Universidad de Alcalá

Departamento de Ciencias de la Computación

Tesis Doctoral

Propuesta de una Arquitectura para la Búsqueda Federada y Agregación Semántica de Objetos de Aprendizaje en Repositorios Distribuidos

Autor:
D. Antonio Ortiz Baíllo

Directores:
Dr. D. José Ramón Hilera González
Dr. D. Salvador Otón Tortosa

2009

Universidad de Alcalá

Dpto Ciencias de la Computación
Campus Universitario. ETS Ingeniería Informática
Crta Barcelona km 33.600
28871 Alcalá de Henares (Madrid)
Teléfonos: 91 885 66 45 Fax: 91 885 66 46

Una vez concluido el trabajo de tesis doctoral titulado "Propuesta de una arquitectura para la búsqueda federada y agregación semántica de objetos de aprendizaje en repositorios distribuidos", realizado por D. Antonio Ortiz Baíllo, del que soy director, estimo que dicho trabajo tiene suficientes méritos teóricos, que se han contrastado adecuadamente y son altamente novedosos.

El trabajo realizado presenta una exhaustiva descripción de una arquitectura software basada en servicios para publicar y localizar objetos de aprendizaje en repositorios distribuidos a través del uso de técnicas semánticas, para de esta forma poder integrarlos en un sistema de aprendizaje. Se han descrito las capas en los que se estructura la arquitectura, sus componentes ó elementos estructurales, sus propiedades visibles y su funcionalidad. La arquitectura propuesta asegura la interoperabilidad de distintos repositorios de objetos de aprendizaje y la reutilización de sus contenidos docentes. Para el desarrollo de la arquitectura se han seguido los estándares más importantes establecidos por los organismos internacionales encargados de establecer normativas para la interoperabilidad entre sistemas de teleformación y la construcción y reutilización de objetos de aprendizaje.

Por todo esto, considero que la aportación de este trabajo es novedosa y representa un avance en la investigación y utilización de sistemas de teleformación.

Y para que así conste firmo la presente en Alcalá de Henares a 1 de Octubre de 2009.

Atentamente,

Los directores de la Tesis Doctoral:

Dr. D. José Ramón Hilera González
Titular de Universidad.
Universidad de Alcalá.

Dr. D. Salvador Otón Tortosa
Titular de Escuela Universitaria.
Universidad de Alcalá.

Universidad de Alcalá

Dpto Ciencias de la Computación
Campus Universitario. ETS Ingeniería Informática
Crta Barcelona km 33.600
28871 Alcalá de Henares (Madrid)
Teléfonos: 91 885 66 45 Fax: 91 885 66 46

Dr. D. José Javier Martínez Herráiz, Profesor Titular de Universidad del área de Ciencia de la Computación e Inteligencia Artificial, en calidad de Director del Departamento de Ciencias de la Computación de la Universidad de Alcalá.

C E R T I F I C O

Que la tesis doctoral titulada "Propuesta de una arquitectura para la búsqueda federada y agregación semántica de objetos de aprendizaje en repositorios distribuidos" realizada por D. Antonio Ortiz Baíllo y dirigida por los doctores, D. José Ramón Hilera González y D. Salvador Otón Tortosa, reúne los requisitos para su presentación y defensa pública.

Y para que así conste, firmo la presente en Alcalá de Henares a 1 de Octubre de 2009.

Director del Departamento
Dr. D. José Javier Martínez Herráiz

AGRADECIMIENTOS

Quiero agradecer este trabajo en primer lugar a José Ramón Hilera y a Salvador Otón, mis directores de Tesis, por todo su trabajo, esfuerzo y dedicación. Gracias a vosotros he aprendido muchísimo en estos cuatro últimos años. También se lo quiero agradecer a muchos otros profesores del departamento de Ciencias de la Computación, por vuestros consejos y por insistirme siempre en que la terminara. Gracias por contar conmigo.

También se lo quiero agradecer a mis padres, porque siempre lo han dado todo por mí, por ese esfuerzo continuo, muchas veces no reconocido. Gracias de verdad por todo vuestro apoyo y por toda una vida de dedicación. A mi hermana, por ser como eres y por todos esos buenos momentos que nos haces pasar. En general a toda mi familia, porque me encanta como sois, y en particular a mis abuelos, por todo el cariño que me habéis dado desde pequeño.

Y a ti también Natalia, porque sin tu apoyo y comprensión esto hubiera sido muy difícil. Gracias por ser como eres, por hacer fácil mi vida y por contestarme siempre con una sonrisa. Me has dado todo. Te quiero.

RESUMEN

El objetivo de esta Tesis es proponer una arquitectura software para la construcción de un sistema que permita la búsqueda federada y la agregación semántica de objetos de aprendizaje de forma universal para, de esta forma, poder integrarlos en un sistema de teleformación o e-learning. Los sistemas de aprendizaje están alcanzando en la actualidad una gran proliferación, como demuestran las cifras que pueden encontrarse en los trabajos publicados sobre la materia. Sin embargo estos sistemas evolucionan constantemente a la par que los estándares a los que tratan de adaptarse.

Los sistemas de aprendizaje utilizan objetos de aprendizaje (LO: Learning Object) como base del contenido de sus cursos. Estos objetos residen en repositorios, que consisten en almacenes digitales de recursos educativos que son accesibles a través de una red de comunicaciones. El objetivo de un repositorio es facilitar la reutilización de dichos recursos educativos, facilitando el acceso a los mismos.

Para que un objeto de aprendizaje sea reutilizado debe ser desarrollado de forma que se ajuste, al menos, a algún estándar de etiquetado de metadatos asociados a los contenidos que contenga, así como que determine su construcción y empaquetado.

En el estado actual de desarrollo, ya existen sistemas que posibilitan una búsqueda más o menos eficiente de material educativo distribuido en repositorios. Estos sistemas, interoperables entre sí en la mayoría de los casos, no presentan mecanismos eficientes de búsqueda, ya que las realizan basándose en una serie de términos introducidos por los usuarios, en lugar de hacerlo en base a sus significados. Si que se ha producido un gran avance en cuanto a lo que la distribución de contenidos se refiere, pero no en cuanto al tratamiento de la información, factor muy importante en este tipo de sistemas.

En resumen, estamos ante una situación en la que las plataformas y sistemas e-learning, aunque presenten aplicaciones que las interconectan permitiendo la distribución de material docente, no cuentan con mecanismos eficientes de búsqueda y análisis de la información. Además, se pueden incorporar nuevas herramientas que faciliten aún más la reutilización del material docente que albergan estos repositorios distribuidos.

La solución que se propone es la definición de un marco funcional y arquitectónico para la adaptación de un sistema basado en SOA, e implementado en base a servicios Web (que proporciona un mecanismo flexible de integración de distintas aplicaciones y de descubrimiento de recursos en Internet), que asegure la interoperabilidad de distintos repositorios de objetos de aprendizaje y que favorezca la reutilización de los mismos. Esta arquitectura, basándose en los principales estándares y especificaciones, tendrá como característica diferenciadora que permitirá no solo la búsqueda y reutilización de material educativo aislado, sino que permitirá además la búsqueda de cursos completos de aprendizaje, obteniendo todos los módulos o partes individuales que compongan dicho curso, utilizando para ello técnicas semánticas que garanticen la efectividad de las mismas. Además, se definen una serie de modificaciones a algunas de las especificaciones que existen hoy en día, de forma que se palien algunas de las deficiencias con las que se ha encontrado el autor a la hora de su seguimiento e implementación.

En esta Tesis, en primer lugar se analiza el estado actual de los sistemas de teleformación ó e-learning, sus propuestas, sus avances y, sobre todo, sus limitaciones. Dentro de este documento se hará un especial hincapié en el estudio de los repositorios que los sustentan y los estándares que indican cómo construirlos. A partir de dichos estudios se podrá demostrar las limitaciones existentes, de forma que será factible definir nuevas propuestas encaminadas a superarlas.

Se propone una arquitectura en niveles o capas, basada en los principales estándares y recomendaciones existentes hasta la fecha, de forma que garanticen su eficiencia e interoperabilidad. Ha de satisfacer una serie de requisitos que deberán observarse como normas fundamentales a considerar en cualquier sistema que se base en dicha arquitectura. También se definen los componentes necesarios de la arquitectura para asegurar la funcionalidad requerida, así como el flujo de información y las relaciones entre ellos.

El documento finaliza con la exposición de las conclusiones y trabajos futuros relacionados con los temas tratados.

Se ha incluido un apartado con las fuentes documentales (incluidos enlaces de Internet) en las que el autor se ha inspirado, sin propósito de exhaustividad, dado que este trabajo se enmarca dentro de un contexto sometido a cambios intensos y continuos.

SUMMARY

The objective of this Thesis is a software architecture suggestion for the construction of a system which allows federated search and semantic aggregation of learning objects in an universal way so it will be possible to integrate them in learning or e-Learning system. At the present time the learning systems are reaching a great proliferation as it is demonstrated by the numbers that can be found in the published works about the subject. Nevertheless these systems develop at the same time that the standards to which they try to adapt.

The learning systems use learning objects (LO) as content base of their courses. These objects are located at repositories, which are digital stores of educative resources. These repositories implement a collection of resources (objects and/or units of learning) that is accessible through a communication network. The aim of a repository is to facilitate the reusability of those educative resources, making easier the access to them.

A learning object can be reused if it is made up in such a way that it is supported at least to some labelled standard associated to the contents that support, as well as that determines its construction and packaging.

In the current state of development, there are systems already implemented that make possible a more or less efficient search of educational content distributed in repositories. These systems, interoperable between them in most cases, do not present efficient searching mechanisms; since they realize them being based on a user introduced terms, instead of doing it on the meanings. It has produced a great advance as for what the distribution of contents refers, but not as for the data processing, very important factor in this kind of systems.

In summary, we are in the presence of a situation in which the platforms and e-learning systems, though they present applications that interconnect them allowing the distribution of educational content, they do not rely on efficient searching mechanisms and information analysis. In addition, it can be incorporated new tools that facilitate moreover the reusability of the educational content that these distributed repositories shelter.

The proposed solution is the definition of a functional and architectonic frame for the adaptation of a system based on SOA and implemented through Web services (what provides a flexible integration mechanism of different applications and resources discovery in Internet) that assures the interoperability of different learning objects repositories and which assure their reusability. This architecture, being based on the main standards and specifications, it will take as a differ characteristic that will allow not only the search and isolated educational material reusability, but it will allow in addition the search of complete courses, obtaining all the modules or individual parts that compose the above mentioned course, using for it semantic technologies that guarantee the efficiency of the same ones. In addition, a series of modifications are defined to some of the specifications that exist nowadays, so that there are relieved some of the faults which the author has found at the moment of his follow-up and implementation.

In this Thesis, in the first place it is analyzed the present state of e-learning systems (also of learning based on the Web or Internet), their proposals, their advances and mainly their restrictions. Within this document it will be made a special emphasis in the repositories study that sustain them and the standards that point out how to construct them. From these studies we will indicate the present restrictions and we will define our proposals to solve them.

It is proposed layers architecture of the system that includes a series of requirements that will have to be noticed like basic norms necessary to consider in every system based on this architecture. Also it is defined the needed components to assure the required functionality, as well as the information flow and the relations among them.

This doctoral document ends with the presentation of conclusions and future works related to the treated subjects.

It has been also included a last point with the documentary sources (including Internet links) in which we have been inspired, without any intention of thoroughly in a referential frame subdue to intense and continuous changes.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

El presente documento es la memoria de la tesis doctoral “PROPUESTA DE UNA ARQUITECTURA PARA LA BÚSQUEDA FEDERADA Y AGREGACIÓN SEMÁNTICA DE OBJETOS DE APRENDIZAJE EN REPOSITORIOS DISTRIBUIDOS”, presentada por Antonio Ortiz Baíllo dentro del programa de Doctorado “Información, Documentación y Conocimiento” del Departamento de Ciencias de la Computación de la Universidad de Alcalá.

Este primer capítulo de la memoria tiene como finalidad introducir los conceptos previos relevantes para la Propuesta y Aplicación, así como describir el propio documento para facilitar su lectura.

1.1. JUSTIFICACIÓN DE LA INVESTIGACIÓN

Actualmente vivimos en una era en la que la información y el conocimiento implican el desarrollo de una sociedad globalizada, rompiendo fronteras y límites geográficos: es lo que se suele definir como la "sociedad de la información". El propio Consejo Europeo en el año 2000 afirmaba que "la Unión Europea se enfrenta a un enorme cambio fruto de la mundialización y de los desafíos que plantea una nueva economía basada en el conocimiento" [CCE, 2000], y estableció para la Unión un objetivo estratégico esencial: "convertirse en la economía basada en el conocimiento más competitiva y dinámica del mundo".

La información es el nuevo y preciado recurso al cual hay que proporcionar la capacidad de acceso de forma universal. Esta cantidad de información que caracteriza la sociedad moderna impone más que nunca la necesidad de aprender; sin embargo, un volumen importante de información a aprender y la velocidad requerida para hacerlo pueden llegar a generar desmoralización. Como señala Pozo [2002], "nunca ha habido una época en la que hubiera tantas personas aprendiendo tantas cosas distintas a la vez, y también tantas personas dedicadas a hacer que otras personas aprendan. Estamos en la sociedad del aprendizaje".

Además, y según escribe Rosenberg [2001], "el caso es que, al mismo tiempo, muchos de los modelos de aprendizaje tradicionales se están demostrando obsoletos e inapropiados ante esta situación".

Tradicionalmente, los libros de texto han sido el mayor soporte para los contenidos educativos, el uso de libros ha definido la forma de aprendizaje en las escuelas de la sociedad moderna hasta nuestros días. Tampoco ha cambiado mucho la forma en que los profesores imparten sus clases, ya que se siguen utilizando medios muy parecidos desde hace siglos, como son la pizarra, la tiza, los libros, etc.

También la tradición cultural y el conocimiento de una sociedad residía en las bibliotecas, es decir, lugares que generalmente almacenaban documentos de distintos tipos normalmente textuales, otros gráficos e incluso audiovisuales. Cuando un profesor necesitaba material para preparar sus clases acudía a la biblioteca de su centro de enseñanza para consultarla.

Con la llegada de los ordenadores la información pasó a tener un formato digital y cuando se empezaron a construir las primeras redes de ordenadores esta información empezó a ser compartida por distintas personas que utilizaban la misma red. Con el paso del tiempo la innovación tecnológica ha conseguido que la difusión mundial del conocimiento y de la información sea algo natural. Este medio de comunicación es Internet, una inmensa cantidad de ordenadores interconectados entre sí y con una capacidad prácticamente ilimitada de almacenamiento de información, sin importar donde reside físicamente esa información.

Si el libro en soporte de papel y la imprenta supusieron un decisivo paso para la reducción de costes y acceso a la cultura de la mayor parte de la población, los nuevos instrumentos que aportan las telecomunicaciones y la informática acentúan la interactividad del texto, redescubren la creatividad de todo tipo de ilustración gráfica sin costes adicionales de reproducción, facilita el inmediato acceso en cualquier parte del mundo y hace innecesaria u opcional la reprografía. Hoy en día, los materiales multimedia y la utilización de las redes de comunicaciones introducen un nuevo elemento: la interactividad, que supone la clave del material educativo en la sociedad de la información.

Las tecnologías de la información están cambiando el acceso al conocimiento, los procesos de aprendizaje y los diferentes procedimientos establecidos dentro del campo de la educación y el adiestramiento. Tal y como afirma Martignago [1998] “la habilidad de los enseñantes, la responsabilidad creativa de los alumnos y los recursos de conocimiento distribuidos en red son las difíciles claves para la revolución didáctica”.

Para que la educación sea más eficiente debe adaptarse a los requerimientos actuales. De hecho, se pone en duda que la educación tradicional basada en la relación maestro-alumno siga siendo válida por sí sola: la sociedad, cada vez más especializada requiere trabajadores con nuevas habilidades [Nasseh, 1996].

La adopción de las nuevas tecnologías por parte de los centros de enseñanza no debe quedarse en la mera compra de ordenadores y su conexión a Internet [Levy, 1993], sino que debe ir acompañado de una estrategia pedagógica.

Resumiendo la situación explicada en los párrafos anteriores, es posible pensar en cómo se adquieren conocimientos, habilidades y destrezas y cómo se pueden generar y utilizar recursos docentes que nos mantengan actualizados en esta nueva sociedad de la información y el conocimiento. Para enfrentarnos a esta situación existen actualmente una serie de tecnologías utilizadas en el ámbito del e-learning las cuales se pueden resumir en:

Los **objetos o unidades de aprendizaje** son la pieza clave en la construcción del material docente de forma que los contenidos educativos se fragmentan en unidades modulares independientes que pueden ser reutilizados en distintos entornos y en diferentes aplicaciones. Por tanto, la labor de un creador de contenidos didácticos será elaborar correctamente "unidades de aprendizaje", esto es, "unidades mínimas en las que se puede organizar el material de formación para facilitar la gestión del conocimiento: creación, indexación, almacenamiento, distribución, uso, reutilización, evaluación y mejora de la formación" [Moreno y Bailly-Baillière, 2002].

Por lo tanto, el empleo de objetos de aprendizaje permite reutilizar los contenidos creados de una determinada experiencia educativa en contextos de aprendizaje diferentes. El proceso de construcción y distribución a los usuarios del material docente implica por tanto la creación, descubrimiento y agregación de unidades de aprendizaje simples en recursos educativos más complejos.

Para una completa reutilización de los objetos de aprendizaje es necesario ceñirse a los **estándares de e-learning**, en los que se han visto involucrados numerosas instituciones como IMS (*Instructional Management Systems Global Learning Consortium*), ADL (*Advanced Distributed Learning*) o IEEE (*Institute of Electrical and Electronics Engineers*). En estos estándares se hace especial hincapié en la necesidad de acompañar a los contenidos de los objetos de aprendizaje con un registro de metadatos que informe sobre el contenido del objeto y de su uso más apropiado [Sosteric y Hesemeier, 2002] y de su adecuación a unos determinados objetivos de aprendizaje.

Los llamados **Sistemas de Gestión del Aprendizaje** (*Learning Management System* – LMS) constituyen una categoría de software que automatiza la administración de acciones de formación: gestión de usuarios, gestión y control de cursos, gestión de los servicios de comunicación, etc. Estos sistemas gestionan los contenidos almacenados

generalmente en repositorios. Aunque suelen ser meras adaptaciones de contenidos tradicionales, cumplen una labor muy productiva y beneficiosa en múltiples ámbitos; si bien presentan algunas carencias (poca formalización y estructuración del conocimiento, poca adaptabilidad a los estándares, contenidos demasiado expositivos, etc.) que los configuran como herramientas con limitaciones en su interacción con los alumnos.

Un **repositorio** o almacén digital de recursos educativos es una colección de recursos (objetos o unidades de aprendizaje) que son accesibles a través de una red de comunicaciones. No se necesita un conocimiento previo de la estructura de la colección, la cual puede contener los propios recursos o únicamente los metadatos que los describen, junto con una referencia para su localización [IMS, 2003a]. Por lo tanto, el objetivo de un repositorio es facilitar la reutilización de recursos educativos, facilitando el acceso a los recursos almacenados en el mismo. Los servicios que un repositorio de este tipo debe ofrecer al exterior, están relacionados con la búsqueda de objetos de aprendizaje a partir de metadatos, con el acceso a los objetos de aprendizaje localizados, o con el almacenamiento de objetos de aprendizaje en el repositorio.

Estas tecnologías son las que utilizan los sistemas de teleformación para la adquisición del conocimiento por parte del alumno; sin embargo, falta una pieza clave para que un sistema de este tipo sea o no útil: que haga un uso adecuado de las posibilidades de la Red.

Generalmente la gran mayoría de los sistemas actuales utilizan la red como un mero medio de comunicación. Sin embargo, si el sistema hace un uso avanzado de la red será posible utilizar sus posibilidades para desarrollar al máximo la interoperabilidad entre sistemas de aprendizaje, la flexibilidad, la disponibilidad y la ubicuidad de los mismos; de forma que los objetos de aprendizaje que gestionan sean reutilizables, accesibles y adaptables. Este enfoque no implica necesariamente mayor complejidad en el software a desarrollar, al contrario, un sistema basado en esta filosofía debe ser sencillo para ser universal.

Parece, por tanto, oportuno intentar una síntesis entre los sistemas de teleformación y las posibilidades de la red, encaminada a una solución convergente para la construcción de una nueva generación de sistemas de teleformación distribuidos y en red, donde la

interoperabilidad entre sistemas y la reutilización de objetos de aprendizaje sean las piezas clave de su construcción.

1.2. OBJETIVOS Y MÉTODO DE TRABAJO

Así pues, es posible determinar como punto de partida que justifique el desarrollo de este trabajo la siguiente proposición:

La investigación objeto de esta Tesis se justifica en la adaptación y actualización de una arquitectura software capaz de cubrir la necesidad de reutilizar objetos de aprendizaje almacenados en diferentes repositorios, que permitirá localizarlos independientemente de su ubicación física y de su tecnología de almacenamiento a través de Internet de manera que haga que los sistemas de teleformación sean interoperables, aportando además sistemas que optimicen la búsqueda de estos contenidos mediantes técnicas semánticas, así como que permita la creación de cursos completos como resultado de la agregación de diversos objetos de aprendizaje provenientes de diferentes repositorios distribuidos.

Un sistema construido en base a la arquitectura que se propone, y que se podría denominar **LORS-SC (*Learning Object Reusability System – Semantic & Composite*)**, en Español: Sistema de Reutilización de Objetos de Aprendizaje – Semántico y de Composición, dotaría a los sistemas tradicionales de docencia y aprendizaje a través de Internet de la flexibilidad y capacidad de poder publicar los objetos de aprendizaje de sus repositorios, y obtener objetos de otros repositorios externos de forma que la reutilización de estos objetos sea máxima y transparente para el usuario.

Gracias a este desarrollo, es posible esperar los siguientes beneficios:

- Se propone una arquitectura abierta y fácil de adaptar a los sistemas actuales de teleformación ya que utiliza protocolos y formatos estándar para permitir la interoperabilidad e integración de dichos sistemas.
- Se ofrece la posibilidad de publicar y descubrir cualquier objeto de aprendizaje independientemente de su localización y los metadatos utilizados para su descripción didáctica.
- Se presenta una forma uniforme y bien definida de acceso a los objetos de aprendizaje.
- Se fomenta que los objetos de aprendizaje sean reutilizables.
- Se fomenta el uso de técnicas semánticas como garantía de unas búsquedas eficientes y más próximas al lenguaje humano.

Conviene, sin embargo, precisar una serie de principios que se creen fundamentales a la hora de desarrollar y adaptar una arquitectura de estas características:

- La tecnología que proporciona Internet es la llave de una profunda revolución para la enseñanza. Pero la tecnología, cualquier tecnología, es una herramienta, no una estrategia.
- "La mejor tecnología no es aquella que reemplaza la realidad o la inteligencia con formas artificiales, sino aquella que hace aumentar las propias de cada uno" [Hodgins, 2000].
- Las clases presenciales continuarán teniendo un papel fundamental en la educación. "Quienes piensen que la tecnología reemplazará totalmente a los profesores al frente de una clase están tan equivocados como los que creen que Internet es una fiebre pasajera" [Rosenberg, 2001].

