http://www.joseki.org/>

¹⁰ <http://sparql.sourceforge.net/>

6.4 Interfacing with existing repositories

This is a fundamental component of the LUISA architecture, since it enables the reuse of the existing investment in learning resource collection. The approach for the LUISA LOMR is a *translation-based* approach. The main underlying idea on this approach is that existing repositories provide LO metadata in non-semantic languages, as for example, the XML mapping of the IEEE LOM standard. The idea of translation is that of enriching such kind of information to enable computational semantics.

Thus, the LUISA LOMR provides translation facilities that bridge the gap between the non-semantic and the ontology-based representations. Essentially, translators convert metadata expressed in plain XML formats to representations in which learning objects are explicit instances of a *LearningObject* concept, and the descriptions thus become attributes – or in general terms – statements about the resource – see [5] for a general description of the approach.

At the time of the writing of this document, and early implementation of a LOM-XML to WSML translator is available [10]. The translator approach will be further described in the following deliverables¹¹:

- Deliverable D4.9: *A LOM-based ontology of learning objects in WSML*. Describes the mapping of LOM metadata elements to a semantic representation in WSML.
- Deliverable D4.10: *LOM XML to WSML Translation*. Describes the details of the translator implementation from LOM XML to WSML.

The building of translators with a degree of flexibility in their design enables adapting existing and future non-semantic repositories in which new non-semantic descriptions can be added incrementally to semantic representations inside the LOMR.

¹¹ These deliverables are not present in the original Project plan, they are supplementary material.

CONCLUSION

In this deliverable, the overall architecture of the LUISA LOMR component has been described. A functional structure has been provided which separates the three main concerns of the LOMR: ontologies, instances and semantic web service descriptions. The functional view of these three aspects has been described, based on a previous assessment of the relationship of the LOMR with the IMS DRI specification.

The functional view on the LOMR has been put in concrete, physical terms by providing an architecture instantiation that relies on a central RDF storage engine, which provides interfaces to the main other components of the LUISA architecture: the Annotation Tool, the final resource utilizers as the LUISA LCMS, and the SWS framework. The technical decisions made in the project have been discussed providing the required directions for the implementation of LUISA LOMR-compliant software.

REFERENCES

- [1] Alonso, G., Casati, F., Kuno, H., Machiraju, V.: *Web Services. Concepts, Architectures and Applications*. Springer-Verlag, 2003.
- [2] Bussler, C.: "B2B Integration - Concepts and Architecture", Springer-Verlag, 2003.
- [3] Domingue, J., Cabral, L., Hakimpour, F., Sell, D. and Motta E. IRSIII: A platform and infrastructure for creating WSMO-based Semantic Web Services. In *Proceedings of the Workshop on WSMO Implementations (WIW 2004)*, Frankfurt, Germany, September 2004. CEUR.
- [4] Roman, D., Keller, U., Lausen, H., de Bruijn, J., Lara, R., Stollberg, M., Polleres, A., Feier, C., Bussler, C., and Fensel, D.: *Web Service Modeling Ontology*, *Applied Ontology*, 1(1): 77 106, 2005.
- [5] Sicilia, M.A. and García-Barriocanal, E. (2005) On the Convergence of Formal Ontologies and Standardized e-Learning. *Journal of Distance Education Technologies* 3(2), 2005, pp. 13-29.
- [6] IEEE (2001). IEEE P1484.1/D9, 2001-11-30 Draft Standard for Learning Technology — Learning Technology Systems Architecture (LTSA).
- [7] IMS DRI (2003) IMS Digital Repositories Interoperability Core Functions Information Model, Version 1, Available: <http://www.imsqlobal.org/digitalrepositories/>.
- [8] Ihsan, I., Mobin-uddin, A., Mohib-ur, R., Abdul Qadir, M. and Iftikhar, N. (2006) Semantically Meaningful Unit – SMU; An Openly Reusable Learning Object for UREKA Learning-Object Taxonomy & Repository Architecture – ULTRA. In *Proceedings of the 4th ACS/IEEE International Conference on Computer Systems and Applications*.
- [9] Polsani, P. R. (2003). Use and Abuse of Reusable Learning Objects. *Journal of Digital information*, 3(4).
- [10] Rodriguez, E., Sicilia, M.A. and Arroyo, S. (2006) Bridging the semantic gap in standards-based learning object repositories. In *Proceedings of the Workshop on Learning Object Repositories as Digital Libraries: Current challenges*, part of the 10th European Conference on Digital Libraries, Alicante, Spain.
- [11] Sicilia, M.A., Sánchez-Alonso, S. and Soto, J. (2006) Designing Flexible Learning Object Repositories: Balancing Flexibility and Delegation in Ontological Characterizations, in *Principles and Practices of the Effective Use of Learning Objects*, edited by Informing Science Institute.
- [12] Soto, J., Sánchez-Alonso, S. and Sicilia, M. A. (2005) Flexibility in semantic learning object repositories. In *Proceedings of the Third International Conference on Multimedia and Information & Communication Technologies in Education*, 124-128.
- [13] Wiley, D. A. (2001). *The Instructional Use of Learning Objects*. Association for Educational Communications and Technology, Bloomington.