Llegado este punto es posible avanzar conforme a los siguientes objetivos concretos:

- Definir una arquitectura orientada a servicios para la localización universal de objetos de aprendizaje.
- Describir cada uno de los servicios presentes en la arquitectura y su organización en la misma.
- Estudiar las características de los repositorios existentes en la actualidad y su adaptabilidad a las especificaciones y estándares.
- Facilitar la interoperabilidad de sistemas de teleformación aunque utilicen distintos estándares de metadatos de los objetos docentes que intercambian.
- Ofrecer una capa de abstracción superior con servicios comunes disponibles para todos los sistemas que cubran sus necesidades de comunicación.
- Basar la propuesta en estándares reconocidos y sólidos.
- Construir un prototipo basado en la arquitectura propuesta.

Estos objetivos que se acaban de proponer están en sintonía con las actuales investigaciones y trabajos sobre la utilización de los ordenadores y los sistemas de teleformación como ayuda a la docencia y el aprendizaje. Esta propuesta contribuye a la definición de un marco avanzado de dicha utilización. Así mismo, se incluye, y por supuesto se sigue, las recomendaciones de los principales grupos y comités tanto de estandarización como de implantación y uso de las nuevas tecnologías en el ámbito del aprendizaje.

Las actividades y el método de trabajo seguidos para conseguir los anteriores objetivos han sido las siguientes:

- Examen y análisis de arquitecturas existentes en el ámbito de los Sistemas de Gestión del Aprendizaje. De forma que la arquitectura propuesta se integre de la mejor forma, desde el punto de vista funcional y físico, con los sistemas existentes.
- Búsqueda y estudio de los diversos estándares existentes relacionados con la teleformación. Se estudiarán las diferentes propuestas y relaciones de los estándares más importantes. Fundamental en el desarrollo futuro de cualquier sistema de teleformación a través de Internet.
- Examen y análisis de los repositorios utilizados por los Sistemas de Gestión del Aprendizaje. Aquí se estudiarán las principales características de los repositorios desde su construcción hasta su utilización, y se analizarán los problemas que tienen los repositorios más utilizados en Internet.
- Definición de la arquitectura propuesta. Donde se realizará una descripción detallada de cada una de las capas que la integran y de cada uno de los servicios utilizados en cada capa.
- Estudio de viabilidad de la propuesta mediante la implementación y prueba de un prototipo de la arquitectura que llevará a la conclusión de la factibilidad o no de su implantación.
- Por último se plantearán una serie de conclusiones a partir de la evaluación de la arquitectura así como una propuesta de futuras líneas de investigación.

1.3. ESTRUCTURA DEL DOCUMENTO

El documento en el que se recoge el resultado de la investigación llevada a cabo consta de cinco capítulos y un anexo, estructurándose tal y como se describe a continuación.

En el **Capítulo 1**, que se concluye en este apartado, se incluye una introducción con la motivación y justificación de la investigación que se ha realizado, así como los objetivos y las actividades realizadas durante el desarrollo del trabajo para alcanzar los fines enunciados.

En el **Capítulo 2** se desarrolla un estudio sobre "el estado del arte" en el ámbito de la teleformación y las arquitecturas orientadas a servicios. Se trata del capítulo más denso y que se podría dividir en siete sub-apartados:

- Sistemas de aprendizaje online: en este apartado se hace una revisión histórica de la evolución que han sufrido los sistemas de aprendizaje a distancia desde su origen hasta nuestros días, para continuar con una introducción a los sistemas de gestión del aprendizaje, en el que se detallarán las principales ventajas e inconvenientes que aportan a la sociedad actual, así como las características más novedosas que han incluido recientemente.
- Estándares: en este apartado se realiza una revisión bibliográfica para conocer realmente el significado del término "estándar" y cómo sería el proceso de toda especificación hasta convertirse en estándar. Posteriormente se realiza una enumeración de las principales organizaciones de estandarización relacionadas con entornos e-learning, indicando para cada una de ellas, cuáles son sus principales trabajos e iniciativas.
- Principales estándares y frameworks para entornos e-learning: como conclusión al anterior apartado, se profundiza en los principales estándares, recomendaciones o frameworks con los que se cuenta actualmente, para que sirva como base para el posterior desarrollo del cometido de esta tesis.
- Arquitecturas: ya que la misión principal de esta tesis es la de investigar en arquitecturas que puedan aportar novedades a los entornos e-learning, es fundamental hacer un estudio del significado del término arquitectura, así como indicar las principales arquitecturas que existen a día de hoy en el mercado, verificando sus principales características y posibles problemáticas.
- Arquitecturas orientadas a servicios: como conclusión al apartado anterior, se comentan en mayor profundidad un tipo de arquitecturas que actualmente

están muy demandadas por la sencillez, independencia, flexibilidad y beneficios que aportan a sus futuros desarrollos.

- La semántica y su aplicabilidad en entornos e-learning: en este apartado se realiza una introducción a las técnicas semánticas que existen en la actualidad y cómo sería su aplicabilidad a los sistemas e-learning, de forma que se detallan las principales ventajas que se obtendrían en función de las partes en las que se añadieran.
- Conclusiones: el apartado final de este capítulo no podía ser otro que aquel en el que se expresen las principales conclusiones que se han obtenido de este estudio inicial, de forma que sirva de base para comenzar un proceso de investigación que se irá desgranando en el resto de capítulos que conforman esta tesis.

El **Capítulo 3** plantea la problemática a resolver, partiendo de un estudio de la evolución de los sistemas de aprendizaje para determinar dónde encaja la arquitectura propuesta. A continuación se hace un estudio de las deficiencias encontradas hoy en día, tanto en los Sistemas de Gestión del Conocimiento como en los Sistemas de Gestión de Contenidos para llevar a cabo la reutilización de los objetos de aprendizaje que gestionan. Para ello se examinarán los principales repositorios que se utilizan en Internet para publicar sus objetos de aprendizaje de forma que sea posible contrastar sus limitaciones. Posteriormente se realizará una reflexión sobre el significado del término arquitectura (ya que al fin y al cabo, este trabajo platea una), así como las condiciones que van a marcar su desarrollo, para posteriormente describir las principales características con las que cuenta la arquitectura y el sistema precedente al estudio de esta Tesis en el que participó el autor de esta Tesis.

El **Capítulo 4** estaría a su vez dividido en dos sub-apartados. Se trata de uno de los dos capítulos principales de este trabajo, ya que es en él donde se detalla la nueva arquitectura propuesta:

- Propuesta de la arquitectura inicial - LORA-SQI: Se trata de la definición de una arquitectura, modificación de la descrita en el capítulo 3, y en la que principalmente se adapta a las nuevas especificaciones existentes para garantizar su completa interoperabilidad con el resto de sistemas.
- Propuesta de la arquitectura ampliada - LORA-SC: Se trata de una modificación de la arquitectura anterior en la que se añaden técnicas semánticas para posibilitar tanto una mayor eficiencia en las búsquedas

como nuevas funcionalidades derivadas de su uso, entre las que se destaca la de permitir la búsqueda de cursos completos de aprendizaje.

El **Capítulo 5** contiene la definición de los sistemas construidos a partir de las arquitecturas descritas en el anterior capítulo, de forma que se valida así la propuesta de Tesis. Para ambos sistemas, se muestran sus diagramas de casos de uso, clases, interfaces, etc.

El **Capítulo 6** contiene las conclusiones de la investigación llevada a cabo junto con algunas líneas de investigación que se podrían abrir a su finalización.

El **Capítulo 7** contiene las referencias bibliográficas citadas a lo largo de la Tesis.

2. ESTADO DEL ARTE

En este capítulo se efectúa un estudio sobre "el estado del arte" de los sistemas de enseñanza y aprendizaje en relación con las nuevas tecnologías de información y comunicaciones, así como todo el conjunto de estándares, especificaciones y recomendaciones que se han creado en los últimos años para garantizar que su funcionalidad sea completa.

Dentro de este capítulo, se encuentran los siguientes apartados que a continuación se resumen:

- Sistemas de aprendizaje online: en este apartado se hace una revisión histórica de la evolución que han sufrido los sistemas de aprendizaje a distancia desde su origen hasta nuestros días, para continuar con una introducción a los sistemas de gestión del aprendizaje, en el que se detallarán las principales ventajas e inconvenientes que aportan a la sociedad actual, así como las características más novedosas que han incluido recientemente.
- Estándares: en este apartado se realiza una revisión bibliográfica para conocer realmente el significado del término "estándar" y cómo sería el proceso de toda especificación hasta convertirse en estándar. Posteriormente se realiza una enumeración de las principales organizaciones de estandarización relacionadas con entornos e-learning, indicando para cada una de ellas, cuáles son sus principales trabajos e iniciativas.
- Principales estándares y frameworks para entornos e-learning: como conclusión al anterior apartado, se profundiza en los principales estándares, recomendaciones o frameworks con los que se cuenta actualmente, para que sirva como base para el posterior desarrollo del cometido de esta tesis.
- Arquitecturas: ya que la misión principal de esta tesis es la de investigar en arquitecturas que puedan aportar novedades a los entornos e-learning, es fundamental hacer un estudio del significado del término arquitectura, así

como indicar las principales arquitecturas que existen a día de hoy en el mercado, verificando sus principales características y posibles problemáticas.

- Arquitecturas orientadas a servicios: como conclusión al apartado anterior, se comentan en mayor profundidad un tipo de arquitecturas que actualmente están muy demandadas por la sencillez, independencia, flexibilidad y beneficios que aportan a sus futuros desarrollos.
- La semántica y su aplicabilidad en entornos e-learning: en este apartado se realiza una introducción a las técnicas semánticas que existen en la actualidad y cómo sería su aplicabilidad a los sistemas e-learning, de forma que se detallan las principales ventajas que se obtendrían en función de las partes en las que se añadieran.
- Conclusiones: el apartado final de este capítulo no podía ser otro que aquel en el que se expresen las principales conclusiones que se han obtenido de este estudio inicial, de forma que sirva de base para comenzar un proceso de investigación que se irá desgranando en el resto de capítulos que conforman esta tesis.

2.1. LOS SISTEMAS DE APRENDIZAJE ONLINE

En este apartado se van a describir todos los conceptos necesarios y que servirán de base para el posterior desarrollo de la arquitectura del sistema, eslabón central del estudio de esta tesis.

Inicialmente se hace una revisión histórica de cómo han evolucionado los sistemas de formación hasta llegar a convertirse en lo que hoy conocemos como sistemas e-learning. Posteriormente se introducen una serie conceptos fundamentales para la comprensión del resto de apartados; conceptos como “estándar”, “proceso de estandarización”, “objeto de aprendizaje”, “metainformación”, “repositorio”, etc.

2.1.1. Evolución histórica

La Educación es un dominio en el que históricamente se han intentado aprovechar las capacidades introducidas por el desarrollo de la tecnología; se podría decir que los cursos por correspondencia postal iniciados por Sir Isaac Pitman en 1840 fueron los primeros [Horton, 2001], ya que constituyeron el primer intento de utilizar las comunicaciones para extender la formación más allá del sonido de la voz humana.

Muchos de los conceptos que forman la base de la teleformación estaban ya presentes entonces: el alumno podía marcar su propio ritmo de aprendizaje, éste podía llevarse a cabo sin contacto directo (cara a cara), y un elevado número de estudiantes podían seguir el curso con una programación diferente.

En la década de los años 60 del pasado siglo, el uso de los ordenadores como soporte al proceso educativo y/o pedagógico propicia el nacimiento de las primeras (y primitivas) aplicaciones de educación asistida por computador, CAI (*Computer Assisted Instruction)*, en la universidad de Stanford. Eran aplicaciones para el aprendizaje de la lectura y las matemáticas, que permitían individualizar dicho aprendizaje liberando al alumno de la asistencia a clase en grupos tradicionales. Estas aplicaciones poseían mecanismos de retro-alimentación a través de preguntas y respuestas de los propios alumnos, que permitían al mismo participar activamente en su propia formación [Molnar, 1990]. Los costes, tanto económicos como de tiempo de desarrollo, eran el mayor obstáculo para la expansión de estos métodos. Inicialmente se trataba de ordenadores *mainframe* programados en ensamblador o incluso lenguaje máquina, lo cual hacia larga y tediosa su programación, con lo que el uso de estos medios fue reducido y más bien experimental.

En los años 70 empieza a aparecer un conjunto de aplicaciones que hacen uso de técnicas avanzadas, como la Inteligencia Artificial o la Ciencia Cognitiva, y que suponen un gran adelanto en la forma en que se realizaba el aprendizaje asistido por computador [Brown et al., 1989]. El término más ampliamente aceptado para designar a estas aplicaciones es el de “sistemas de tutorización inteligentes” o “aplicaciones inteligentes de enseñanza asistida por computador” (ICAI - *Intelligent Computer Assisted Instruction*). El objetivo era conseguir sistemas que literalmente “entendiesen” un dominio de conocimiento y fuesen capaces de orientar al alumno a la hora de articular sus propias ideas y estrategias de funcionamiento.

En la década de los 80, cuando los ordenadores comenzaron a ser habituales en los hogares y en las oficinas, se empezaron a implantar los primeros métodos de aprendizaje por medio de éstos de una forma amplia. Empresas como IBM o Digital comenzaron a crear los primeros cursos para ser usados por medio de un ordenador. Estos cursos eran bastante sencillos, constaban fundamentalmente de texto y estaban distribuidos en forma de capítulos para ser seguidos de forma secuencial. Los cursos instruían sobre el manejo de aplicaciones de interés general, tales como procesadores de texto y hojas de cálculo; solían distribuirse junto con la aplicación relacionada y generalmente usando un disco magnético como soporte; con esto se conseguía que el futuro usuario de la aplicación pudiera aprender por sí mismo. Se trataba de cursos de autoestudio y prácticamente lo único que cambiaba con respecto a un libro era el soporte (de papel a soporte magnético). Esto fue el origen de lo que hoy se conoce como aprendizaje asistido por computador, *Computer Based Training* - CBT[1].

Seguidamente surgieron empresas particulares que se dedicaron de forma específica a crear materiales didácticos para ser utilizados por medio del computador. Generalmente los temas tratados estaban relacionados con la propia Informática, como son los sistemas operativos, lenguajes de programación o uso de aplicaciones. Los soportes utilizados para la distribución fueron discos magnéticos y posteriormente discos ópticos como CD-ROMS, incluso llegando a hacer uso de redes de comunicación tanto locales como globales.

A lo largo de los años 90 aparecen y se extienden dos grupos de tecnologías que impactan profundamente en el mundo de la Informática y en los métodos de adiestramiento y enseñanza que usan el computador como punto central del proceso de aprendizaje:

- Las Tecnologías Multimedia.
- Las Redes de Comunicación.

En lo que se refiere a la actualidad, Internet se ha presentado como el gran paradigma de los nuevos tiempos, su abrumadora y acelerada penetración en nuestras vidas ha decantado en lo que se conoce como la Sociedad de la Información y del

[1] Hay términos alternativos a éste que vienen a significar lo mismo, como son, el ya citado CAI (*Computer Assisted Instruction*), CBI (*Computer Based Instruction*) o CAL (*Computer Assisted Learning*).

Conocimiento. Los espacios educativos no han escapado a ello y han venido aplicando esta nueva tecnología de manera poco planificada y de una forma tal que muchos la ven como poco efectiva y deficiente.

Como nueva tecnología, trata de ganar terreno al impulsar prácticas y procesos que le ayuden a superar un inicio poco claro. La combinación de la telecomunicación y la enseñanza ha permitido acuñar el término "*e-learning*" [Anido et al., 2002]. El e-learning, concepto adoptado para denominar el proceso enseñanza–aprendizaje que generalmente usa Internet en la educación, trata de formalizarse a través de métodos y herramientas de calidad. Los estándares e-learning están llamados a ser uno de los pilares fundamentales que ayudarán a gestionar con eficiencia uno de los activos más preciados de la denominada nueva economía: el conocimiento.

Este tipo de formación es realmente parte de lo que tradicionalmente se ha conocido como "Enseñanza Asistida por Ordenador" (EAO), si bien haciendo uso exhaustivo de las tecnologías de telecomunicación; por lo que el concepto de EAO está inevitablemente unido a la teleformación, como así lo refleja la siguiente definición de Lanfranco [1993]: *"la enseñanza asistida por computadora es la organización y combinación de los recursos educativos y tecnológicos para permitir el tele-aprendizaje por parte de los alumnos"*.

El e-learning aporta a la educación la desaparición real de los problemas de espacio y de horarios. Los alumnos pueden realizar su aprendizaje desde cualquier sitio y a cualquier hora. Los sistemas de enseñanza asistida por computador están disponibles 24 horas al día. Proporcionan un canal de comunicación entre los propios alumnos, y entre éstos y los profesores. Hiltz y Norwood [1994] concluyen que la participación de los estudiantes puede llegar a ser superior en un entorno como éste que en un aula convencional. El canal de comunicación que se establece puede utilizarse con finalidades de seguimiento y tutorización de los alumnos por parte de los profesores. La información extraída de este seguimiento puede ser empleada posteriormente para labores de evaluación.

Dado que, habitualmente, el ámbito de actuación de estos sistemas es universal, los alumnos pueden elegir entre una gran diversidad de materias, cursos y especialidades. Éstos pueden ser preparados por los mejores especialistas en cada materia, para ser

distribuidos a un conjunto amplio de estudiantes dispersos geográficamente. No sólo aquellos alumnos cercanos físicamente a ellos se benefician de éstos conocimientos.

El profundo cambio que ha supuesto Internet en el campo de las aplicaciones educativas no sólo afecta a aspectos puramente tecnológicos: también los paradigmas educativos se están alterando como consecuencia de las nuevas demandas sociales, en las que las redes globales de comunicación juegan un papel esencial. Actualmente se imponen los modelos educativos centrados en el alumno: ha crecido la demanda hacia una formación continua y "de por vida" en contraposición a los métodos educativos tradicionales en los que la formación se recibe en periodos determinados y centros específicos. Se trata de formar a alumnos con poco tiempo disponible y que necesitan obtener un elevado rendimiento.

Por otro lado, en éste análisis no se puede dejar de lado el componente principal de cualquier aplicación educativa: los contenidos educativos. Parece evidente que en última instancia, el éxito de una aplicación educativa radica en la calidad de sus contenidos. Por lo general, en la mayoría de las aplicaciones disponibles, los cursos se crean con el fin de cubrir una necesidad de aprendizaje concreta. Sin embargo, producir desde cero material educativo de alta calidad es una labor ardua que lleva mucho tiempo y que requiere del conocimiento de diversos expertos en distintos campos.

La gestión del conocimiento se ha convertido en un tema recurrente en Informática. En este contexto, el conocimiento no sólo se identifica como un nuevo factor de producción de las sociedades industrializadas, sino como un producto en sí mismo. Para poder determinar su valor de cambio, el conocimiento debe ser creado, almacenado y gestionado; razón por la cual emergen las tecnologías de la información y la comunicación como soporte a la gestión del conocimiento [Dodero, 2002].

En este sentido, como forma de compartir el conocimiento, una de las pretensiones en el ámbito del e-learning es la aplicación de **procedimientos que permitan la reutilización efectiva de material docente ya desarrollado**, e idealmente, que faciliten dicha reutilización no sólo dentro del mismo entrono del e-learning para distintos cursos, sino entre aplicaciones diferentes con herramientas de creación de contenidos y plataformas distintas.

La Tabla 2.1 (adaptada de Sigh [2001]) muestra una comparación entre los requisitos de los sistemas educativos tradicionales asistidos por computador (*Computer Based Training*) y los de las nuevas aplicaciones e-learning derivados de la utilización masiva de Internet como medio de distribución de información y comunicación; así como del cambio de mentalidad asociado a la adquisición del conocimiento y la formación en la sociedad actual.

ACTIVIDAD	ANTIGUOS SISTEMAS CBT	NUEVOS SISTEMAS E-LEARNING
Creación de contenidos	Realizado por el instructor. Los cursos se crean desde cero de principio a fin	Realizado por el instructor/diseñador del curso. Necesita conocimientos sobre la herramienta. Los cursos se crean recombinando material existente con nuevo material: incrementando el valor del nuevo curso
Distribución	Cara y complicada	Internet. Barato
Modelo educativo	Centrado en el instructor	Centrado en el alumno
Objetivo	Distribuir conocimiento	Distribuir y capturar conocimiento
Elemento o pieza de distribución / creación	Cursos completos	Módulos u objetos educativos (Learning Objects)
Actualizaciones	Reconstruir el curso y reenviarlo	Actualizar módulo
Velocidad	Depende del tamaño de la audiencia y de la extensión del contenido	Dependiente de la extensión del contenido y de la velocidad de la red de acceso
Tiempo típico	4 – 6 meses	4 – 6 semanas
Medidas de la efectividad	Observaciones del instructor	Sistemas de seguimiento y evaluación interactivos
Fuentes de conocimiento	Se crean todas	Buscar si existe material reutilizable y ensamblarlo (idealmente sin necesidad de adaptar cambios)

Tabla 2.1 Comparativa entre las aplicaciones tradicionales CBT y los sistemas e-learning

Teniendo en cuenta todo lo anterior, a partir de este punto del documento de tesis, se considera lo siguiente:

“Teleformación o e-learning es la aplicación de Tecnologías de la Información y Comunicación para desarrollar, gestionar y distribuir cursos de formación.”

Cuando se habla de *“aplicación de Tecnologías de la Información y Comunicación”*, se refiere en particular a las que se encuentran relacionadas con Internet. El término e-learning, no sólo cubre la distribución del curso, sino que también cubre el seguimiento, la programación, la gestión y otros aspectos del proceso de la enseñanza. Es decir, los sistemas de e-learning no sólo comprenden el contenido del curso, sino la plataforma tecnológica que lo distribuye y lo gestiona, y los servicios que soportan el mantenimiento. De hecho, las mayores compañías de e-learning no desarrollan el contenido, sino que se centran en las tecnologías (plataformas, gestión de desarrollo de contenidos, etc.) y servicios que permiten que el contenido sea eficazmente diseñado, distribuido y gestionado.

Se hacen necesarias, por tanto, nuevas tecnologías que faciliten y permitan llevar a cabo todos esos aspectos a través de Internet de una manera rápida y eficaz.

2.1.2. Sistemas de Gestión del Aprendizaje

Dentro del mundo del e-learning existe una inmensa cantidad de siglas, herramientas, arquitecturas, sistemas y tipos diferentes de aplicaciones que han aparecido en los últimos años; además, las diferencias y límites entre unas denominaciones y otras no están siempre claros. Antes de centrar el desarrollo del apartado, se va a presentar una clasificación de este tipo de herramientas en función del cometido de las mismas [Gram et al., 1998]:

- **Herramientas de creación de contenidos**: Para la creación y edición de documentos HTML, PDF y otros formatos de texto, así como la creación de gráficos, animaciones, audio, video, etc.
- **Herramientas de creación de contenidos Web**: Aquellas ideadas para la edición basada en los formatos que se usan para ser publicados en Internet, como HTML. Es un subgrupo de la categoría anterior.
- **Herramientas de comunicación**: Aplicaciones de *chats* de texto, videoconferencias y otros medios de comunicación, tanto síncrona como asíncrona.
- **Herramientas de autor para Internet**: Permiten la creación de contenidos y su publicación en el Web, incluso de forma simultánea.
- **Entornos educativos integrados distribuidos**: Enfocados a la distribución y gestión de materiales, a veces facilitan la administración de participantes, gestión de pruebas, incluyendo muchas veces partes o herramientas completas como las clasificadas anteriormente.

Como ya se ha indicado anteriormente, este estudio fue realizado en el año 1998 y muchas de las herramientas consideradas han desaparecido o, si aún existen, han ampliado sus funcionalidades y características; además, han aparecido muchas nuevas. Sin embargo, esta aproximación es considerada todavía como válida a la hora de clasificar las herramientas existentes.

R.H. Jackson, de la Universidad de Tennessee, insiste en la necesidad de distinguir los entornos según sus funcionalidades, pero además incluye una nueva opción, clasificar dichos entornos según la forma en la que se realiza el aprendizaje. De esta manera Jackson ofrece dos clasificaciones de entornos de aprendizaje, según la forma y según la funcionalidad [Jackson, 2001]. Esta clasificación sólo tiene en cuenta entornos de aprendizaje diseñados para dicho fin, sin incluir herramientas de creación de contenidos ni de comunicación.

Según la forma, Jackson clasifica los entornos en función del formato en que se realiza la enseñanza. Usando esta clasificación podemos determinar primero el formato que mejor se adecua a las necesidades del proceso y, después, seleccionar el entorno que más se ajuste a ellas.

Los entornos se clasifican en tres tipos, según el formato:

- Estudio dirigido.
- Eventos dirigidos por instructor.
- Entornos colaborativos.

Un entorno de e-learning puede proveer más de uno de estos formatos y por tanto pertenecer a más de un grupo.

Estudio dirigido se refiere a aquel donde el alumno realiza su propio aprendizaje haciendo uso de tutoriales en CD-ROM, entorno *on-line* de aprendizaje, etc. Este estudio es complementado con interacción asíncrona con un instructor y con el resto de alumnos. El instructor guía o dirige al alumno en su aprendizaje y le resuelve dudas, aunque es el propio alumno quien se responsabiliza de aprender.

Eventos dirigidos por un instructor son aquellas actividades formativas síncronas donde existe una figura de profesor que imparte los conocimientos y dirige el aprendizaje. Permite un mejor ajuste del flujo de la enseñanza a las necesidades de los alumnos. Este método puede ser útil para complementar el estudio dirigido.

Finalmente, los entornos colaborativos son comunidades de aprendizaje. Proporcionan herramientas de comunicación síncronas y asíncronas para que pequeños grupos de alumnos colaboren en sus estudios comunes, resolución de problemas, proyectos de investigación, etc. Puede existir un instructor o tutor que actué como soporte o guía.

Mientras que esta clasificación según la forma se centra en el aprendizaje, no tiene en cuenta las características técnicas; por eso Jackson propone una segunda clasificación basada en la funcionalidad de los entornos; bajo esta categoría se incluyen tres tipos:

- ***Educational Delivery Systems*** (Sistemas de Distribución): Estos productos facilitan la publicación y distribución de contenidos *on-line*, no se centran en la creación de los mismos y no contienen mecanismos para medir el rendimiento o administrar recursos.
- ***Learning Content Management Systems*** (LCMS) (Sistemas de Gestión de Contenidos): Estos entornos combinan la creación y distribución de materiales educacionales con mecanismos para medir el resultado y monitorizar el progreso de los estudiantes.
- ***Learning Management Systems*** (LMS) (Sistemas de Gestión del Aprendizaje): Son similares a los LCMS pero dan a los estudiantes y a las organizaciones una visión integrada de todos los trabajos activos en múltiples cursos. Se centran más en la distribución de materiales y en el seguimiento y control de todos los elementos involucrados en el proceso educativo.

La diferencia entre los LCMS y los LMS es sutil. Los primeros permiten tanto la distribución como la creación de materiales, y pretenden que los educadores creen sus propios materiales *on-line*, mientras que los LMS se centran en la distribución y control de los materiales y no suelen incluir herramientas de creación de contenidos, o estas suelen ser muy simples; por tanto, los materiales deben ser creados externamente. Los LMS permiten siempre la creación de materiales o cursos completos exteriormente y después su integración al sistema. Con la implantación de los estándares, será incluso posible migrar los cursos de una plataforma a otra con todos sus contenidos.

Learning Management Systems

Según lo definido anteriormente, un Sistema de Gestión del Aprendizaje (*Learning Management System*) es aquel sistema software que permite la distribución y gestión de conocimientos *on-line*. Un LMS permite realizar el aprendizaje habilitando métodos para ello, permitiendo la distribución de materiales docentes y contenidos. Además ofrece funcionalidades para gestionar y administrar todos los componentes que forman parte del proceso educativo.

La Web EduTools [EduTools, 2009], supervisada por WCET (*Western Cooperative for Educational Telecommunications*), se encarga de realizar una evaluación independiente de los distintos proveedores de herramientas de e-learning del mercado fijándose en sus principales funcionalidades, siendo algunas de ellas las siguientes:

- **Gestión e informe de los alumnos:** Permite a los administradores organizar a los participantes (otros administradores, educadores y estudiantes) en grupos lógicos, y monitorizar e informar sobre su progreso y actividades. Algunas utilidades de este tipo son:
  - Creación de registros de los alumnos y asignación de derechos de acceso.
  - Reparto de responsabilidades y funcionalidades entre varios administradores.
  - Creación de informes.
  - Mensajería.
- **Gestión de recursos y eventos de aprendizaje**: Permite a los administradores organizar los cursos y eventos en catálogos, al igual que gestionar todos los recursos que componen cada curso, incluyendo contenidos, instructores, "aulas virtuales", etc. También permite comunicarse a estudiantes y administradores e informar sobre las actividades de los alumnos. Las principales funciones son:
  - Creación y gestión de recursos.
  - Distribución de derechos de acceso y del resto de permisos de los cursos.
  - Gestión de eventos, exámenes, foros de discusión, etc.
- **Distribución de cursos on-line**: Ofrece una infraestructura a los administradores para habilitar métodos de gestión del aprendizaje:
  - Medios síncronos, todos los participantes se reúnen a la misma hora y los eventos son dirigidos por un instructor.
  - Medios asíncronos, los materiales se encuentran a disposición de los alumnos y estos se responsabilizan de su propio aprendizaje.

  Se incluye también la creación del plan de estudios del curso y el calendario del mismo.
- **Creación y publicación de contenidos**: Los cursos pueden ser creados dentro del propio entorno o exteriormente al mismo:
  - Para cursos creados exteriormente, en un LMS:
    - Se permite la publicación de contenidos creados en formatos Web estándar.
    - Se exige ajustarse en lo posible a los estándares existentes.
  - En las herramientas de creación propias:

- Se permite la integración de los sistemas de creación y publicación.
- Se ofrece flexibilidad en las plantillas de creación de contenidos.

Por otra parte se debe permitir importar y exportar cursos en un único paso.

- **Evaluación de conocimientos y habilidades:** Permite valorar las habilidades de los estudiantes, bien para determinar el curso o el plan de estudios que más se ajusta a sus necesidades o bien para saber el nivel de conocimientos adquirido en el aprendizaje. Todo ello mediante:
  - La creación de tests y pruebas y su distribución y asignación temporal.
  - La generación de informes sobre la realización y resultados de las pruebas.
- **Gestión de bases de conocimiento:** En un LMS, debe integrarse un sistema específico de gestión de recursos accesible desde todos los cursos, que incluya contenidos de cualquier tipo en diversos formatos. También permite acceder a recursos externos como complemento a los cursos *on-line*.

  En todo caso, aunque muy publicitada últimamente, esta funcionalidad está escasamente desarrollada y se detectan carencias de formalización, estructuración y conexión entre sus elementos.
- **Personalización del entorno centrada en el alumno:** Hace que el entorno reconozca al alumno y de esta manera le entregue sus mensajes y noticias, liste sus cursos, etc. También posibilita que los participantes personalicen el entorno en idioma, posición de los menús, etc. Esto en cuanto al alumno, pero además debe ser posible que la organización que instala un LMS personalice el entorno añadiendo su imagen corporativa, colores y demás elementos característicos; finalmente, debe existir un sistema de ayuda *on-line* sobre el funcionamiento del entorno.

  Estas personalizaciones se quedan en general en un nivel externo y de presentación, dándose muy raras veces el caso de que se profundice en funcionalidades de adaptación al alumno, no sólo en cuanto a contenidos, sino también a la manera de presentar estos contenidos.

Además de estos grupos de funcionalidades genéricamente implementadas, aunque en muy diverso grado, en los diferentes sistemas de gestión del aprendizaje existen otras características importantes que tratan aspectos menos funcionales, y sí más técnicos desde el punto de vista de la implementación:

- **Integración del LMS con otros sistemas:** A pesar que los LMS ofrecen una amplia gama de posibilidades, puede resultar útil a algunas organizaciones la integración con otros sistemas, como otros LMS, bases de datos, entornos de creación de contenidos, etc. Estas integraciones suelen ser muy triviales en el sentido que no van más allá del intercambio de datos poco estructurados, por ejemplo, los datos de los alumnos. En cualquier caso, a veces, proporcionan el medio a través del cual se pueden construir módulos que se integran con los LMS.
- **Seguridad del LMS:** El entorno debe mantener y guardar la seguridad de los participantes y los contenidos. Se debe asegurar que nadie sin permiso entre en el sistema, para ello se utilizarán identificadores de entrada y contraseñas.
- **Fiabilidad del LMS:** Si se pretende que los conocimientos estén siempre disponibles, el entorno debe de ser fiable, no debe fallar ni ante ataques externos, ni ante errores internos.

Como señala Morrison [2004], un software LMS puede tener otras funcionalidades, como son:

- **Administración:** Debe permitir al administrador iniciar, dirigir, controlar y generar información de las clases, los estudiantes y los grupos. Un administrador puede configurar todas las opciones del sistema, controlar los costes de los cursos, establecer y modificar los campos definidos por los usuarios, especificar los requisitos de certificación y periodos de notificación (cuando el usuario deba ser avisado), gestionar los accesos, importar registros existentes y toda la información necesaria, añadir, borrar, modificar e importar toda la información de los cursos, controlar grupos de gestores, usar todas las herramientas de información, establecer las opciones de seguridad, etc.
- **Gestión "on-line":** Permitiendo a los profesores documentar, controlar e informar sobre las clases, los estudiantes y los grupos de estudiantes asignados a cada gestor "*on-line*", autorizado por el administrador. Un gestor "*on-line*" puede gestionar una serie de recursos asignados, como: cursos, estudiantes, grupos de estudiantes, control del uso de los cursos, ver e imprimir los informes de los estudiantes, monitorizar su progreso, asignarles cursos y currículos, añadir y modificar su información, etc.
- **Gestión de estudiante "on-line":** Se trata de un usuario de una herramienta, a través de la cual, el estudiante registrado en el sistema puede acceder a los cursos, ver las tareas, y toda la información referente al curso. Un estudiante puede emplear su navegador Web para acceder a una interfaz, simple e

intuitiva, donde ver el catálogo de cursos "*on-line*", ver información detallada de los cursos, planificaciones, ejecutar informes personales, acceder a las lecciones de los cursos, ver su información personal, cancelar el registro, etc.

- **Conectividad e Interoperabilidad:** Son dos conceptos que han de estar muy presentes cuando un LMS puede ofertar, gestionar y medir las actividades de un curso: ejecución, control, puntuación, ejecutar cualquier tipo de recurso electrónico del curso (presentaciones, archivos, vídeos, etc.) desde el navegador, descargar cursos, etc., dado que todas estas actividades suelen necesitar que se intercambien ficheros.
- **Notificación vía e-mail automática:** Sirve para informar de los próximos cursos, sobre los cursos completados, sobre matrículas, clases y cursos superados, para el envío de documentos necesarios, etc.
- **Gestión de base de datos embebida:** En ocasiones se precisa de una base de datos para las necesidades de algunos clientes, esta es generalmente embebida, aunque cabe la posibilidad de emplear una base de datos externa al sistema.

De esta manera el LMS maneja todas las tareas administrativas del e-learning, lo cual resulta muy útil, puesto que dichas tareas pueden ser verdaderamente complicadas cuando se trabaja con centenares de cursos y miles de estudiantes.

Desde el punto de vista técnico, un LMS es un software basado en un sistema servidor, que controla el e-learning. Al acceder los usuarios normalmente vía navegador Web, el sistema es sencillo de manejar.

Existen tres tipos fundamentales de LMS [Piskurich, 2003]:

- **Propietarios:** Son sistemas muy herméticos, y generalmente con poca interoperabilidad con otros componentes e-learning.
- **Basados en estándares:** Aquellos sistemas que soportan los estándares e-learning, proporcionando interoperabilidad, pero que se ven limitados por las propias carencias de los estándares que soportan.
- **Basados en sistemas de arquitecturas abiertas:** Proporcionan una interoperabilidad muy elevada gracias al conocimiento de su funcionamiento interno.

Como se aprecia en la Figura 2.1 los principales componentes de un LMS típico según Henderson [2003] son:

- **Portal Web:** El sitio Web al que los usuarios acceden para aprender. Se puede considerar como la "puerta principal" del LMS, que conduce al catálogo de los cursos disponibles y a las demás partes del LMS. En ocasiones también incluye noticias, anuncios y otra información.
- **Catálogo de cursos:** Se trata de una descripción de cada uno de los cursos e-learning que están disponibles en el sistema. Los alumnos pueden seleccionar aquellos cursos que deseen cursar.
- **Gestión de los alumnos y registros:** Se trata de la parte administrativa del LMS y gestiona los registros de los alumnos, a la vez que proporciona informes a los administradores.
- **Evaluaciones (preguntas, test):** Muchos LMS incluyen la funcionalidad de realizar test sobre los cursos, algunos incluyen evaluaciones de temas concretos, en las que los estudiantes pueden realizar evaluaciones "*on-line*" de los conocimientos adquiridos y mediante las mismas, obtener recomendaciones de cursos que pueden hacer, para mejorar en aquellas áreas en las que sus resultados fueron bajos.

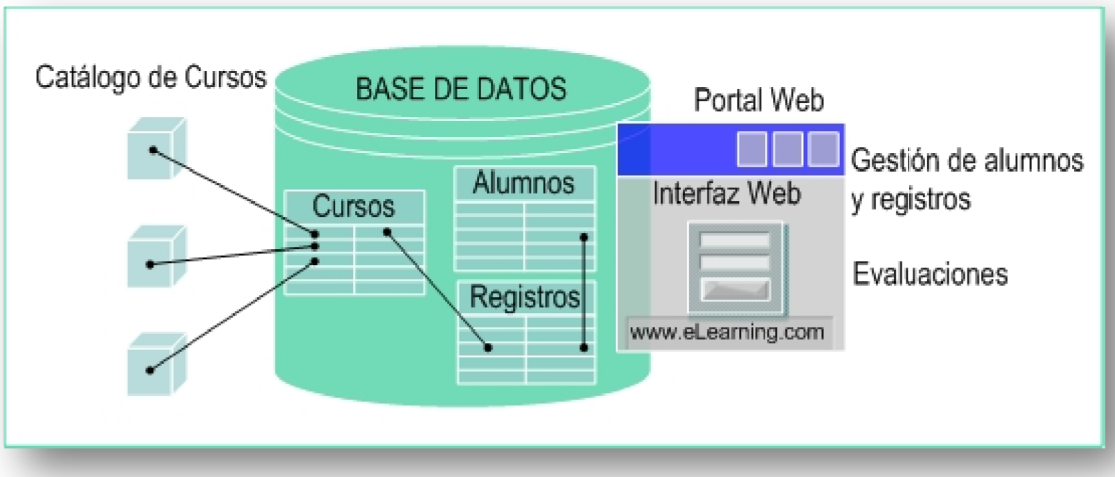


Figura 2.1 Esquema de componentes de un LMS

Learning Content Management Systems

La definición más sencilla de sistema de gestión de contenidos (LCMS) es la que describe a éste como un sistema que permite la creación, almacenamiento, gestión y desarrollo de

los contenidos e-learning bajo la forma de objetos de aprendizaje, para servir las necesidades de los individuos.

Los sistemas de gestión de contenidos, en adelante LCMS, aparecen por dos razones fundamentales: primero por la necesidad de algunos usuarios de LMS que precisaban crear y manipular contenidos, y en segundo lugar, por el rápido incremento de las prestaciones que se le exigían al sistema y para las cuales los LMS's no tenían la respuesta adecuada.

Los LMS y LCMS tienen una naturaleza complementaria, los segundos no fueron diseñados para reemplazar a los primeros, sino para proporcionar a los clientes que así lo requerían, la capacidad de gestionar también los contenidos de aprendizaje. Los LCMS facilitan a los profesores la división de los cursos en unidades cada vez más pequeñas (granulación), y de esta manera fomentan la reutilización de contenidos y simplifican la creación de los cursos.

La Tabla 2.2 muestra cómo los resultados establecen las diferentes funcionalidades de los dos tipos de sistemas:

	LMS	LCMS
Usuarios principales	Gestores del aprendizaje, Profesores, Administradores.	Desarrolladores de contenidos y diseñadores de contenidos educativos
Provee herramientas para la gestión primaria de...	Estudiantes	Contenidos educativos
Realiza gestión de las clases, del aprendizaje...	Si. Pero no siempre	No
Realización de informes del aprendizaje...	Es una de las tareas principales.	Es una tarea secundaria.
Colaboración con el estudiante	Si	Si
Guarda datos de los perfiles de los estudiantes	Si	No
Comparte los datos de los usuarios con otros sistemas	Si	No
Planificación de eventos	Si	No

Control de competencias, análisis de las habilidades	Si	Sí, en algunos casos
Capacidades de creación de contenidos	No	Si
Organización de contenidos reutilizables	No	Si
Creación de preguntas de evaluación y gestión de test	Si	Si
Pre-evaluación dinámica y enseñanza adaptable	No	Si
Herramientas de gestión de aprendizaje para el proceso de desarrollo de contenidos	No	Si
Entrega de los contenidos, proporcionando controles de navegación e interfaces para el alumno	No	Si

Tabla 2.2 Comparación de las funcionalidades de un LMS y un LCMS

Habitualmente los LCMS proporcionan módulos para la creación de contenidos, la creación de evaluaciones, publicación, administración, repositorio de datos, y un módulo para la personalización del entorno de trabajo.

El LCMS simplifica y acelera el proceso de creación de contenidos, facilitando a expertos en cualquier materia, pero no necesariamente expertos en el desarrollo de software, las funciones para diseñar, crear, distribuir y controlar la calidad del proceso de aprendizaje de una forma sencilla, rápida y eficiente. Es debido a esta funcionalidad por lo que se puede pensar que los LCMS disponen de potencial suficiente para emplearse como herramientas autónomas capaces de compartir y gestionar los conocimientos, reemplazando así, en cierto modo, al LMS. Pero los LMS y LCMS no son excluyentes, como muestra la Figura 2.2, sino complementarios. Han de trabajar conjuntamente para así intercambiar entre ambos la información, resultando un aprendizaje más enriquecedor para el alumno y una herramienta más completa para el administrador del e-learning. El LMS gestiona los grupos de usuarios, permitiendo a cada uno de ellos el acceso a los objetos apropiados almacenados y gestionados por el LCMS. A su vez, el LCMS también guarda los progresos individuales de cada alumno y sus notas en las evaluaciones, pasándole los datos al LMS para realizar informes.

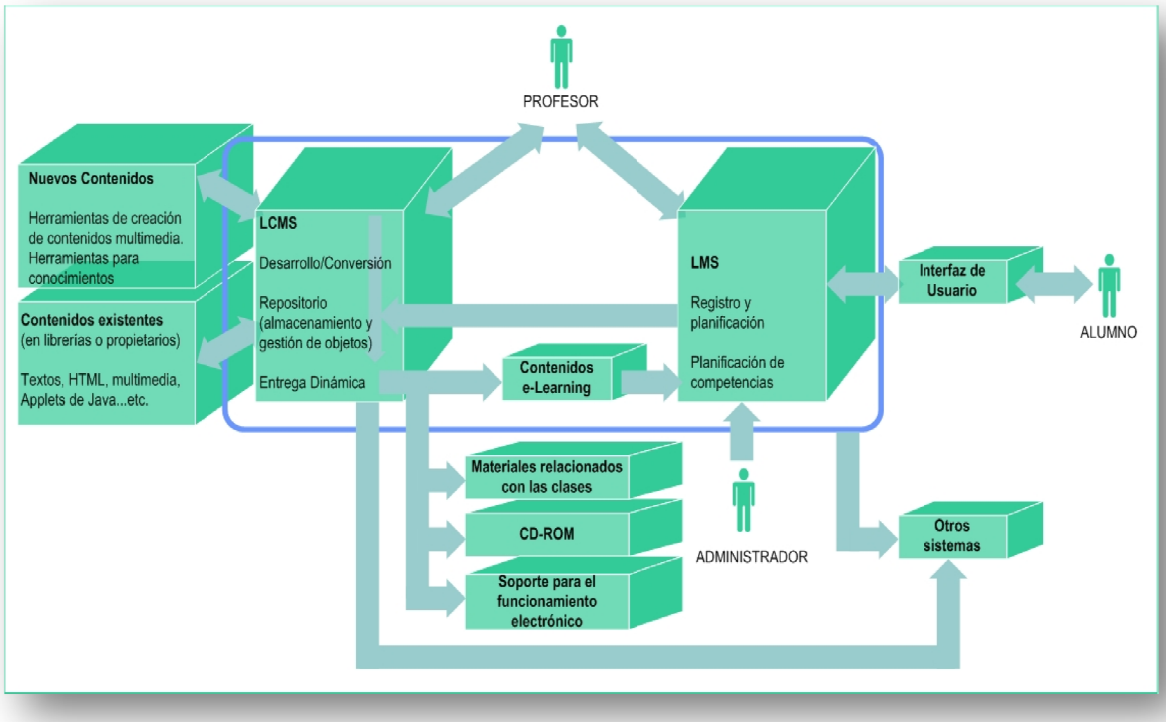


Figura 2.2 Integración entre LMS y LCMS

Los LCMS se construyen en torno a una base de datos central en la que se almacenan los elementos que constituyen los contenidos de los cursos; dichos elementos se presentan en formatos portables (reutilizables), y generalmente incluyen herramientas de creación y publicación de contenidos. Estos sistemas, que funcionan en un entorno Web, pueden crearse con una estructura y apariencia diferentes unos de otros; además, sus contenidos pueden ser fácilmente actualizados por usuarios sin conocimientos especiales en aspectos técnicos y de mantenimiento de entornos Web.

Dichos contenidos pueden ser usados por más de un sitio Web, de manera que se reduzca el esfuerzo del desarrollo, al poder reutilizar los mismos contenidos para diferentes cursos. Son estas posibilidades las que han popularizado el uso de las plataformas e-learning entre los profesores que han necesitado poner material de los cursos a disposición de los estudiantes (a distancia, por el método tradicional, o ambos tipos) de una manera rápida y fácil.

El siguiente diagrama de la Figura 2.3 muestra algunos de los componentes de un LCMS y como se relacionan unos con otros.

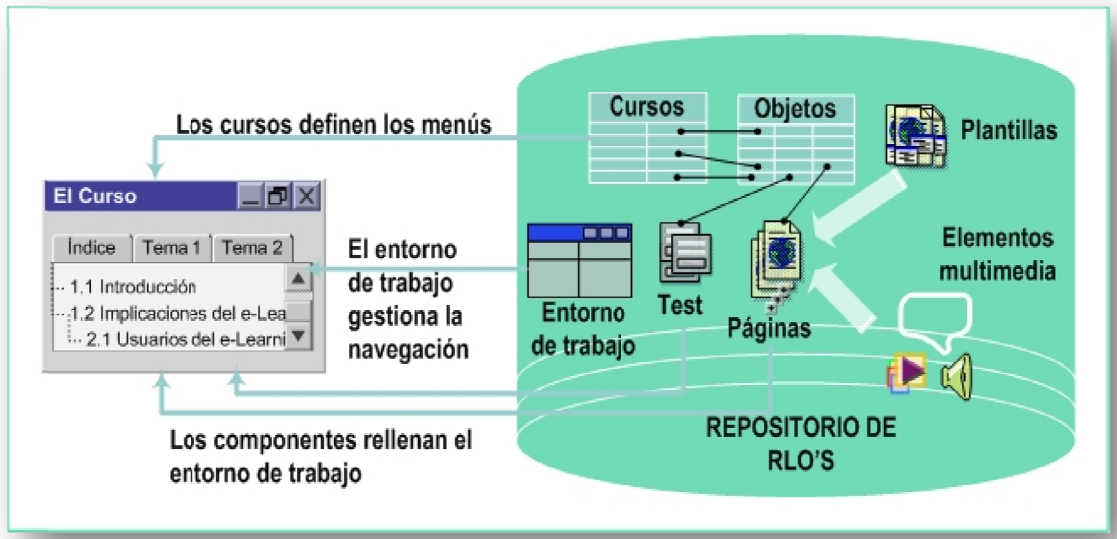


Figura 2.3 Componentes de un LCMS

Los creadores de contenidos generan los elementos multimedia, las evaluaciones y todos los componentes necesarios para la creación de un curso y los almacenan en el repositorio de objetos de aprendizaje (RLO).

Sirviéndose de plantillas ya existentes, o comenzando desde cero, se agrupan los componentes almacenados que se precisen para responder a un requerimiento de un usuario, esos grupos de componentes se etiquetan en términos de los objetivos buscados y de los contenidos que se necesitan para alcanzar dichos objetivos, constituyendo así la materia de los cursos. Las lecciones y los cursos se definen en términos de los objetos de aprendizaje que contienen.

Los creadores de contenidos deben también definir el entorno de trabajo, para controlar la navegación y proporcionar la interfaz de usuario del curso. Cuando se necesita un curso se realiza una copia del entorno de trabajo; mediante la definición del curso, se genera el menú correspondiente y los estudiantes seleccionan de dicho menú las páginas y cualesquiera otros componentes del curso, disponibles en el entorno de trabajo.

Por lo tanto, una plataforma LCMS, además de garantizar el control del proceso de aprendizaje, debe facilitar la creación, almacenamiento y reparto de los contenidos, ateniéndose a las siguientes características:

- Proporcionar herramientas sencillas que faciliten la creación de contenidos, en forma de aplicaciones o software de autor embebidas en el sistema, incluyendo editores para la creación de contenido Web, con el fin de eliminar la necesidad de manejar editores HTML o XML.
- Emplear sistemas flexibles de diseño y distribución de los recursos, que permitan adaptarse a las necesidades de la organización y a los diferentes sistemas y ritmos de aprendizaje de los usuarios.
- Posibilitar la reutilización de los objetos de aprendizaje; de hecho, cada elemento de aprendizaje debiera ser tratado como un objeto de aprendizaje reutilizable y mantenido a disposición de los profesores, quienes pueden aprovecharse de esta facilidad para emplear el mismo objeto en cursos diferentes, de manera que se minimice el esfuerzo de desarrollo, o actualización.
- Proporcionar herramientas para la administración del sistema que permita las matriculaciones, el seguimiento del aprendizaje, controlar el uso de los tiempos, el seguimiento de los usuarios, la adecuación de contenidos, etc.
- Proporcionar herramientas para la evaluación, tanto inicial como de la evolución de los aprendizajes que se produzcan a lo largo del curso, ya sea en lo que se refiere a los cursos en general, como a los objetos de aprendizaje en particular; el sistema debe proveer recursos suficientes para valorar los aprendizajes bajo distintos niveles de dificultad y diferentes modalidades de medición.
- Permitir la conectividad con otros LMS y la adecuación a los estándares más importantes.
- Disponer de herramientas para la comunicación y el aprendizaje en colaboración. Incluyendo recursos, tanto síncronos como asíncronos, que faciliten la comunicación sencilla entre iguales y con el profesorado; y, asimismo, recursos para el aprendizaje en colaboración que permita compartir conocimiento y realizar trabajos en grupo.
- Establecer mecanismos de seguridad y protección del conocimiento almacenado; dicha seguridad dependerá del uso de los privilegios de los diferentes usuarios y de las diversas funciones que los mismos desarrollan dentro de la organización, y afectará a las cargas y descargas de documentación, así como al acceso a la misma.
- Simplificar la migración de contenidos para facilitar la adaptación a las diferentes necesidades y escenarios de formación que se puedan presentar.

- Facilitar el proceso de instalación de manera que haga innecesarias las adaptaciones, localizaciones, personalización y demás tareas que encarecen el producto y retrasan ese proceso.

Viendo las características y funcionalidades de cada sistema, se puede decir que la principal diferencia de los LMS con respecto a los LCMS, radica en que para los primeros el elemento con el que trabajan es la totalidad de un curso ya existente, sin entrar en el detalle de los contenidos que constituyen dicho curso; mientras que para un LCMS su elemento de trabajo son dichos contenidos, vistos como componentes aislados de cursos, sea de cursos ya existentes o de cursos que podrán crearse en un futuro.

La necesidad de respetar esas características y proporcionar esas funcionalidades de forma satisfactoria en términos de eficiencia y seguridad, obligó a los desarrolladores a potenciar el uso generalizado de técnicas sin las cuales puede decirse que el e-learning no alcanza los objetivos para los que ha sido concebido.

Ya se ha hecho referencia, en páginas anteriores, al cumplimiento de estándares universales como condición necesaria para garantizar el éxito de una aplicación e-learning; igualmente hay otros dos aspectos que es imprescindible tomar en consideración cuando se estudia una plataforma, se trata de la capacidad de reutilización de los Objetos de Aprendizaje y la conveniencia de su almacenamiento en repositorios públicos (conceptos que serán detallados en el siguiente punto).

Satisfacer adecuadamente las necesidades de acceso, distribución y creación de contenidos de cursos e-learning, tal y como se viene planteando en esta tesis, supone el empleo de Objetos de Aprendizaje Reutilizables y de Repositorios. Dada la importancia que esos conceptos tienen en la actividad de desarrollo de plataformas e-learning, se exponen seguidamente algunas ideas básicas para facilitar su comprensión.

Objetos de aprendizaje reutilizables

El enfoque de Objeto de Aprendizaje Reutilizable ("*Reusable Learning Objects*" o RLO) vino impuesto por la práctica de los desarrolladores de cursos, quienes constataron que,

frecuentemente, existía una considerable redundancia, o solapamiento, en el contenido de los cursos, pese a que éstos iban dirigidos a usuarios con perfiles diferentes y con diferentes objetivos de aprendizaje. Esa redundancia suponía una multiplicación del esfuerzo de desarrollo y encarecía considerablemente los productos finales, a la par que prolongaba los tiempos de desarrollo y dificultaba los objetivos de intercomunicación, migración, escalabilidad, actualización, etc.

Al definir los Objetos de Aprendizaje Reutilizables como los componentes elementales en que puede descomponerse el contenido "formacional" de un curso, se está hablando de la fragmentación de contenidos (granulación) en "pedazos autónomos de material de aprendizaje" [Patron, 2003]. El grado que se alcance en esa fragmentación o granulación de contenidos (respetando siempre la exigencia de autonomía aludida por Patron), indicará, a su vez, el grado de la posibilidad de reutilización de dichos objetos. Tanto mayor será ese grado de reutilización, tanto mayor será la eficacia del desarrollo y su idoneidad para satisfacer las necesidades exigidas al sistema.

La fragmentación de los contenidos trajo como consecuencia la multiplicación de los RLO's que era necesario manejar; a título de ejemplo de este hecho, se suele emplear el juego de construcción Lego, donde puede apreciarse la diferencia entre construir un castillo como un todo (prácticamente una sola pieza), y hacerlo mediante las piezas del Lego, en el que es necesario manejar un número considerablemente mayor de piezas elementales; en contrapartida, con las piezas elementales del Lego se pueden hacer castillos de diferentes formas, o incluso otro tipo de construcciones que nada tengan que ver con un castillo. Una postura razonable en el grado de fragmentación de los contenidos, ha de buscar el equilibrio entre las ventajas obtenidas por la granulación y las dificultades de manejar un número muy elevado de RLO's.

La dificultad de manejo de los RLO's, principalmente en lo que atañe a su acceso y ubicación, obligó a la creación, para cada RLO, de metadatos o metainformación, asociado indisolublemente al objeto, por lo que puede decirse que un RLO está compuesto de dos partes: por un lado el contenido del objeto consistente en la información didáctica que éste aportará al curso en que habrá de integrarse; y por otro lado la meta-información que proporciona datos referentes a lo que el objeto de aprendizaje encierra en sí mismo, y en ella figuran por ejemplo: título, palabras clave relativas al contenido, a los objetivos del mismo, así como nivel, prerrequisitos, evaluación, autor, fecha, lenguaje, versión etc. La meta-información se trata, a su vez,

como un elemento independiente que puede ser ejecutado en una plataforma Web estándar o sistema operativo, no necesita de "*plug-ins*" o de la instalación de aplicaciones especiales. La razón de ser de la meta-información es permitir la localización de los objetos de aprendizaje para integrarlos en un curso determinado, eso se consigue gracias a la información, estructurada en una serie de campos, que se obtiene de la meta-información.

Repositorios digitales

Para muchos autores la idea de repositorio es intrínseca a los objetos de aprendizaje. No es posible pensar en objetos de aprendizaje si no se los concibe albergados en repositorios.

Los repositorios digitales, en el sentido más amplio de la definición, se emplean para almacenar cualquier tipo de material digital. No obstante, los repositorios digitales para objetos de aprendizaje son mucho más complejos en términos de qué es necesario almacenar o cómo se almacenará. El término "objeto de aprendizaje" se refiere a cualquier elemento digital que puede ser usado para permitir el aprendizaje o la enseñanza. Se puede tratar de cualquier tipo de objeto, como: gráficos, imágenes simples o videos, pasando por documentos, exámenes complejos o agrupaciones de objetos. Cada elemento ha de tener su propia identidad y ser localizable, por ello, los criterios de búsqueda de esos objetos han de considerar bastante más que títulos, autores o palabras clave.

De hecho, un examen de la granulación de los objetos de aprendizaje que se almacenan en un repositorio digital es un buen punto de inicio para determinar la complejidad del manejo de dicho repositorio.

Si los objetos son cursos completos, o módulos substanciales de los cursos, entonces el repositorio no es más que un portal, desde el cual acceder al material. Lo que hace de los repositorios digitales de objetos de aprendizaje algo más que un simple portal, es la capacidad de encontrar los objetos de aprendizaje y darles nuevos usos. El propósito de un repositorio digital de objetos de aprendizaje no es simplemente almacenar y distribuir dichos objetos, sino permitir que los mismos sean compartidos por distintos estudiantes

y, sobre todo, facilitar su reutilización en diferentes actividades formativas. La ventaja de los repositorios desde el punto de vista de los usuarios es el hecho de tener acceso a los contenidos depositados en el repositorio/almacén; y para usar esos servicios ha de pagar el coste de someterse a unos procedimientos, normas, y estándares, cuya aplicación va dirigida a potenciar uno de los aspectos más interesantes y fructíferos del repositorio, como es la creación de objetos de aprendizaje reutilizables.

Se puede argumentar que un repositorio digital es una biblioteca digital en la que se realizan intercambios entre sus contenidos. Pero tal argumentación no es del todo correcta, ya que los gestores de una biblioteca digital tienen como principal responsabilidad establecer donde colocar los objetos, para que el usuario pueda tener acceso a los mismos; mientras que los repositorios han de ser vistos, principalmente, como almacenes de objetos que son puestos a disposición de diferentes usuarios, en el momento y lugar que éstos los soliciten, permitiendo asimismo que esos usuarios no sólo se sirvan de la información contenida en dichos objetos, integrándola en un curso determinado, sino que puedan actuar sobre ella modificándola, ampliándola, incorporándole mejoras; actividades todas ellas que, al estar sometidas a las normas y estándares a las que antes se aludió, permiten obtener e incorporar al repositorio objetos de aprendizaje con diferentes contenidos, pero de características homogéneas en cuanto a su composición y estructura, lo cual permite su reutilización cuantas veces sea preciso de forma cómoda y sencilla.

El empleo del término repositorio con preferencia al de biblioteca digital, tiene como objetivo enfatizar el hecho de que, en el primero, al permitir la aportación de mejoras por parte de los usuarios, se consigue que diferentes personas puedan compartir los objetos de aprendizaje para la creación de cursos que, desde el punto de vista de la formación proporcionada, tengan objetivos diferentes.

Existen dos tipos de repositorios digitales de objetos de aprendizaje, según Downes [2002]:

- Aquellos que contienen tanto los objetos de aprendizaje con su contenido de información, como los metadatos de dichos objetos de aprendizaje.
- Aquellos que contienen sólo los metadatos de los objetos de aprendizaje, mientras que los objetos de aprendizaje con su contenido de información se encuentran almacenados en otra ubicación en la que el repositorio puede

localizarlos, a partir de la información contenida en los metadatos y mediante el empleo de una herramienta adecuada para ello.

La mayor parte de los repositorios suelen ser autónomos. Esto es, funcionan como portales a los que se accede mediante una interfaz basada en Web, proporcionando un mecanismo de búsqueda y una lista de categorías donde buscar, sin necesidad de emplear otro producto. Algunos otros repositorios digitales de objetos de aprendizaje funcionan como una base de datos conectada a otro producto (generalmente una plataforma).

Para definir los posibles usos de los repositorios digitales es útil considerar primero quien los usará y qué procesos educativos se beneficiarán de ellos. Los usuarios de un repositorio digital de objetos de aprendizaje son generalmente, aunque no siempre, los profesores, ya que normalmente son los encargados de producir los cursos. El repositorio digital no será afectado por el uso pedagógico del material y actuará simplemente como un almacén en el que no tiene influencia el dónde o quién emplea los recursos almacenados, o el propósito con que estos son empleados.

Aquellos que usan repositorios digitales no deben preocuparse de la arquitectura interna de los mismos. Es útil, sin embargo, seguir las especificaciones de "*IMS Digital Repository Interoperability Working Group*" (DRIWG) [IMS, 2003a] que permiten a los repositorios digitales interactuar entre sí ignorando su arquitectura interna. En dichas especificaciones se definen los repositorios digitales en términos de lo que pueden contener.

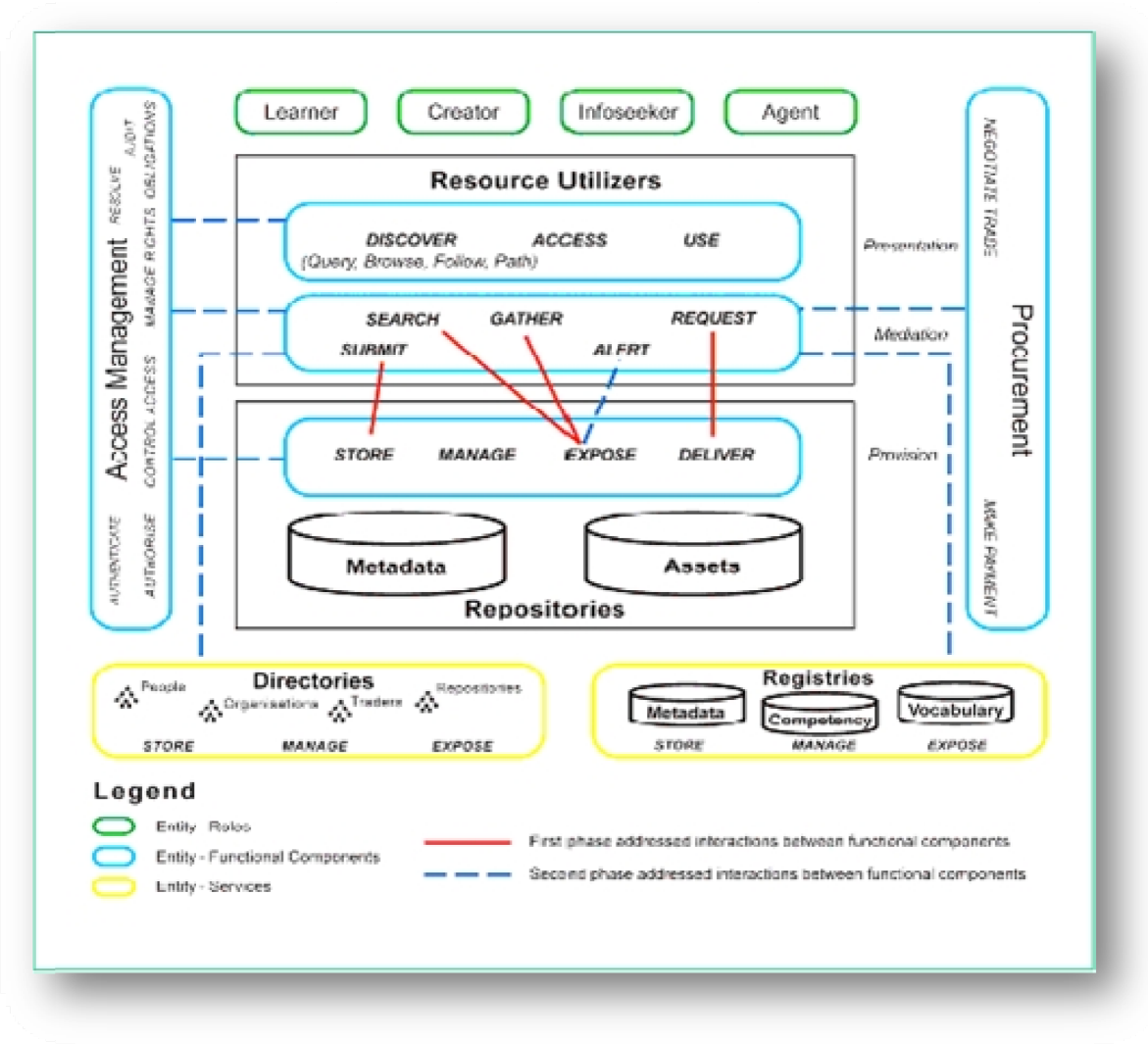


Figura 2.4 Contexto del IMS Digital Repository Interoperability [IMS, 2003a]

La Figura 2.4 muestra la estructura interna propuesta por IMS; en la parte inferior se representan los repositorios digitales (de cualquier tipo) de objetos de aprendizaje, mientras que en la parte superior se encuentran los "usuarios de recursos" ("*resource utilizers*"), que representan cualquier persona o sistema que emplee esos recursos.

El DRIWG ha hecho corresponder las tareas más comúnmente empleadas por los usuarios con las respuestas que debe proporcionar el repositorio digital. Esto se detalla en la Tabla 2.3, que además muestra tareas adicionales que los usuarios pueden demandar.

TAREA DEMANDADA POR EL USUARIO	RESPUESTA DEL REPOSITORIO DIGITAL
Buscar, reunir, alertar, hojear (discover, gather, alert, browse)	Presentar los datos requeridos por el usuario
Configurar el repositorio	Cambiar la Interfaz
Solicitar contenidos	Proporcionar los contenidos solicitados
Publicar	Almacenar
Entregar contenidos (desde el repositorio)	Almacenar (en otro repositorio)

Tabla 2.3 Respuestas de un repositorio digital

En el empleo de los repositorios digitales, los usuarios pueden tener distintos roles en distintos momentos. Los roles que los repositorios digitales deben soportar son:

- **Bibliotecario:** Es el responsable de mantener el sistema de clasificación del repositorio y asegurar la integridad de los metadatos. Un bibliotecario tiene los privilegios suficientes para editar y crear los metadatos, así como también para enlazar y desenlazar los objetos con los nodos del sistema de clasificación.
- **Contribuyente:** Es aquella persona que introduce los recursos (objetos de aprendizaje) en el repositorio; esta actividad forma parte de las prestaciones primordiales exigidas al repositorio, y tiene la ventaja de que quiénes crean los objetos son los mismos que crean los metadatos asociados, consiguiendo con ello que las descripciones sean más precisas.
- **Prestatario:** Aquel que toma prestados objetos de los repositorios de manera regular, y que suele personalizar la interfaz con la que interactúa con el repositorio; así como preservar y dejar constancia de las búsquedas realizadas en las sesiones con el repositorio.
- **Usuario casual:** En algunas circunstancias, los usuarios invitados pueden tener permisos para buscar, explorar o descargar objetos del repositorio pero sin tener su espacio personalizado propio. Generalmente los invitados no tienen que estar registrados como usuarios habituales.
- **Administrador:** Tiene la responsabilidad de gestionar los usuarios del repositorio, crear los usuarios nuevos, y borrar a los que ya no usen el repositorio. Es también el encargado de establecer las prerrogativas de acceso de que disponen los usuarios casuales.

- **Agente Software:** Es el nombre que reciben aquellos entornos visuales de aprendizaje u otros repositorios digitales, que pueden consultar el repositorio y descargar elementos.

En muchas ocasiones se da el caso de que el contribuyente y el prestatario es la misma persona, generalmente eso ocurre cuando el repositorio se emplea para compartir recursos entre los componentes de un grupo que generan recursos de aprendizaje y los usan para construir cursos e-learning.

Además de estos roles individuales también se pueden crear grupos de usuarios que trabajen conjuntamente, por ejemplo cuando un grupo asume la responsabilidad compartida de crear e impartir un curso.

2.2. ESTÁNDARES

Los estándares son acuerdos internacionales documentados o normas establecidas por consenso mundial. Contienen las especificaciones técnicas y de calidad que deben reunir todos los productos y servicios para cumplir satisfactoriamente con las necesidades para las que han sido creados, y para poder competir internacionalmente en condiciones de igualdad. El objetivo primordial al crear un estándar es impedir que en el mercado se impongan determinadas tecnologías ofrecidas por las empresas de un ámbito industrial concreto, evitando así que imperen intereses económicos privados.

Las diferentes organizaciones internacionales de estandarización ofrecen definiciones oficiales de lo que es un estándar. De acuerdo a la organización internacional de normalización (ISO), estándar se define como lo que "*contribuye para hacer la vida más fácil, y para incrementar la confiabilidad y efectividad de los bienes y servicios que utilizamos*" [ISO, 2009]. También, según ISO, se trata de "acuerdos documentados que contienen especificaciones técnicas u otros criterios, para ser utilizados constantemente como reglas o definiciones de características, para asegurar qué materiales, productos, procesos y servicios son adecuados para sus propósitos".

Por su parte, el BSI (*British Standard Institute*), describe un estándar como "*una especificación publicada que establece un lenguaje común, y contiene una técnica u otro criterio, que está diseñado para ser usado constantemente, como una regla o una definición*" [BSI, 2009].

Los estándares ofrecen:

- Una base de comparación.
- Una medida de la calidad, cantidad o nivel.
- Un consenso de opiniones entre individuos, grupos u organizaciones.

Los estándares son desarrollados por organizaciones oficiales con ánimo de evitar que intereses privados determinen normas y garantizar la comunicación entre los diferentes dispositivos y/o plataformas con los denominados estándares oficiales o de jure. En contraposición a éstos estándares se encuentran los designados de facto, los cuales sin estar impuestos por ninguna organización, y a veces sin estar emitidos por

ninguna norma, hacen que se conviertan en estándares al usarse de manera repetitiva. El estado ideal de un estándar es cuando un estándar de jure es también de facto.

Existe cierta confusión sobre los términos estándares y especificación. La diferencia fundamental entre ambos es que las especificaciones son desarrolladas por comités no acreditados. Algunos de los más conocidos son: IETF (*Internet Engineering Task Force*), W3C (*World Wide Web Consortium*), OMG (*Object Management Group*), IMS (*Instructional Management Systems*). Por el contrario, un estándar es una especificación desarrollada y acreditada por comités de estandarización específicos. Ejemplo de este tipo de comités incluye a: IEEE (*Institute of Electrical and Electronics Engineers*), ISO, ANSI (*American National Standards Institute*), BSI, AENOR (Agencia Española de Normalización), etc. Incluso, en algunas industrias, el producto no puede ser vendido hasta que no recibe la certificación necesaria o es aprobada por el gobierno pertinente, como en el caso de los productos y servicios eléctricos o electrónicos, acreditados por IEEE.

Por lo tanto, una especificación es una descripción documentada de una tecnología. Las especificaciones son recomendaciones técnicas y de calidad que no se han adoptado de manera oficial en un ámbito tecnológico o de cualquier otra índole. Es decir, no es obligatorio su uso. La mayoría de los estándares tampoco son obligatorios, ya que solo lo son cuando los impone la ley.

El proceso de estandarización

El proceso de elaboración de un estándar es similar al de creación y aprobación de las leyes: una vez se ha realizado el grueso del trabajo, este debe ser ratificado por un organismo oficial. Puede parecer un proceso lento y poco efectivo, pero hay que tener en cuenta que el éxito de un estándar radica en su nivel de aceptación, por lo que un grupo de estandarización debe ser un organismo que se encargue de recopilar requisitos de múltiples fuentes y elabore con ellos una especificación consensuada [Sancho, 2002].

Algunas especificaciones acaban convirtiéndose en estándar, lo que significa que han recibido una acreditación oficial.

Para que una especificación evolucione hacia estándar debe pasar por un proceso, el cual tiene sus fases. Entre las principales fases en el diseño y desarrollo de estándares, y según Masie [2002], se pueden mencionar las siguientes (Figura 2.5):

- **Fase 1- Requisitos**: El proceso comienza con grupos denominados consorcios y formados por gobiernos, empresas, individuos, académicos, etc. Se recogen requisitos e ideas y experiencias de la comunidad de usuarios, instituciones académicas, la industria y demás entes involucrados.
- **Fase 2- Especificaciones:** Expertos en realizar especificaciones preparan un borrador técnico del tema en cuestión, donde se presentan las principales ideas de la especificación que se desea desarrollar.
- **Fase 3- Prueba y uso**: Se desarrollan modelos de referencias y documentos específicos para ser usados y validados por grupos específicos de usuarios.
- **Fase 4- Estándar:** Si la especificación generada es válida y ampliamente aceptada después del período de prueba, puede ser enviada a cuerpos especializados para someterlos a su aprobación como estándar. Algunas oficinas de acreditación en e-learning son IEEE LTSC (Learning Technology Standards Committee), ANSI, o el CEN/ISSS (European Committee for Standardization / Information Society Standardization System). Una vez aprobada la especificación será reconocida como un estándar (por ejemplo, ISO) aprobado.

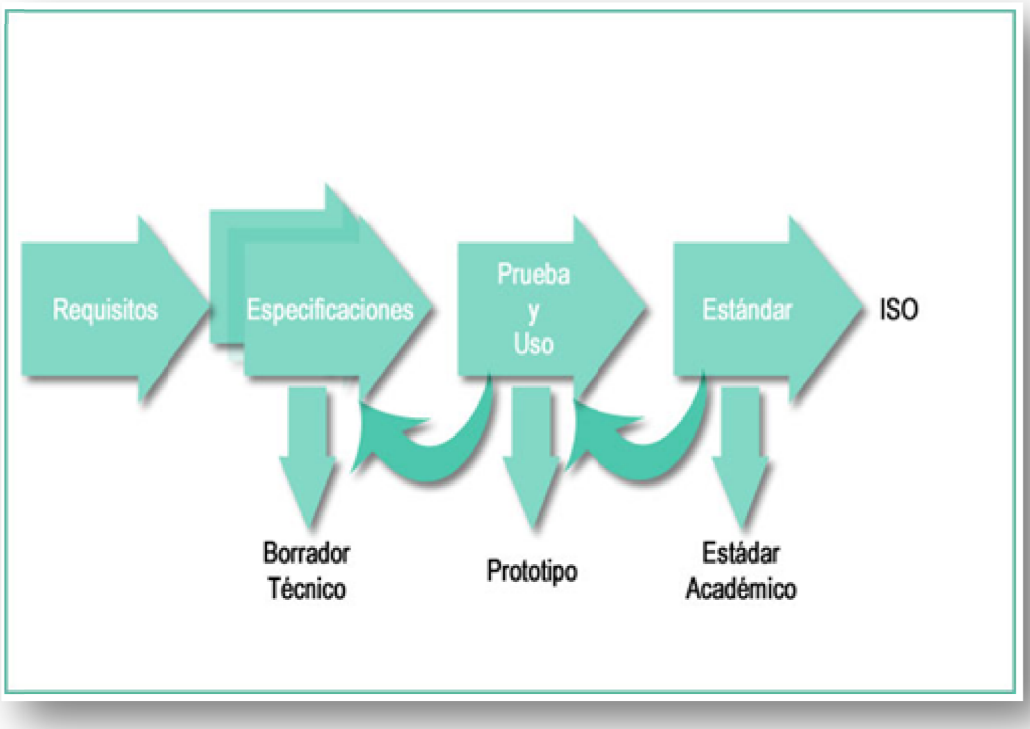


Figura 2.5 Proceso de creación de estándares

Estándares de e-learning

Sin duda, el mayor problema que aborda la industria del e-learning en la actualidad es la ausencia de unas metodologías técnicas, documentales y psicopedagógicas comunes y aceptadas, que garanticen los objetivos de accesibilidad, interoperabilidad, durabilidad y reutilización de los materiales curriculares encontrados en las redes.

Hay estándares genéricos apoyados por diferentes organismos que, desde hace años, establecieron sus propias especificaciones. Actualmente se está produciendo una convergencia hacia estándares comunes e intercambiables que soportan la definición de recomendaciones y nuevos estándares para campos de actividad específicos como en el caso del e-learning.

Las principales razones que impulsan la creación de estándares en el área de e-learning son según Rehak [2003]:

- Protección de la inversión ante quiebra de proveedores.
- Portabilidad de contenidos de cursos entre diferentes tecnologías de adiestramiento y/o plataformas e-learning.
- Integración de iniciativas e-learning con sistemas de Recursos Humanos Corporativos, Sistemas de Control o Administración de gestión académica.
- Integración de plataformas e-learning en la infraestructura tecnológica existente.
- En definitiva, la mejora del e-learning.

Por otra parte, Hodgins [2001], afirma que los objetivos que persiguen cumplir los estándares e-learning son:

- **Durabilidad:** Que la tecnología desarrollada con el estándar evite la obsolescencia de los cursos.
- **Interoperabilidad:** Que se pueda intercambiar información a través de una amplia variedad de LMS o LCMS.
- **Accesibilidad:** Que permita que todas las personas puedan utilizar los recursos independientemente de sus capacidades técnicas o físicas.
- **Reutilización:** Que los distintos cursos y objetos de aprendizaje puedan ser reutilizados con diferentes herramientas y en distintas plataformas.

- **Adaptabilidad:** Los estándares se refieren al hecho de poder facilitar la adaptación o personalización del entorno de aprendizaje.
- **Productividad:** Si los proveedores de tecnología e-learning desarrollan sus productos siguiendo estándares comúnmente aceptados, la efectividad de e-learning se incrementa significativamente y el tiempo y costos se reducen.

Tipos de estándares de e-learning

En el mercado existen tanto LMS como LCMS de muchos fabricantes distintos. Por ello se hace necesaria una normativa que compatibilice los distintos sistemas y cursos a fin de lograr dos objetivos:

- Que un curso de cualquier fabricante pueda ser cargado en cualquier LMS de otro fabricante.
- Que los resultados de la actividad de los usuarios en el curso puedan ser registrados por el LMS.

Como se puede ver en la Figura 2.6, los distintos estándares que se desarrollan hoy en día para la industria del e-learning se pueden clasificar en los siguientes tipos:

- **Sobre el Contenido o Curso**: Estructuras de los contenidos, empaquetamiento de contenidos, seguimiento de los resultados.
- **Sobre el Alumno**: Almacenamiento e intercambio de información del alumno, competencias (habilidades) del alumno, privacidad y seguridad.
- **Sobre la Interoperabilidad:** Integración de componentes del LMS, interoperabilidad entre múltiples LMS.

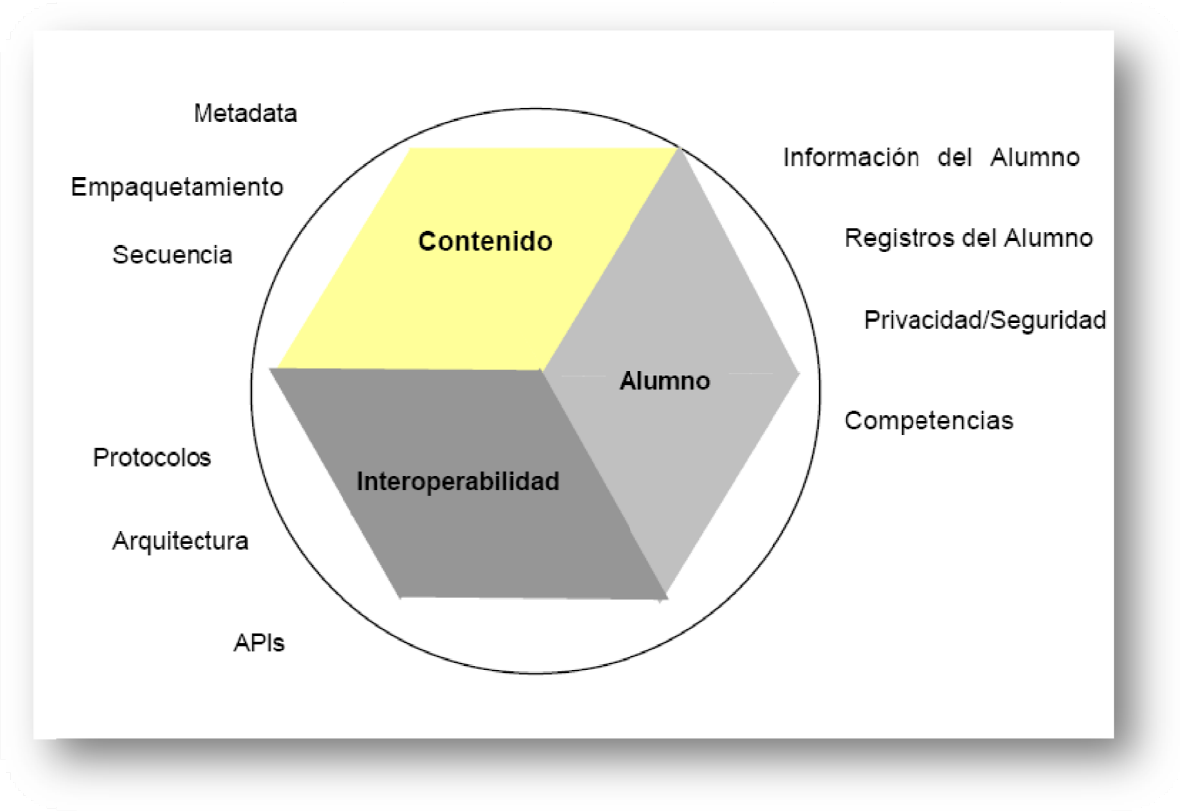


Figura 2.6 Áreas afectadas por los estándares de e-learning

Un estándar de e-learning se refiere a un conjunto de reglas en común para las compañías dedicadas a la tecnología e-learning. Estas reglas especifican cómo los fabricantes pueden construir cursos *on-line* y las plataformas sobre las cuales son impartidos estos cursos de tal manera de que puedan interactuar unas con otras. Estas reglas proveen modelos comunes de información para cursos e-learning y plataformas LMS, que básicamente permiten a los sistemas y a los cursos compartir datos o comunicarse con otros. Esto también ofrece la posibilidad de incorporar contenidos de distintos proveedores en un solo programa de estudios.

Estas reglas, además, definen un modelo de empaquetamiento estándar para los contenidos. Los contenidos pueden ser empaquetados como objetos de aprendizaje (*learning objects* o LO), de tal forma que permita a los desarrolladores crear contenidos que puedan ser fácilmente reutilizados e integrados en distintos cursos.

Si se consigue la estandarización de esta tecnología, desarrolladores de cursos *on-line* y constructores de componentes y plataformas e-learning beneficiarían a toda la comunidad de usuarios, ya que además de facilitar la interoperabilidad de componentes,

preservaría las inversiones que se realizaran en este campo. La vida de los cursos *on-line* se vería incrementada al poder intercambiarse cursos virtuales entre diferentes plataformas, sin la necesidad de realizar costosas modificaciones.

Los estándares han iniciado el camino hacia una forma cómoda y viable de empaquetar los recursos y contenidos, tanto para los estudiantes que cambian de sistema, los docentes que utilizan en distintos contextos estos materiales y los desarrolladores que tienen que construir nuevas herramientas y mejorar las vigentes.

Beneficios de los estándares de e-learning

Los estándares en e-learning no son totalmente sólidos ni están definidos en estos momentos. La madurez de la industria presionará por estándares claros y bien definidos. Existe una gran confusión en la proliferación de consorcios, especificaciones e individuos que desean ganar terreno en la nueva industria que se vislumbra.

Salvaguardar las inversiones realizadas, garantizar la interoperabilidad de los componentes tecnológicos que se adquieran, y el acceso a la información desde diferentes puntos de vista, es una tarea prioritaria para las personas involucradas en el e-learning.

Las instituciones educativas deben estar atentas a las iniciativas de estandarización de su tecnología, ya que sería muy costoso quedar con contenido aislado en un mundo cada vez más interconectado y que clama por la colaboración institucional como mecanismo para garantizar una educación de calidad.

Una industria que nace y carece de estándares es atractiva para incorporar nuevas maneras y formas de trabajar sistematizadas y organizadas. Por ello, educarse y fomentar la cultura de los estándares es imperativo, la experimentación debe dar paso a la organización y consolidación de una industria que nace y se fortalece cada día.

No sólo el uso y aplicación de estándares es propicio para el desarrollo de contenidos y materiales educacionales. Es de igual importancia, la utilización de estándares en los llamados portales educacionales. Un portal, la puerta virtual de acceso a la institución, debe reflejar consistencia y organización coherentes que guíen a los visitantes a las distintas instancias de información a las cuales deseen acceder. Como consecuencia, una imagen institucional sería de mayor beneficio para los visitantes del portal, que múltiples imágenes esparcidas a lo largo del mismo.

Los estándares e-learning proporcionan beneficios multifacéticos, esto incluye instituciones académicas, corporaciones, individuos y a la industria en general. He aquí algunos casos:

- **La industria e-learning como un todo**. La interoperabilidad entre diferentes componentes tecnológicos de e-learning elimina temores de inversión en la tecnología, al mismo tiempo que incentiva la adopción más generalizada del e-learning, lo cual facilita el desarrollo de la industria como un todo.
- **Proveedores de tecnología**. Con un sistema de estándares, los proveedores pueden ver expandidos sus mercados. Los contenidos y las plataformas basadas en estándares son más sostenibles a largo plazo. Los proveedores de contenido podrán fácilmente reutilizar contenidos entre diferentes programas. De igual manera, las herramientas estándares facilitan el desarrollo de contenidos y de nuevas herramientas.
- **Instituciones Académicas**. Compartir contenidos de cursos será mucho más fácil para profesores. Teniendo como estándar un navegador de Internet, los estudiantes y los profesores podrán fácilmente intercambiar información. Los estándares e-learning ayudan a preservar el capital invertido en tecnología y desarrollo de profesores. Transferir contenidos y evaluaciones entre instituciones será mucho más sencillo.
- **Corporaciones**. El poder de adquirir una gran gama de contenidos y que puedan funcionar correctamente en cualquier plataforma expande las potencialidades de formación de las empresas. La rapidez de puesta en marcha de cursos y programas enriquece los programas de formación corporativos. Todo esto trae consigo una mejor rentabilidad de la inversión realizada en e-learning.
- **Individuos**. Personas independientes tendrán acceso a mucho más conocimiento en diferentes formatos y lenguajes, esto conlleva a una reducción en costes de formación.

Futuro de los estándares de e-learning

La tendencia actual (principios de 2009) que están siguiendo las principales organizaciones de e-learning es obtener especificaciones relacionadas con las siguientes temáticas:

1. Repositorio de Contenidos: Las organizaciones están orientando su esfuerzo en desarrollar estándares de contenidos e-learning. **El principal objetivo es tener repositorios de objetos de aprendizaje reutilizables, de tal manera que puedan ser agrupados en objetos de aprendizaje adaptables y expedidos por cualquier plataforma e-learning. Sin embargo, uno de los mayores problemas al que se enfrenta hoy en día la industria del e-learning es la interoperabilidad entre distintos sistemas de aprendizaje que quieran compartir sus contenidos.**
2. Internacionalización y Localización: Los distintos grupos que están desarrollando especificaciones para e-learning participan de forma activa en todo el mundo, y cada día existe una mayor colaboración entre ellos. Esto genera dos desafíos: la creación de estándares "culturalmente" neutrales (internacionalización), y la adaptación de los estándares a las necesidades locales (localización).
3. Programas de certificación: Existe un creciente énfasis en crear test de compatibilidad y programas de certificación. ADL está trabajando en un programa de certificación. Actualmente sólo existen los programas de certificación de AICC.
4. Arquitectura: La industria del e-learning ha estado creciendo sin tener una clara visión de los componentes de un sistema de e-learning y de la forma en que interactúan. **La necesidad de definir una arquitectura global es crítica para la evolución del desarrollo de estándares.**

2.2.1. Organizaciones de estandarización e-learning

La implantación y difusión de estándares ha sido el medio de generalización de las aplicaciones en Internet y de la extensión de la propia red. No se podría entender la generalización del Web sin la definición del protocolo estandarizado HTTP (*HyperText Transfer Protocol*), del lenguaje HTML (*HyperText Markup Language*), o la de la propia Internet sin el protocolo TCP/IP (*Transmission Control Protocol/Internet Protocol*). El desarrollo de recomendaciones y estándares por parte de la World Wide Web Consortium, como XML (*eXentesible Mark up Language*), derivado de SGML (*Standard Generalized Markup Language*), meta lenguaje de marca con el cual se pueden crear lenguajes específicos, supone un nuevo empuje para la creación de contenidos en Internet.

Estos estándares genéricos han sido apoyados por diferentes organismos que, desde hace años establecieron sus propias especificaciones. Actualmente se está produciendo una convergencia hacia estándares comunes e intercambiables que soportan la definición de recomendaciones y nuevos estándares para campos de actividad específicos como el e-learning.

Los organismos de estandarización sobre e-learning más importantes son:

- AICC (*Aviation Industry Computer Based Training Committee*).
- IEEE (*Institute of Electrical and Electronic Engineers*) con el grupo de trabajo LTSC (*Learning Technology Standards Committee*).
- IMS (*Instructional Management System*) Global Learning Consortium.
- ADL (*Advanced Distributed Learning*).
- ARIADNE (*Alliance of Remote Instructional Authoring and Distribution Networks for Europe*).
- ISO (*International Standards Organization*), con el subcomité SC36 denominado "*Information Technology for Learning, Education and Training*".

Estos cubren el espectro de necesidades de definición en el entorno del e-learning: Estructura de cursos, contenidos reutilizables, metadatos, arquitecturas de plataformas e intercambio de datos. Los metadatos caracterizan a los datos y las aplicaciones, describiendo el contenido; los principales usos son catalogar, organizar, mantener los datos e intercambiarlos.

A manera de resumen, a continuación se presenta la Tabla 2.4 con las principales especificaciones relacionadas con los contenidos para e-learning:

ESPECIFICACIONES	ORGANIZACIÓN	DESCRIPCIÓN
Runtime Communication	ADL, también está siendo estandarizada por IEEE	APIs para la comunicación entre LMS y SCOs
CMI Data Model	AICC, adoptada también por AADL y estandarizada por IEEE	Define vocabulario y respuestas para la comunicación entre LMS y SCOs
Learning Object Metadata	IEEE, adaptada también por IMS, ADL e ISO	Define categorías usadas para describir los contenidos de aprendizaje
Aggregation Model	IMS, adaptado también por ADL	Indica cómo empaquetar los contenidos de un curso

Tabla 2.4 Principales especificaciones relacionadas con los contenidos para e-learning

Estos organismos están trabajando conjuntamente en la elaboración de estándares y especificaciones para el diseño de entornos tecnológicos en lo relativo al proceso de enseñanza mediante Internet; así pues se procede a detallar cómo está ayudando cada organización en el proceso de estandarización oficial.

IEEE

El IEEE (*Institute of Electrical and Electronics Engineers*) [IEEE, 2009] es un organismo que elabora normas para el instituto de estandarización americano (ANSI), muchas de las cuales se convierten posteriormente en estándares ISO. Dentro del IEEE se ha creado el comité LTSC (*Learning Technologies Standards Committee)*, encargado de preparar normas técnicas, prácticas y guías recomendadas para el uso informático de componentes y sistemas de educación y de formación, en concreto, los componentes software, las herramientas, las tecnologías y los métodos de diseño que facilitan su desarrollo, despliegue, mantenimiento e interoperación.

Para ello se basó en los trabajos desarrollados por el comité de la AICC y trató de mejorarlo, creando la noción de metadatos para los objetos de aprendizaje, o *Learning Object Metadata* (LOM) [IEEE, 2002], que define elementos para describir los recursos de

aprendizaje. IMS y ADL utilizan estos elementos y las estructuras de LOM en sus respectivas especificaciones. En la actualidad ISO está en proceso de asumir LOM como estándar.

IEEE tiene como misión principal la de "desarrollar estándares técnicos, prácticas recomendadas y guías para componentes software, herramientas, tecnologías y métodos de diseño que faciliten el desarrollo, implantación, mantenimiento e interoperabilidad de implementación en ordenadores de sistemas educativos" [IEEE, 2001].

El comité encargado de elaborar estándares y recomendaciones para entornos e-learning, el LTSC, tiene más de una docena de grupos de trabajo (*working groups* o WGs) y grupos de estudio (*study groups* o SGs) que desarrollan especificaciones para la industria e-learning. A continuación se presenta en la Tabla 2.5 los principales grupos de trabajo del LTCS con sus principales tareas realizadas.

GRUPO DE TRABAJO	TAREAS REALIZADAS DENTRO DEL IEEE LTCS
IEEE 1484.1 Architecture and Reference Model IEEE 1484.3 Glossary	Actividades generales
IEEE 1484.12 Learning Object Metadata IEEE 1484.14 Semantics and Exchange Bindings IEEE 1484.15 Data Interchange Protocols	Actividades relacionadas con los datos y metadatos
IEEE 1484.11 Computer Managed Instruction IEEE 1484.18 Platforms and Media Profiles IEEE 1484.20 Competency Definitions	Actividades relacionadas con los LMS y las aplicaciones

Tabla 2.5 Relación de los principales grupos de trabajo del LTCS y sus tareas realizadas

LTSC también trabaja de forma coordinada con otra iniciativa denominada ISO JTC1 SC36, que es un subcomité formado en forma conjunta por la ISO (*International Standard Organization*) y por la IEC (*International Electrotechnical Commission*), dedicado a la normalización en el ámbito de las Tecnologías de la Información para la formación, educación y aprendizaje.

El trabajo más importante de este comité, es el estándar que especifica Metadatos para Objetos Educativos, *Learning Object Metadata* (LOM). Este estándar especifica un esquema conceptual de datos que define la estructura de una instancia de metadatos para un objeto educativo. Para este estándar, un objeto educativo se define como cualquier entidad, digital o no, susceptible de ser usada en aprendizaje, educación o formación.

En lo que respecta a este estándar, una instancia de metadatos para un objeto educativo describe las características relevantes del objeto educativo al que se aplica. Dichas características se pueden agrupar en las categorías general, ciclo de vida, meta-metadatos, técnica, uso educativo, derechos, relación, anotación y clasificación.

El esquema conceptual de datos definido en este estándar permite la diversidad lingüística, tanto de los objetos educativos como de las instancias de metadatos que los describan.

Este esquema conceptual de datos especifica los elementos de datos de los que se compone una instancia de metadatos para un objeto educativo. Se asume que esta parte del estándar será referenciada por otros estándares que definirán descripciones de implementación del esquema de datos; de manera que una instancia de metadatos para un objeto educativo pueda ser usada por un sistema basado en tecnología educativa para gestionar, localizar, evaluar o intercambiar objetos educativos. No define como un sistema basado en tecnología educativa representará o usará una instancia de metadatos de un objeto educativo.

El propósito de este estándar es facilitar la búsqueda, evaluación, adquisición y uso de los objetos educativos, por ejemplo, por alumnos, profesores o procesos automáticos de software. Este estándar también facilita el intercambio y uso compartido de objetos educativos, permitiendo el desarrollo de catálogos e inventarios al tiempo que se toman en consideración la diversidad cultural y los contextos lingüísticos en los que los objetos educativos y sus metadatos serán reutilizados.

Especificando un esquema conceptual de datos común, se asegura que las implementaciones de los Metadatos de Objetos Educativos tendrán un alto grado de

interoperabilidad semántica. Como consecuencia, se simplificarán las transformaciones entre implementaciones. También especifica un esquema base que puede extenderse a medida que se avanza en su desarrollo práctico, por ejemplo, facilitando la planificación adaptativa y automática de los objetos educativos por agentes software.

IMS GLC

IMS (*Instructional Management Systems*) [IMS, 2009] es un proyecto iniciado en el año 1997 por la asociación internacional EDUCAUSE [2009]. Dicha organización pretende incidir en los cambios en la educación superior por medio de "la introducción, el uso y la administración de recursos de información y tecnologías en la enseñanza, en el aprendizaje, en la investigación y en la administración institucional" para lo cual han iniciado toda una serie de programas para el desarrollo de actividades profesionales, como, entre otros, la publicación de revistas, la investigación, la estructuración de iniciativas políticas y estratégicas, o el desarrollo de servicios de información en línea. Una idea de la magnitud de este tipo de esfuerzos se refleja en los siguientes datos: pertenecen a EDUCAUSE más de 2200 colegios, universidades y organizaciones educativas, así como más de 250 corporaciones que tienen que ver con los niveles superiores de la educación.

Actualmente IMS es una organización sin ánimo de lucro (conocida como *IMS Global Learning Consortium*) en la que participan más de 50 miembros y afiliados, provenientes del sector del e-learning. Incluyendo vendedores de hardware y software, instituciones educativas, organismos gubernamentales, proveedores de contenidos multimedia, etc. El consorcio facilita un foro neutral en el que los miembros que son competidores entre sí colaboran estableciendo y/o empleando estándares y especificaciones comunes, para satisfacer las necesidades del mundo real.

En sus inicios el estándar IMS surgió para su empleo en sistemas de educación superior, no obstante las especificaciones actuales no se limitan a éste área y abarcan un amplio rango de contextos de aprendizaje, desde las "escuelas K-12" (en inglés *K-12 school*, término empleado para referenciar a las guarderías hasta los 12 años) hasta los métodos de aprendizaje para grandes empresas y los organismos gubernamentales.

El ámbito de las especificaciones IMS, en ocasiones denominadas como "aprendizaje distribuido" (en inglés, "*distribuited learning*"), incluye entornos "*on-line*" y "*off-line*", produciendo lo que se denomina entornos síncronos (en tiempo real) o asíncronos. Esto supone que los contextos de aprendizaje beneficiarios de las especificaciones IMS incluyen entornos específicos para Internet (como los cursos basados en el Web) así como situaciones de aprendizaje que requieran recursos "*off-line*" (como los cursos basados en CD-ROM). Los estudiantes pueden encontrarse en un entorno tradicional (el aula de una escuela o la universidad), en un curso de aprendizaje de una empresa, o en casa. Por ejemplo, la especificación de recursos de aprendizaje IMS Meta-Data que será detallada en apartados posteriores, facilita al estudiante la búsqueda de información a través de una herramienta buscadora que actúa tanto sobre los recursos "Web" como sobre el CD-ROM o DVD.

Los documentos producidos por IMS establecen especificaciones que definen formatos y protocolos para el intercambio de información entre entornos IMS. Pero no pretende imponer a los desarrolladores la forma en que deben hacer sus aplicaciones educativas. El grupo técnico de desarrollo asumió como punto de partida del trabajo, unos requerimientos o especie de lista de deseos generada por los usuarios y desarrolladores de recursos de aprendizaje y a partir de éstos, se inicia la elaboración de las especificaciones.

El resultado del proceso de diseño de requerimientos, está plasmado en un documento público liberado por EDUCOM/NLII IMS [IMS, 2009]. El documento presenta cuatro secciones: resumen, diseño, requerimientos propiamente dichos y por último implementación. Se destacan a continuación los elementos más interesantes y relevantes del documento:

- Se establecen cuatro tipos de interesados o actores afectados por el proyecto IMS. Son los siguientes: Alumnos, Profesores, Proveedores y Coordinadores.
- Se definen cuatro dimensiones básicas de interacción de estas personas: Distancia, Tiempo, Afiliación y Modo de entrega. Respecto al tiempo no está limitado a una progresión lineal, incluye interacciones síncronas y asíncronas. El entorno puede situarse en una localización física o distribuirse a través de espacios físicos y electrónicos, lo que implica un modelo de partes interrelacionadas.
- La modulación. Es consecuencia de la concepción de partes interrelacionadas antes mencionada. La idea de la modulación de los materiales y procesos de enseñanza está también fuertemente apoyada por el predominio del diseño

orientado a objetos en el área del software, y supone un camino a ambientes de aprendizaje distribuidos.

- El proceso de determinación de requerimientos, parte de las características deseables, y a partir de ellas analiza los requisitos para satisfacerlas. De modo que en el documento se listan una serie de grupos de características y en cada caso se especifican los requerimientos derivados. Ello implica que no hay correspondencia uno a uno, entre características deseables y requerimientos, pues uno de ellos puede servir a más de uno de aquéllos. La Figura 2.7 ilustra los grupos de requerimientos básicos.

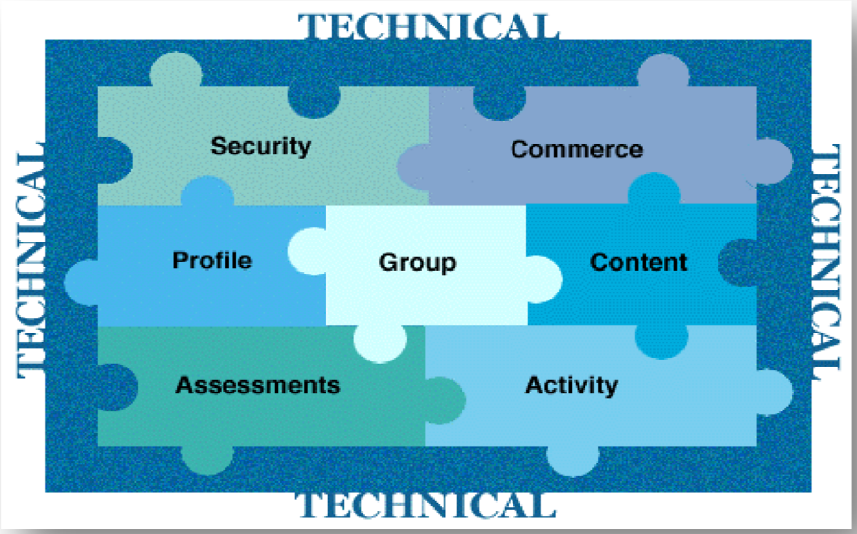


Figura 2.7 Grupos de requerimientos básicos

- Respecto a los requerimientos, el documento finaliza haciendo tres observaciones con relación a la interacción entre los grupos de funcionalidades establecidos:
    - En primer lugar, los requisitos técnicos actúan como un armazón o infraestructura que sostiene todos los requerimientos juntos.
    - El segundo punto importante del diagrama es la posición central ocupada por los requerimientos de gestión de grupo.
    - Finalmente, la descripción de las características como partes modulares reitera la escalabilidad e interoperabilidad de los recursos IMS.

Todas las especificaciones de IMS se detallan en tres documentos:

- **Guía de Implementación y Consejos.** Es el documento más narrativo de los tres. En él se incluyen: la forma de uso de la especificación, la relación con otras especificaciones, y cualquier tipo de información variada que pueda servir de ayuda. Suele ser el documento que se usa para iniciarse en la especificación.
- **Modelo de Información.** Documento que muestra la estructura de datos completa. Normalmente posee una tabla detallada de cada uno de los elementos de la especificación. En ella, se enumeran las propiedades de los elementos tales como el nombre, la multiplicidad, el tipo o si son obligatorios.
- **Documento de Enlace.** Documento que ofrece la forma de representar la estructura de datos de la especificación en XML. Muestra el árbol XML con cada uno de sus elementos y sus atributos.

Las especificaciones elaboradas por IMS son:

- **Accessibility**: Las especificaciones de accesibilidad tratan de garantizar que los productos y tecnologías que se desarrollen en el área de la educación sean útiles y accesibles para todo el mundo, en todo momento y en todos los lugares.
- **Competency Definitions**: Las especificaciones de definición de competencias tratan de establecer las formas de gestión de competencias relacionadas con el sistema e-learning, las cuales engloban, entre otras, la gestión de las competencias de los alumnos, la creación de definiciones de competencias y la asociación de dichas definiciones con los alumnos, y los prerrequisitos. Este modelo de información puede usarse para el intercambio de definiciones entre diferentes sistemas de aprendizaje, sistemas de recursos humanos, contenidos de aprendizaje, repositorios de competencias o habilidades y en cualquier otro sistema relevante. Estas especificaciones también proporcionan referencias únicas de descripciones para su inclusión en otros modelos de información.
- **Content Packaging:** Estas especificaciones establecen las pautas y descripciones para el empaquetamiento de los materiales de aprendizaje (como pueden ser un curso individual o una colección de cursos), en un paquete que se pueda distribuir. El "*Content Packaging*" sirve de enlace con la descripción, estructura, localización de los materiales y la definición de contenidos especiales "*on-line*".

- **Common Cartridge:** Es uno de los tres grandes estándares que componen la nueva generación de lo que se conoce como *Digital Learning Services Standards*. Estos son:
    - Organizar y distribuir contenido educativo (Common Cartridge – CC)
    - Aplicaciones, sistemas y mash-ups o aplicaciones Web híbridas (Learning Tools Interoperability – LTI)
    - Información referente al aprendizaje: privilegios y resultados (Learning Information Services – LIS)

  El nuevo estándar (Julio 2008) está basado en especificaciones ya existentes:
    - IEEE LOM (metadatos)
    - IMS Content Packaging v1.2 (empaquetado de contenidos)
    - IMS Question & Test Interoperability v1.2.1 (cuestionarios de evaluación)
    - IMS Authorization Web Service v1.0 (autorización de acceso)

  Con este estándar, IMS ha optado por simplificar lo máximo posible y dejar de lado la mayoría de características opcionales y extensiones. Así, para los metadatos sólo se usan los quince elementos de Dublin Core [2008] (mapeados a los elementos correspondientes de LOM) y en cuanto a QTI (*IMS Question & Test Interoperability*) sólo se contemplan los seis tipos de preguntas más comunes. Se han añadido, en cambio, nuevas características cuando se ha estimado necesario:
    - Un nuevo tipo de recurso que sirve para iniciar un fórum de debate.
    - Un protocolo de autorización (*IMS Authorization Web Service*) que permite al editor de un paquete controlar el acceso a sus contenidos.
- **Digital Repositories**: El propósito de las especificaciones de los repositorios digitales es describir los contenidos y desarrollos de los mismos, y proporcionar modelos de información y protocolos para habilitar la interoperabilidad entre diferentes repositorios, tanto para las operaciones de búsqueda como en las de publicación y almacenamiento a través de la red.
- **Enterprise**: Estas especificaciones son un modelo de información que da soporte a la interoperabilidad entre personas, grupos y afiliaciones entre dos o más sistemas de aprendizaje.
- **Enterprise Services**: Se trata de la definición de cómo los sistemas de aprendizaje son capaces de gestionar el intercambio de información concerniente a las personas, grupos y componentes del contexto de aprendizaje.
- **ePortfolio**: Las especificaciones de ePortfolio están diseñadas para que la documentación sea intercambiable entre diferentes sistemas e instituciones. Esta especificación:

- Soporta el avance de los aprendizajes de larga duración, que son importantes para ciertas iniciativas.
- Facilita el intercambio de los documentos.
- Permite a educadores e instituciones mejorar las capacidades de determinadas competencias.
- Realza el aprendizaje e incrementa el desarrollo del empleado.

- **General Web Services**: La base de los Servicios Web Generales promueve la interoperabilidad de las especificaciones de los servicios Web basados en software y plataformas de diferentes vendedores. Se centra en la especificación de un conjunto de servicios Web y en los problemas más comunes encontrados en su implementación. Trata de dirigir la interoperabilidad en la capa de aplicación, en particular, la descripción de los comportamientos expuestos vía servicios Web.
- **Learner Information**: Se trata de la información referente al estudiante (entendiendo como tal tanto una persona como a un grupo de ellas) o a los productores del contenido de aprendizaje (creadores, proveedores o vendedores) La especificación del paquete de información de aprendizaje IMS (LIP) establece el modelo de interoperabilidad de los sistemas de aprendizaje basados en Internet con aquellos otros sistemas que soportan dicho entorno. El propósito de la especificación es definir un conjunto de paquetes que puedan ser usados con otros servidores de información de aprendizaje. Un servidor de información de aprendizaje intercambia datos con los distintos sistemas de aprendizaje y proporciona una descripción de dichos datos.
- **Learning Design**: Estas especificaciones proporcionan un amplio espectro de teorías pedagógicas para el estudio "*on-line*". En lugar de tratar de englobar las cuestiones específicas de muchas pedagogías, proporciona un lenguaje flexible y genérico para su descripción. Este enfoque tiene la ventaja de que tan solo es necesario aprender a manejar un conjunto de herramientas de diseño sobre las cuales aplicar las distintas pedagogías. Esta especificación amplía el concepto de "*Learning Object*" en el de "*Unit of Learning*".
- **Meta-data:** Es el modelo de información IMS empleado para describir los contenidos que los Metadatos han de contener.
- **Question and Test Interoperability (QTI)**: Es un modelo de información que describe los datos correspondientes a las preguntas y cuestionarios junto con sus soluciones.
- **Resource List Interoperability (RLI)**: Es la especificación que detalla como los metadatos estructurados pueden ser intercambiados entre los diferentes sistemas que almacenan recursos. Las especificaciones se basan en un

servicio abstracto y en un modelo de datos que describe de forma general los recursos, una colección de esos recursos, y los comportamientos asociados con el servicio de gestión de recursos. El modelo de datos se expresa en el lenguaje XML, combinando elementos del estándar IEEE-LOM e ISO 690-2.

- **Shareable State Persistence**: Esta especificación describe una extensión para los sistemas en tiempo de ejecución (como SCORM) que facilita el almacenamiento del estado de la información compartida. Actualmente no hay un método de almacenamiento de la información de estado en los sistemas de tiempo real que permita que la información sea recuperada por el mismo objeto ni mediante otro que no sea propietario. Esta capacidad es crucial para la persistencia de una información, a veces en un estado complejo, que es generada por una variedad de contenido interactivo (ej.: simulaciones) y que actualmente se almacena y recupera a través de métodos propietarios.
- **Simple Sequencing**: Define un método para representar el comportamiento y funcionalidades que los sistemas deben implementar. Incluye reglas que describen las ramificaciones o el flujo de instrucciones según los resultados de la interacción entre el estudiante y los contenidos.
- **Tools Interoperability**: Este estándar posibilita la creciente demanda de mecanismos reutilizables para integrar herramientas de terceros en plataformas LMS. Estas herramientas pueden añadir funcionalidades a los LMS como evaluaciones o artículos de aprendizaje específicos de la disciplina.
- **Vocabularies Definition Exchange (VDEX)**: Define una gramática especial para el intercambio de vocabulario, más concretamente para el intercambio de una lista simple de valores legibles por la máquina, o términos, junto con información que ayuda para que el ser humano pueda entender el significado o aplicación de los diferentes términos. VDEX se emplea para expresar los datos válidos en instancias del IEEE-LOM, IMS Meta-data, IMS LIP, ADL, SCORM, etc.
- **Abstract Framework**: Es un mecanismo para permitir a IMS describir el contexto dentro del cual se desarrollan sus especificaciones de todas sus tecnologías e-learning. Este framework no se ocupa de definir la arquitectura IMS, sino de definir un conjunto de servicios para los cuales se definen sus especificaciones de interoperabilidad. En los casos donde IMS no puede producir una especificación, trata de adoptar o recomendar una especificación conveniente de otra organización.

AICC

La industria de la aviación (*AICC - Aviation Industry CBT Committee*) [AICC, 2009] ha sido tradicionalmente un gran consumidor de formación, por lo que en 1992 decidieron crear un comité que desarrollase una normativa para sus proveedores de formación e-learning. De este modo garantizaban la armonización de los requerimientos de los cursos, así como la homogeneización de los resultados obtenidos de los mismos.

Fue el primer organismo creado para desarrollar un conjunto de normas que permitiese el intercambio de cursos CBT (*Computer Based-Training*) entre diferentes sistemas. Sus dos principales aportaciones son sus propuestas para entornos de ejecución y para estructuración de cursos, que constituyen la base de SCORM.

El comité de aprendizaje e-learning del AICC contribuyó a la estandarización de las estructuras de cursos con su documento "*Guidelines for Interoperability*". En la nomenclatura del AICC, las partes de un curso que se pueden reubicar para definir el orden en el que serán mostradas a los estudiantes se denominan elementos de estructura. Hay dos tipos de elementos de estructura: las unidades asignables, que es el elemento educacional más pequeño que se le puede mostrar al alumno (una página HTML, un simulador, un test, etc.) y bloques, que son agrupaciones de unidades asignables y otros bloques. Existe además otro elemento de construcción denominado objetivo, que se puede utilizar para definir requisitos en un curso. Las unidades asignables, bloques y objetivos se denominan elementos del curso. La especificación es neutral en cuanto al número de niveles que se pueden establecer en la jerarquía de un curso. Es posible anidar bloques del curso hasta el nivel que se desee. Sin embargo, AICC ha establecido una jerarquía de referencia de 10 niveles. El modelo de referencia define reglas de nombrado útiles para evitar inconsistencias.

AICC define los entornos de ejecución cómo *Computer Management Instruction* (CMI), mientras que se utilizan los términos *Computer Based Training* (CBT), lecciones o unidades asignables para identificar los contenidos entregados a los alumnos. El entorno de ejecución de AICC ha evolucionado: primero estaba orientado a equipos autónomos utilizando comunicación mediante ficheros, y posteriormente ha desarrollado una interfaz basada en HTTP que permite comunicación entre redes de computadores. El primer avance que produjo AICC, y en el que resultó ser pionero, fue la separación del CBT y del CMI. Las razones en las que se basó para llegar a esta separación son:

- Los instructores están acostumbrados a la utilización de un CMI, la adopción de un nuevo CMI para la entrega de contenidos es un proceso costoso.
- El mantenimiento de varios CMI´s es muy costoso.
- El CMI determina, en gran medida, la apariencia de la interfaz de usuario que se ofrece a los alumnos. Es importante que esta apariencia sea la misma con independencia del contenido que esté gestionando.
- El CMI que mejor se adapta a las necesidades de una organización puede no proporcionar los contenidos del curso adecuados y viceversa. En caso de que se necesiten nuevos contenidos, sería preferible integrarlos en la plataforma existente.

En esta situación, AICC consideró que deberían proporcionarse estándares para la transferencia de un curso de un sistema dado a uno nuevo, incluyendo la estructura del curso, los criterios de navegación y los contenidos. Además también deberían ser estandarizadas tanto la comunicación entre el CMI y las lecciones, como la forma en la que debe ser gestionada la información de los estudiantes.

Las especificaciones del AICC cubren nueve áreas principales, que van desde los *Learning Objects* (LO) hasta los *Learning Management Systems* (LMS). Normalmente, cuando una compañía afirma que cumple con las especificaciones AICC, significa que cumple con al menos una de estas "*guidelines*" y recomendaciones (*AICC Guidelines and Recommendations, AGRs*). La lista completa de AGRs es la siguiente:

- **AGR 001 AICC Publications**: Este documento describe todas las publicaciones de AICC. Identifica y proporciona un extracto de las pautas actuales y de las recomendaciones de AICC y de los documentos técnicos del documento.
- **AGR 002 Courseware Delivery Stations**: Este documento contiene las recomendaciones de la industria de la aviación para la adquisición de una estación de entrenamiento computerizado. Las recomendaciones proporcionan los requisitos para la CPU de una estación determinada de entrenamiento: la velocidad de reloj, el bus, la fuente de alimentación, el sistema operativo, la memoria RAM, la memoria ROM, el adaptador gráfico, el monitor, el ratón, el teclado, el sistema de audio digital y de red.
- **AGR 003 Digital Audio**: Este documento recomienda las pautas que promueven la interoperabilidad del audio digital. La interoperabilidad en este ámbito proporciona que el audio del curso pueda ser leído en diversos PC's

con diferentes tarjetas de sonido de la estación de entrenamiento. También implica compatibilidad con diferentes formatos de audio.

- **AGR 004 Operating/Windowing System**: Este documento proporciona una recomendación formal a la industria de la aviación para el sistema operativo usado y para las ventanas e iconos del curso.
- **AGR 005 CBT Peripheral Devices**: Este documento proporciona las pautas a seguir para conseguir la interoperabilidad de los dispositivos o periféricos, tales como la entrada de video XY (una pantalla, un ratón, o un Trackball), y los drivers.
- **AGR 006 Computer-Managed Instruction**: Este documento contiene las recomendaciones para lograr la interoperabilidad CMI en sistemas de ficheros locales. La interoperabilidad en este aspecto se refiere a la capacidad de un sistema CMI para manejar lecciones del CBT de distintos orígenes, así como los datos de intercambio entre sistemas CMI.
- **AGR 007 Courseware Interchange**: Este documento recomienda las pautas para el intercambio de los elementos del curso del CBT. Estos elementos incluyen: Texto, gráficos, movimientos, audio, y lógica. Estas pautas abarcan: 1) los componentes principales de los datos del curso del CBT, y 2) los formatos de datos estándares para esos componentes.
- **AGR 008 Digital Videos**: Este documento recomienda las pautas para la creación, la distribución, y el uso del vídeo digital del curso del CBT.
- **AGR 009 Icon Standards**: Este documento contiene las recomendaciones para la interfaz de usuario y del diseño gráfico del curso del CBT.
- **AGR 010 Web-Based Computer-Managed Instruction**: Contiene las recomendaciones sobre la interoperabilidad de las plataformas de formación y los cursos.

Aunque AICC ha publicado varias guías, la más seguida es la AGR 010 que trata de la interoperabilidad de las plataformas de formación y los cursos. En esta guía se resuelven dos de los problemas fundamentales:

- La carga sin problemas de cursos creados por terceros en un LMS. Este objetivo se consigue definiendo el curso como una entidad totalmente independiente de la plataforma, y creando un sistema (ficheros) de descripción del curso que pueda ser entendido por cualquier plataforma.
- La comunicación entre el LMS y el curso, de tal modo que el curso pueda obtener información necesaria sobre el usuario, y después transmitir los resultados de las interacciones y evaluaciones realizadas por el mismo a la plataforma, con objeto de su almacenamiento y tratamiento estadístico.

Este segundo objetivo es logrado mediante la definición de un mecanismo de comunicación entre el curso y la plataforma, y un conjunto de datos mínimos que deben ser transmitidos del curso a la plataforma y viceversa. AICC describe dos mecanismos, uno más sencillo y extendido basado en el protocolo http, y otro mediante una API.

AICC cuenta con un programa de certificación (a diferencia de las otras iniciativas) y dispone de un "*test suite"* que le permite a las compañías verificar que sus productos son compatibles con otros sistemas que cumplen con las especificaciones AICC. Actualmente la AGR 010 de AICC es el "estándar de facto" en la industria del e-learning.

ADL

ADL (*Advanced Distributed Learning*) [ADL, 2009], es un programa del Departamento de Defensa de los Estados Unidos y de la Oficina de Ciencia y Tecnología de la Casa Blanca para desarrollar principios y guías de trabajo necesarias para el desarrollo y la implementación eficiente, efectiva y en gran escala, de formación educativa sobre nuevas tecnologías Web.

Este organismo recogió especificaciones de IMS, IEEE LTSC y AICC y las refundió y mejoró en su propio estándar: SCORM (*Sharable Content Object Reference Model* - Modelo de Referencia para Objetos de Contenidos Intercambiables), cuya última versión es la 1.3, más conocida como SCORM 2004 [ADL, 2004].

De lo recogido por ADL de las distintas organizaciones se destacan los siguientes elementos:

- El sistema de descripción de cursos en XML de IMS.
- El mecanismo de intercambio de información mediante una API (Application Programming Interface) de AICC.
- Los metadatos de objetos provistos por el estándar LOM (Learning Objects Metadata) de IEEE.

SCORM proporciona un marco de trabajo y una referencia de implementación detallada que permite a los contenidos y a los sistemas usar SCORM para comunicarse

con otros sistemas, logrando así interoperabilidad, reusabilidad y adaptabilidad [ADL, 2004].

SCORM permite cubrir con suficientes garantías los aspectos siguientes: descripción de los contenidos; empaquetamiento y organización de los contenidos; presentación y secuenciación de los contenidos; y, por último, seguimiento del proceso de aprendizaje.

Sin embargo, la especificación de SCORM no cubre todos los aspectos del e-learning, por ejemplo, no especifica cómo es almacenada la información y qué informes son generados, qué modelos pedagógicos y de aprendizaje deben ser usados, o cómo la información del estudiante es recopilada.

Las especificaciones de SCORM están organizadas como "libros" separados (Figura 2.8):

- **Libro 1: SCORM Overview**: Contiene una descripción general de la iniciativa de ADL, un análisis de SCORM, y un resumen de las especificaciones técnicas contenidas en las siguientes secciones.
- **Libro 2: SCORM Content Aggregation Model (CAM)**: Como características principales se destacan:
    - Contiene una guía para identificar y agregar recursos dentro de un contenido de aprendizaje estructurado.
    - Describe una nomenclatura para el contenido de aprendizaje.
    - Define responsabilidades y requerimientos para construir agregaciones de contenidos, como pueden ser cursos, lecciones, módulos, etc.
    - Contiene información sobre la creación de contenidos aplicando metadatos y el secuenciamiento y navegación en el contexto del empaquetamiento de los contenidos (*SCORM Content Packaging*).
    - Está fuertemente relacionado con el libro 3 (*SCORM Run-Time Environment*).

  Los metadatos SCORM describen los diferentes componentes del SCORM Content Aggregation Model (Agregaciones de contenidos, Actividades, SCO's y Assets). Los metadatos son una forma de etiquetar esos componentes para facilitar su búsqueda.

Un paquete de contenido, en un sentido general, une objetos de contenido con la organización de los mismos descrita en un manifiesto. El manifiesto (imsmanifiest.xml) es la parte esencial de SCORM Content Packages y está definido en XML. Describe los contenidos del paquete y debe incluir una descripción de la estructura del contenido, aunque ésta es opcional. Está basado en la especificación IMS Learning Resource Meta-data Information Model, la cual está basada, a su vez, en el IEEE LTSC Learning Object Metadata (LOM) Specification, que fue el resultado de un esfuerzo en conjunto entre el IMS Global Learning Consortium y ARIADNE.

- **Libro 3: SCORM Run-Time Environment (RTE):** Incluye una guía para lanzar contenidos y hacerles un seguimiento en un ambiente basado en el Web de un entorno de ejecución que incluye: un protocolo específico para la ejecución de contenidos Web, un API entre el contenido y el LMS y un modelo de datos que define el flujo de datos intercambiado entre el entorno LMS y el contenido que se ejecuta en el entorno de ejecución.

  El *Launch* (motor de presentación) define las relaciones entre los SCO's y el LMS. La API JavaScript de SCORM es una pieza de código que en un ambiente virtual de educación (*Virtual Learning Environment*, VLE) provee el navegador de un estudiante cuando éste requiere contenido e-learning de un objeto SCORM. El código captura acciones específicas del estudiante, por medio de instrucciones, en el contenido SCORM y las pasa por el VLE. Las acciones incluyen eventos tales como: cuán lejos ha llegado un estudiante a través de una pieza de contenido, puntuaciones en test y cuánto tiempo le llevó trabajar con el material.

  Este libro es derivado del CMI001 *Guidelines for Interoperability* de la AICC. (Basado en los estándares de IEEE API 1484.11.2 y IEEE Data Model 1484.11.1)

- **Libro 4: Scorm Sequencing and Navigaton (SN):** Describe cómo el contenido debe ser secuenciado a través del sistema junto con los eventos de navegación. El contenido puede ser descrito por un conjunto de actividades predefinidas, definidas durante el diseño de los contenidos. Describe además cómo un LMS de SCORM interpreta las reglas de secuenciamiento expresadas en un entorno de desarrollo y sus efectos en el *Run-Time Environment*.

  Describe las ramificaciones y el camino que siguen las actividades de aprendizaje, representadas mediante un Árbol de Actividad, basadas en los resultados de las interacciones del estudiante con los objetos de contenido y una estrategia de secuenciamiento. Un Árbol de Actividad es una estructura conceptual de actividades manejadas por el LMS propias de cada alumno. En SCORM, una actividad de aprendizaje hace referencia a objetos de

contenidos que cursa el estudiante. Está basado en la especificación propuesta para el secuenciamiento de contenidos de IMS.

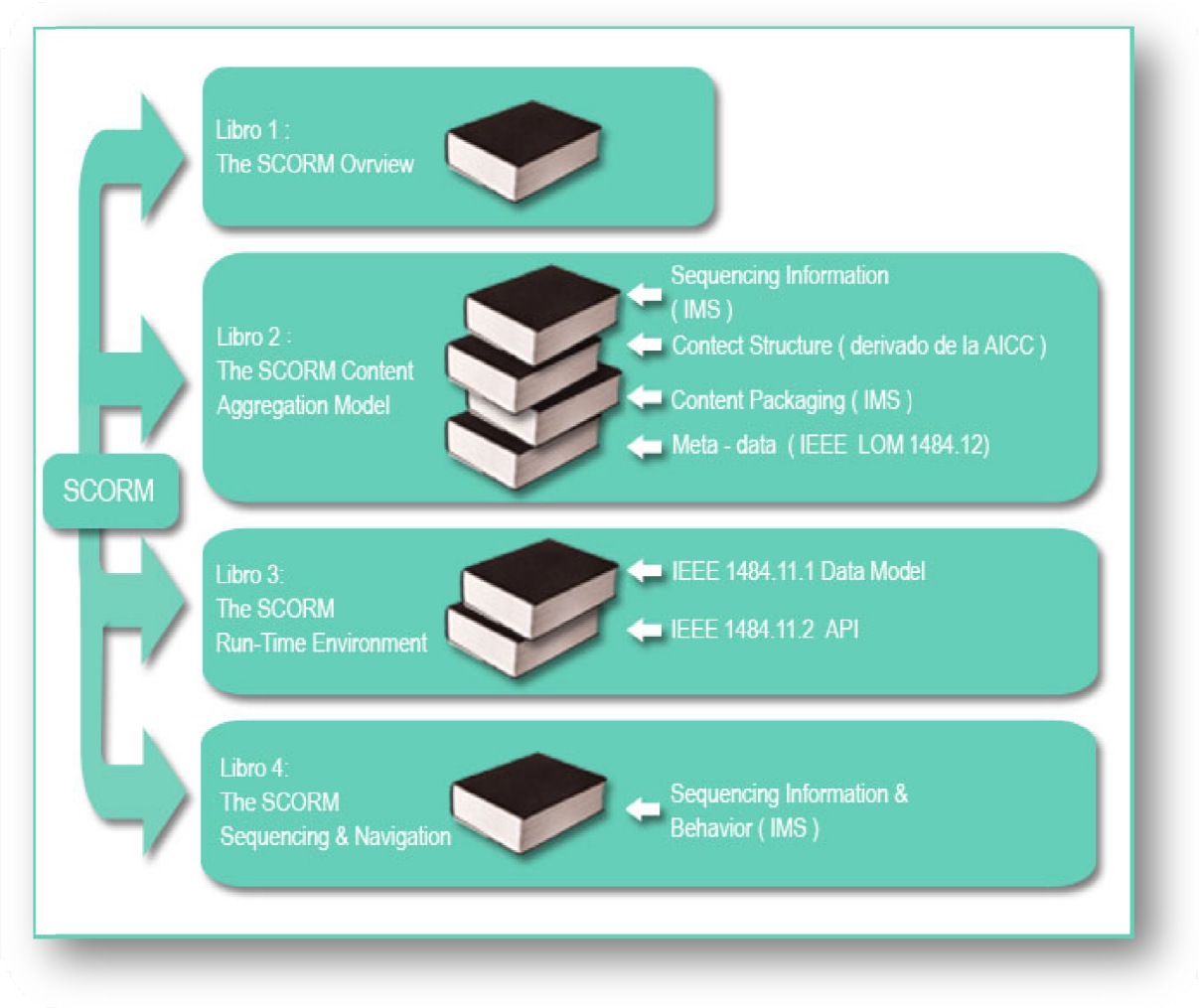


Figura 2.8 Organización de las especificaciones SCORM [ADL, 2009]

SCORM reúne los esfuerzos de diferentes organizaciones para llegar a un modelo general que puede ser tomado como referencia para la creación de sistemas de aprendizaje. En la Figura 2.8 se puede apreciar esta labor de integración y recolección, pues en ella se muestra qué organización ha tomado cada una de las aportaciones que propone ADL para SCORM.

ARIADNE

ARIADNE (*Alliance of Remote Instructional Authoring & Distribution Networks for Europe*) tiene su origen en un proyecto tecnológico del sector de "Telemática para la Educación y Enseñanza" para la investigación y desarrollo llevado a cabo por la Unión Europea y comenzado en el año 1996. El proyecto se centra en el desarrollo de herramientas y metodologías para la generación, gestión y reutilización de los componentes pedagógicos basados en computadores y en la educación con soporte telemático. La validación de los conceptos y herramientas se lleva a cabo en diversos entornos (tanto académicos como corporativos) en toda Europa. El núcleo central del proyecto es la distribución de una librería con componentes digitales reutilizables para la educación, llamado KPS (*Knowledge Pool System*). Se trata de una red europea de recursos educativos distribuidos, alrededor de la cual se han creado una serie de herramientas [ARIADNE, 2009].

La contribución más significativa del proyecto ARIADNE en lo concerniente a los estándares educativos, ha sido el conjunto de especificaciones educativas de metadatos ARIADNE (el "encabezamiento pedagógico"), que es uno de los componentes principales del estándar IEEE/LTSC LOM.

El proyecto ARIADNE establece las siguientes categorías de usuarios [ARIADNE, 2009]:

- Autores que crean nuevo material pedagógico: bien empleando las herramientas de software pedagógico facilitadas por ARIADNE, o bien reutilizando material existente en el repositorio KPS.
- Educadores que almacenan e indexan su material pedagógico en el repositorio KPS.
- Motores pedagógicos, que crean o modifican cursos, empleando el editor de currículos.
- Gestores de cursos, que administran los cursos empleando las funciones del sistema ARIADNE.
- Administradores del KPS, que emplean el conjunto de funcionalidades técnico - administrativas del KPS.
- Estudiantes de las categorías expuestas, que se ciñen al sistema de currículos KPS gracias a la interfaz de aprendizaje proporcionada por ARIADNE. (individualizada y siempre actualizada -"*up-to-date*"-).

Actualmente, el mantenimiento, uso, y desarrollo adicional en la mejora de las herramientas ARIADNE son competencias de la fundación ARIADNE, la cual surge al concretarse el plan de explotación de ARIADNE y formalizarse el soporte que éste recibe de la Unión Europea y Suiza.

Las principales características que presenta ARIADNE son las siguientes:

- **Usuarios implicados:**
  - Autores de documentos pedagógicos: como universidades, directores de educación, estudiantes, etc.
  - Productores y administradores de cursos de aprendizaje: como educadores, directores de educación, etc.
  - Usuarios finales: estudiantes.
- **Tecnologías y/o enfoques usados:**
  - Compartir y reutilizar componentes pedagógicos a través de la indexación o almacenamiento de los mismos en KPS.
  - Emplear canales telemáticos adecuados, ajustándose a las diferentes situaciones.
  - Facilitar el uso de los componentes pedagógicos, currículos estructurados y visualización individual de los cursos.
- **Ventajas para la ciudadanía:**
  - Mejores esquemas de formación continua.
  - Fácil acceso a la formación para las categorías sociales menos favorecidas.
- **Beneficios para los usuarios de las aplicaciones:**
  - Estudiantes: escenarios de aprendizaje atractivos y mucho más efectivos.
  - Autores del material pedagógico: mejor productividad y nuevas filosofías de colaboración.
  - Investigadores: mejores comunicaciones y esquemas de colaboración.
- **Beneficios previstos para las industrias europeas:**
  - Niveles de enseñanza continua mejores, más rápidos y mucho más económicos.
- **Contribución a las políticas de la Unión Europea:**
  - Posible factor de homogenización de las políticas de educación y enseñanza en toda Europa. Promoción de la colaboración entre los educadores europeos y simplificación de la comunicación entre los estudiantes europeos.

ISO

La organización internacional de estandarización (ISO, *International Organization for Standardization*) tiene un comité (JTC1) especializado en tecnologías de la información, al que a su vez, pertenece un subcomité (SC36) que se encarga de elaborar estándares relacionados con las tecnologías de la información para educación y formación [ISO-SC36, 2009]. Actualmente, este subcomité, cuenta con la colaboración de 28 países, de los cuales 22 son miembros "P", es decir, miembros participativos involucrados en actividades técnicas y con derecho a voto, entre ellos España con AENOR; y 6 son miembros "O", sólo observadores, que pudiendo estar involucrados en actividades técnicas, no votan. El subcomité SC36 está compuesto por siete grupos de trabajo activos más un grupo de compensación. A continuación se presenta un breve resumen del objetivo y tareas en la que están involucrados cada uno de estos grupos de trabajo:

- **SC36/WG1 – Vocabulary:** Este grupo se encarga de definir los principales términos asociados a las aplicaciones educativas, presentando definiciones y términos, los cuales denotan conceptos relevantes en este campo, constituyendo un punto crucial para el desarrollo de estándares.

  Su origen se debe, en principal medida, al reciente crecimiento de los productos relacionados con el estudio, la educación y el aprendizaje basados en la tecnología, ya que conduce al empleo de nombres diferentes para el mismo (o similar) concepto. La necesidad de interoperabilidad y reutilización de los productos y componentes utilizados en ámbitos educativos, requiere cierta unificación, para que de esta manera sea posible identificar y referirse a estos conceptos y productos mediante la utilización de una terminología común.

  Los principales interesados que podrían beneficiarse del estándar terminológico propuesto por este grupo son: los productores de estándares en el campo de la tecnología asociada al aprendizaje, los productores de herramientas, productos y servicios educativos, así como agencias que ofrecen servicios en este campo, compradores y usuarios de los productos (educadores, tutores y alumnos).

  La terminología estandartizada facilitará el entendimiento mutuo y la comunicación, mientras que el empleo de estándares en la descripción de productos promoverá su utilización mundial.

  Entre las principales ventajas que se pueden destacar están las relacionadas con la uniformidad en la descripción de los productos, sus características, componentes y requerimientos, facilitando el empleo

internacional del producto y la traducción de sus descripciones en otros idiomas.

- **SC36/WG2 - Collaborative Technology:** Este grupo encuentra su justificación en la evolución gradual que han sufrido los sistemas de aprendizaje, los cuales han pasado de ser sistemas independientes a conformar ambientes educativos distribuidos.

  Su propósito es especificar la estructura y los componentes de colaboración del lugar de trabajo, (tales como áreas de discusión, áreas de actividades de colaboración, espacios de trabajo para el aprendizaje electrónico, pizarras compartidas, objetos de usuario, etc.) así como sus cualidades y los enlaces que se producen entre ellas.

  Este proyecto considera agentes en un ambiente de aprendizaje colaborativo. Los agentes pueden representar a alumnos, o pueden ser agentes independientes como por ejemplo, los que manejan las herramientas compartidas, los que supervisan la comunicación entre los alumnos o entre otros agentes, etc.). El proyecto debe tratar las características específicas de comunicación de los agentes que están colaborando en un ambiente de aprendizaje, por lo que será necesario que se especifique el protocolo de comunicación agente-agente, así como el protocolo de sincronización existente entre los distintos sistemas de aprendizaje.
- **SC36/WG3 - Participant Information:** El alcance de este grupo reside en la información asociada a los participantes en lo que a los sistemas de aprendizaje se refiere. La información puede incluir elementos relacionados con la cultura, la lengua, los dispositivos, las preferencias cognitivas, las capacidades o las interfaces persona-ordenador.

  Su propósito es proporcionar un modelo de datos y los enlaces a las categorías específicas de información. La información personalizada puede ser utilizada, por ejemplo, para adaptar los sistemas educacionales a las necesidades de cada usuario.
- **SC36/WG4 - Management and Delivery of Learning, Education, and Training (MDLET):** Este grupo trabaja en el estándar ISO/IEC 2382, que representa solamente una parte de todo el conjunto de elementos que componen el modelo de datos necesario para importar contenido dentro de un LMS, permitiendo además establecer las comunicaciones entre el contenido y el LMS.

  Esta parte es la encargada de definir el *framework* de los modelos de datos y enlaces que permitirán importar dicho contenido en el LMS, así como establecer su comunicación. Estos modelos de datos y los enlaces serán

definidos en las diferentes partes del estándar, difiriendo del trabajo realizado en el proyecto IEEE P1484.11 en que adopta un mayor rango de pedagogías educativas y añade soporte para el contenido empaquetado, exigencia identificada de la práctica comercial.

- **SC36/WG5 - Quality Assurance and Descriptive Frameworks:** Este grupo trata de describir y caracterizar procesos, componentes y atributos relacionados con la calidad y la arquitectura de entornos tecnológicos en el campo del aprendizaje.

  El propósito de este grupo es proveer de un *framework* común con definiciones que incluyan conceptos, especificaciones, términos y definiciones a describir, especificando y entendiendo las características y métricas de calidad.
- **SC36/WG6 - International Standardized Profiles (ISP):** Un ISP es un documento internacionalmente reconocido y armonizado que identifica un estándar o un grupo de ellos, junto con las opciones y parámetros necesarios para acoplar una función o conjunto de funciones.

  Por otro lado, los perfiles definen una combinación de estándares que en conjunto funcionan como una función "bien-definida". Los perfiles estandarizan el uso de opciones y otras variaciones en los estándares, y proporcionan una base para el desarrollo de sistemas de pruebas internacionalmente reconocidas. Los ISPs no se generan simplemente para "legitimar" una opción particular de estándares y opciones, sino para promover la interoperabilidad real en sistemas.
- **SC36/WG7 - Culture/Language/Human-Functioning Activities:** Este grupo está íntimamente relacionado con el grupo de trabajo que garantiza la accesibilidad para DC (*Dublin Core*) [DublinCore, 2009]. Todo el trabajo realizado hasta el momento por el grupo de trabajo de DC, ha sido realizado en colaboración con otros grupos que también trabajan en el campo de la accesibilidad de metadatos. El trabajo comenzó con dos perfiles de metadatos desarrollados por IMS: uno encargado de describir las necesidades y las preferencias de usuarios individuales y otro para describir los recursos que ellos podrían querer usar. Este trabajo fue adoptado por este grupo y está actualmente en su etapa final antes de convertirse estándar ISO.

Los grupos de trabajo presentados anteriormente trabajan en la definición de los estándares que se resumen en la Tabla 2.6.

Estándar	Descripción
ISO/IEC 2382	Information technology -- Vocabulary -- Part 36: Learning, education, and training
ISO/IEC 19780	Information Technology -- Learning, Education and Training -- Collaborative Technology -- Learner to Learner Interaction Scheme.
ISO/IEC 19788	Information Technology -- Metadata for Learning Resources
ISO/IEC 23988	A code of practice for the use of information technology (IT) in the delivery of assessments
ISO/IEC 24725	Information technology -- Learning, education and training -- Profiles of standards and specifications
ISO/IEC 24751	Individualized Adaptability and Accessibility in E-learning, Education and Training
ISO/IEC 19796	Information technology -- Learning, education and training -- Quality management, assurance and metrics

Tabla 2.6 Estándares en los que trabaja ISO/IEC JTC1/SC36

2.3. PRINCIPALES ESTÁNDARES Y FRAMEWORKS PARA ENTORNOS E-LEARNING

Durante este apartado se comentarán los principales estándares y frameworks que han surgido para ofrecer interoperabilidad, eficiencia, transparencia e integración entre otros beneficios a los entornos de enseñanza y aprendizaje electrónico. No solamente existen estos estándares, sino que a lo largo de este capítulo ya se han comentado muchos otros, sin embargo estos son los de mayor importancia para el posterior desarrollo de la arquitectura objetivo de estudio en esta tesis.

En este apartado se comentarán los siguientes estándares y frameworks:

- **IMS Abstract Framework:** framework que aporta una representación abstracta del conjunto de servicios que se deberían utilizar para construir un sistema e-learning en su sentido más amplio.
- **IMS Digital Repositories Interoperability:** especificación cuya misión principal es la de facilitar el acceso a los contenidos en los repositorios.
- **CEN CWA Simple Query Interface:** especificación encargada de definir una interfaz de consulta sencilla e independiente.
- **CEN CWA Simple Publishing Interface:** especificación encargada de definir una interfaz de publicación de recursos junto con sus metadatos.
- **IMS Digital Learning Services:** conjunto de estándares encargado de garantizar la interoperabilidad, ofreciendo cambios en sistemas empresariales, servicios Web y software basado en servicios entre otros. Este grupo de estándares lo componen:
    - **IMS Common Cartridge:** especificación creada para definir un formato de empaquetado de objetos de aprendizaje común.
    - **IMS Learning Tools Interoperability:** herramientas creadas para mejorar la interacción entre los sistemas de gestión del aprendizaje y las herramientas del aprendizaje.
    - **IMS Learning Information Services:** estándar encargado de apoyar las interacciones y el intercambio de datos entre los sistemas de aprendizaje y los administradores, estudiantes o los sistemas de recursos.
- **IMS Resource List Interoperability:** especificación encargada de la definición de los listados de recursos con los que cuentan los cursos.
- **IMS Vocabulary Definition Exchange:** esta norma define un formato basado en XML para el intercambio de listas de valores de diferentes tipos, que son

usadas como fuente de los vocabularios que se usan para etiquetar metadatos.

- **IMS General Web Services:** especificación encargada de promover la interoperabilidad de las especificaciones de los servicios Web basados en software y plataformas de diferentes vendedores.

2.3.1. IMS Abstract Framework

Es interesante explicar de forma resumida el IMS Abstract Framework [IMS, 2003b], ya que es el contexto en el que se basa IMS para realizar las especificaciones de todas sus tecnologías de e-learning. Este framework no se ocupa de definir la arquitectura IMS, más bien es un mecanismo para definir un conjunto de servicios para los cuales se definen sus especificaciones de interoperabilidad. Como ya se indicó en un apartado anterior, en los casos donde IMS no puede producir una especificación, trata de adoptar o recomendar una especificación conveniente de otra organización.

El framework de IMS tiene como particularidades:

- Es una representación abstracta del conjunto de servicios que se utilizan para construir un sistema e-learning en su sentido más amplio.
- Está centrado en el soporte de los sistemas de formación distribuidos.
- Es un *framework* que cubre todo el rango de posibles arquitecturas e-learning que se podrían construir a partir de un conjunto de servicios definidos.

Y como principios:

- **Interoperabilidad:** Las especificaciones están basadas en el intercambio de información entre sistemas pero sin tomar decisiones acerca de cómo se tratan los datos en la comunicación.
- **Orientada a servicio:** La interacción entre sistemas se define en términos de los servicios expuestos entre ellos para establecer esa colaboración. Esta colaboración puede realizarse de múltiples formas desde la basada en "*peer-to-peer*" a técnicas cliente - servidor.
- **Basada en componentes:** El conjunto de servicios será ofrecido por componentes que pueden ser combinados para formar un servicio particular. Un componente puede proporcionar todos o una parte de los servicios.
- **Por capas:** El conjunto total de servicios requeridos para hacer un sistema e-learning será modelado como un conjunto de capas y cada capa proporcionará un conjunto claro de servicios definidos.
- **Comportamientos y Modelos de Datos:** Un servicio será definido en términos de sus comportamientos y su modelo de datos. Los comportamientos causarán cambios en el estado del modelo de datos y el

estado del modelo de datos sólo cambiará como consecuencia de un comportamiento claramente definido.

- **Múltiples ligaduras:** El modelo de información para una especificación (comportamiento y datos) puede ser apoyado por múltiples ligaduras como por ejemplo Java, XML, servicios Web, etc.

Antes de empezar a describir este *framework*, es interesante introducir la visión que IMS hace de la arquitectura de los sistemas de e-learning en general. Da dos visiones distintas de estos sistemas, una sería la arquitectura lógica y otra sería la física.

La arquitectura lógica está basada en un modelo multicapa o multinivel como el que se utilizará para describir la que se propone en esta tesis. Este modelo se puede ver en la Figura 2.9 y consiste en las siguientes capas:

- **Usuarios**: Representa al conjunto de usuarios de un sistema de e-learning, como los estudiantes, administradores, profesores, etc. Los usuarios tienen acceso al sistema a través de "agentes de usuario".
- **Agentes de usuario**: Los agentes que proporcionan los servicios a los usuarios. Los agentes de usuario son los sistemas e-learning que proporcionan la interfaz para tener acceso y actuar con todos los servicios e-learning. Los agentes de usuario incluyen sistemas como LMS, LCMS, herramientas de autor y contenido. Los agentes de usuario son construidos usando la colección de servicios proporcionados por la capa de servicios.
- **Herramientas**: Permiten el acceso a los diferentes servicios de una forma conveniente y fácil de usar. Esto incluye la evaluación, tutorización, simulación, etc.
- **Servicios de educación**: Incluyen todos los servicios y los componentes que tienen la funcionalidad específica de e-learning.
- **Servicios de soporte**: Servicios comunes tales como la autenticación, el descubrimiento de recursos, etc.
- **Repositorios digitales**: Para el almacenamiento del material docente.
- **Infraestructura de comunicaciones**: La interconexión básica y los servicios de transporte de datos que entregan la información punto a punto.

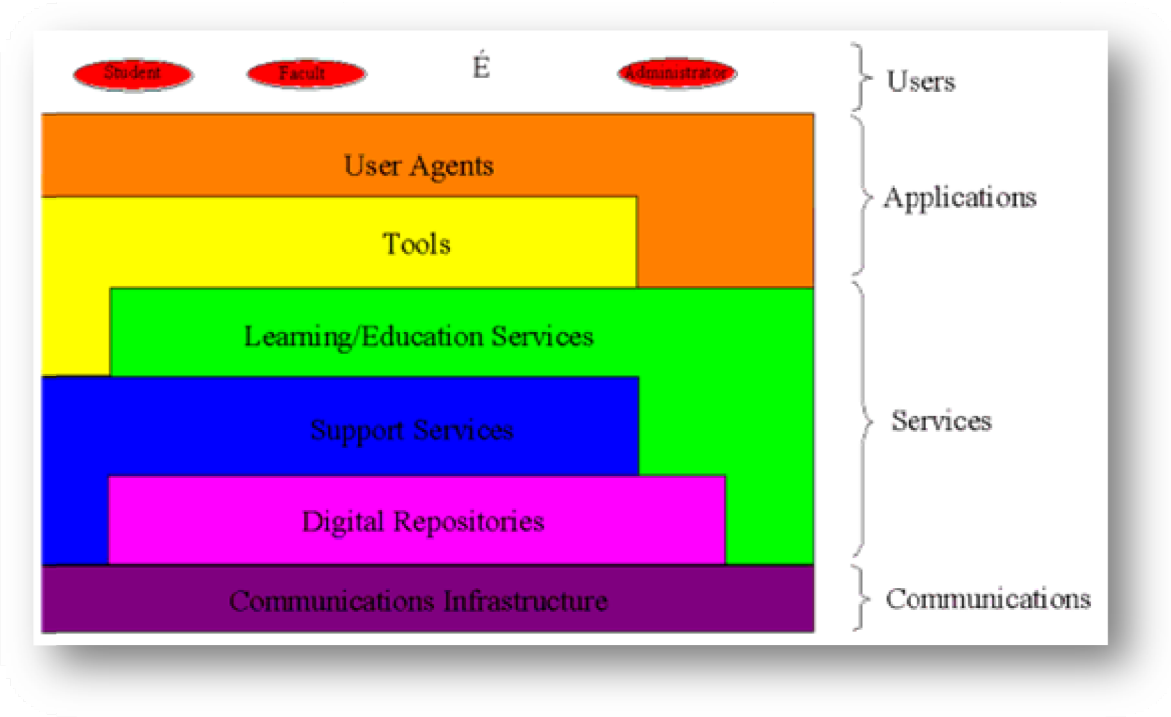


Figura 2.9 Arquitectura lógica de los sistemas de e-learning [IMS, 2003b]

La arquitectura física es la que se muestra en la Figura 2.10, está compuesta por las siguientes estructuras:

- **La red principal**: Red primaria que interconecta los sistemas informáticos principales. Estaría representada por Internet. Es una parte de la infraestructura de comunicaciones en el modelo lógico.
- **La red de acceso**: La red que une los dispositivos de entrega para la red principal. Ejemplos típicos son redes de cable, redes inalámbricas, etc. Es una parte de la Infraestructura de comunicaciones en el modelo lógico.
- **Repositorios digitales federados**: La serie de los recursos digitales que están disponibles en una variedad de repositorios digitales, bases de datos, servidores Web, etc. Representa la capa de repositorios digitales en el modelo lógico.
- **Motores de servicios de entrega**: Los sistemas responsables de la provisión de servicios de e-learning. Corresponde a los servicios de educación y de soporte en el modelo lógico.
- **Dispositivos de entrega**: Los dispositivos encargados de entregar el material docente al usuario. Corresponde a las herramientas y a los agentes de usuario en el modelo lógico.

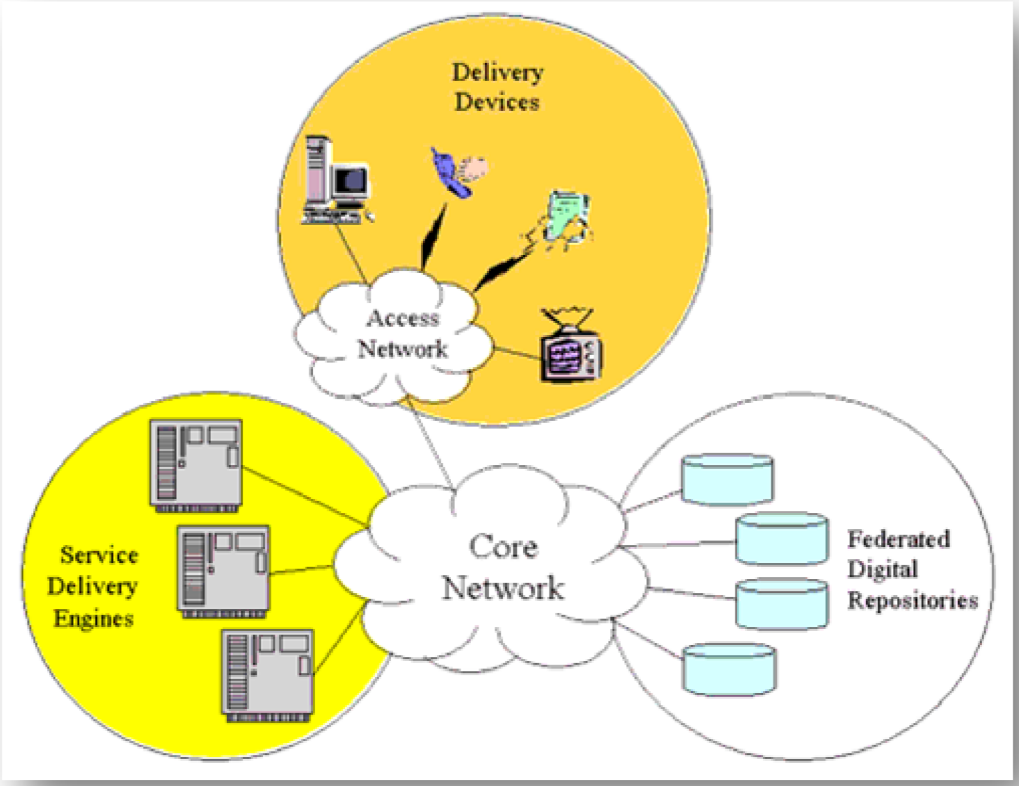


Figura 2.10 Modelo físico de los sistemas de e-learning [IMS, 2003b]

Una vez vista la arquitectura general que representa a los sistemas de aprendizaje, se pasa a describir el *framework* de IMS, que ha sido diseñado para que pueda ser usado en una amplia gama de arquitecturas. Este *framework* se representa como un modelo en capas, como se aprecia en la Figura 2.11; IMS justifica la adopción de este tipo de modelo debido a la gran proliferación de su uso en la definición de arquitecturas e-learning.

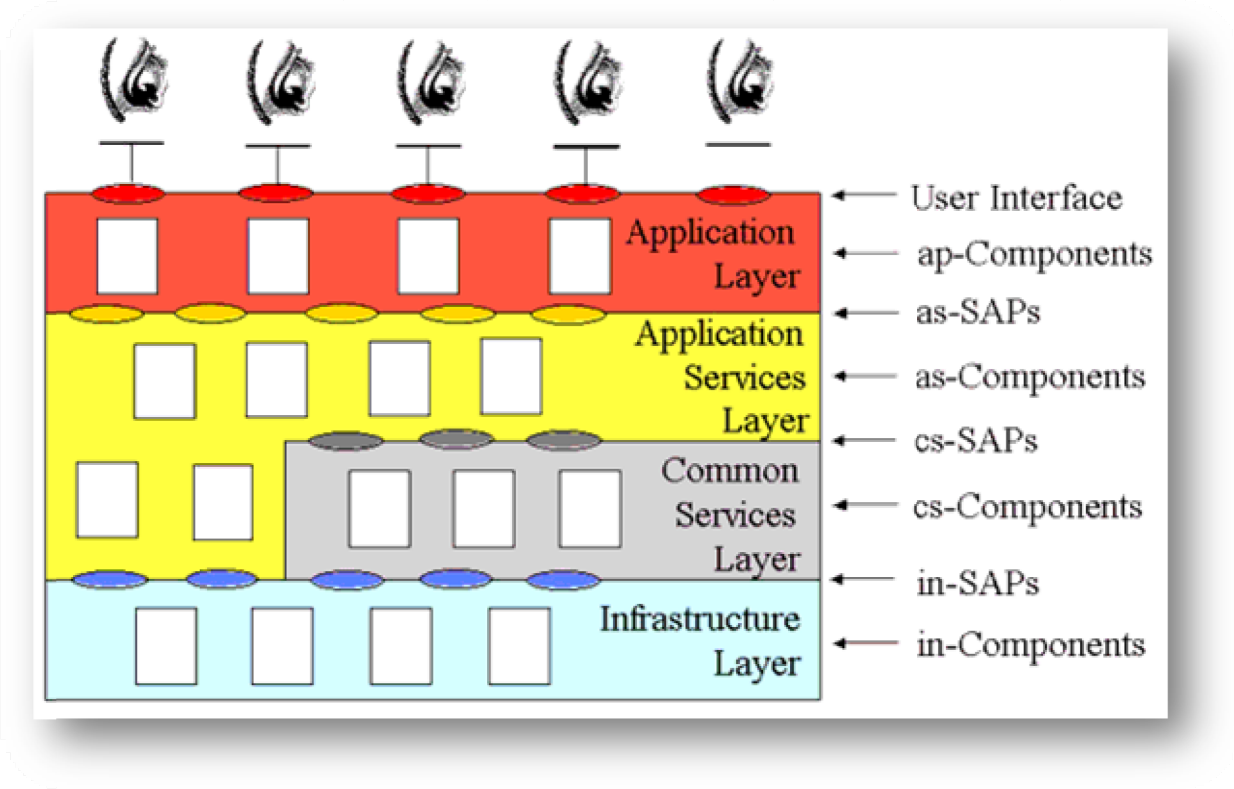


Figura 2.11 Modelo en capas del IMS Abstract Framework [IMS, 2003b]

Las principales características de cada una de las capas que definen este framework son las siguientes:

- **Capa de aplicación:** Sistemas, herramientas y aplicaciones que exponen los servicios de aplicación al usuario final proporcionando un conjunto particular de funcionalidades e-learning.
- **Capa de servicios de aplicación:** Conjunto de entidades que proporcionan los servicios específicos de funcionalidades e-learning. IMS se centra principalmente en la especificación de estos servicios.
- **Capa de servicios comunes:** Conjunto de entidades que proporcionan servicios generales que serán usados por los servicios de aplicación. Por ejemplo autenticación.
- **Capa de infraestructura:** Servicios encargados del intercambio de información (mensajes, transacciones, etc.).
- **Puntos de acceso a los servicios:** Los puntos de acceso o interfaces correspondientes a cada servicio.
- **Entidades:** Los procesos que se utilizan para representar un servicio particular. La implementación de una entidad con sus puntos de acceso de

servicio se denomina componente y su representación abstracta se llama clase.

El acceso a un servicio se realiza mediante su Punto de Acceso al Servicio (denominado en la figura SAP: *Service Access Point*) apropiado. Cada servicio tiene un SAP único. Un componente puede soportar uno o varios SAP (en una representación orientada a objetos un SAP podría ser soportada por una o varias operaciones donde la clase es la definición del servicio).

Uno de los principios de diseño para el IMS Abstarct Framework es la adopción de la orientación a servicio como forma de describir la funcionalidad de e-learning. El servicio se oculta detrás de un SAP y sólo se puede acceder usando ese SAP. La Figura 2.12 muestra una representación esquemática de un servicio.

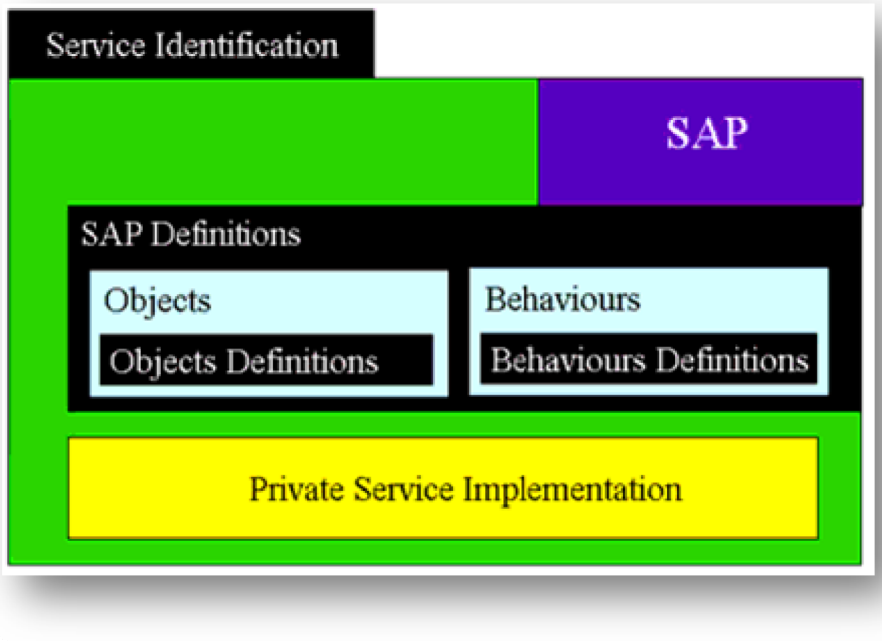


Figura 2.12 Descripción de un servicio en IMS AF [IMS, 2003b]

- El servicio tiene un punto de acceso de servicio claramente definido. Cada servicio tiene sólo un SAP. El SAP se define en términos de sus objetos constituyentes y sus comportamientos.

- El SAP puede consistir en uno o varios objetos y cada objeto va a tener, en general, más de una operación. Cada objeto es definido usando una definición de clase y consiste en un conjunto de atributos y operaciones. La operación describe como el estado de los atributos puede ser cambiado. El juego de comportamientos permitidos para cada clase también debe ser definido.
- La implementación de Servicio Privado está fuera del alcance del IMS AF. La única exigencia es que debe proporcionar todas las características apropiadas del servicio y nada más.

En la capa de infraestructura lo más importante es la interoperabilidad en el intercambio y manipulación de datos. En esta capa, las especificaciones de IMS se utilizan para definir las estructuras de datos y los comportamientos permitidos para el intercambio de esas estructuras de datos entre los componentes que forman la capa. Esta interoperabilidad se define en términos de intercambio de documentos XML. El framework también describe los mecanismos de transporte que se pueden utilizar para intercambiar los documentos XML.

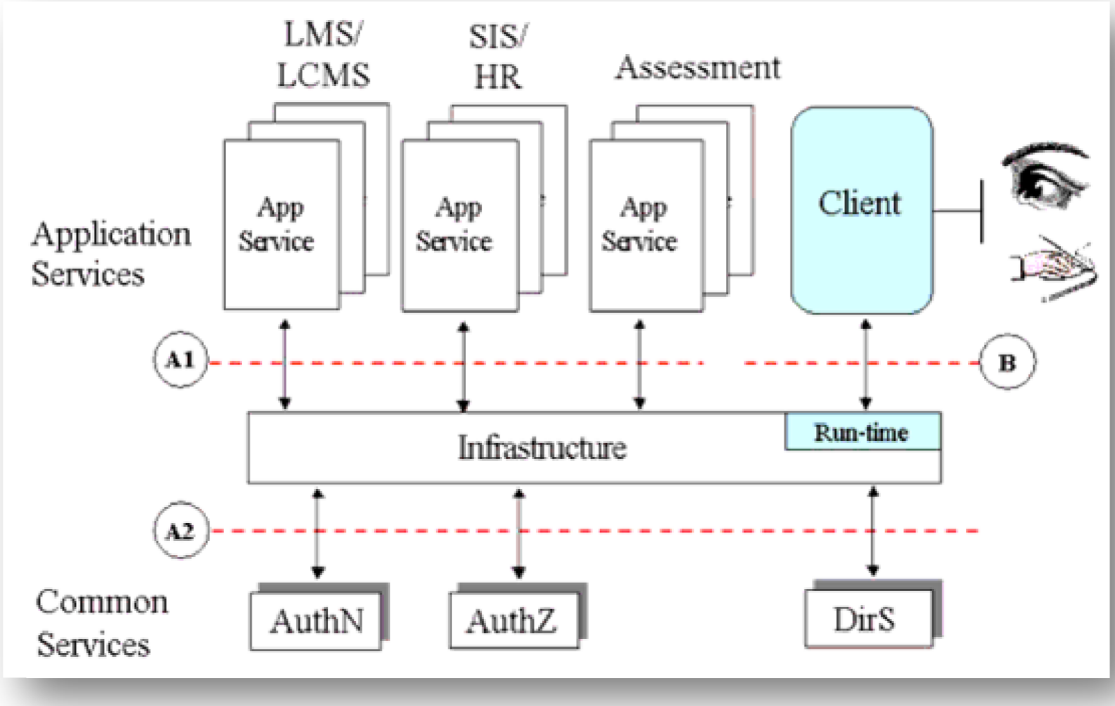


Figura 2.13 Interacción de los servicios en IMS AF [IMS, 2003b]

En una implementación distribuida de este *framework*, como en un ambiente de servicios Web, la interacción entre los servicios sería como la mostrada en la Figura 2.13. En este framework de interacción hay tres categorías de interfaces que deben ser apoyadas por la capa de infraestructura, como son:

- **El interfaz de Servicios de Aplicación (A1)**: este interfaz es usado para proporcionar la interoperabilidad entre servicios comunes de aplicación; por ejemplo, entre sistemas de empresa-a-empresa (B2B).
- **El interfaz de Servicios Común (A2)**: este interfaz es usado para proveer la interoperabilidad del conjunto de servicios comunes poniéndolos a disposición de los servicios específicos de aplicación; por ejemplo, la autenticación y la autorización
- **El interfaz de ejecución (B)**: este interfaz es usado para interconectar la ejecución del cliente con el proveedor de servicios remoto.

Hay dos tipos de comportamientos de interacción que tienen que ser especificados para asegurar la interoperabilidad:

- El paso de Mensajes - donde la información es intercambiada entre sistemas que usan alguna forma de paso de mensajes. El contenido y la secuencia de los mensajes definen el comportamiento esperado.
- Ejecución - donde los sistemas finales tienen que usar algún algoritmo predefinido sobre la información que les llega. Los datos determinarán los resultados de los algoritmos pero el comportamiento estará bien definido para todos los resultados posibles.

2.3.2. IMS Digital Repositories Interoperability

IMS DRI tiene como objetivo facilitar el acceso a los contenidos en los repositorios de contextos educativos, con los LMS y los LCMS, pero también con otros sistemas como los portales de búsquedas. Esta especificación se propone para la interoperabilidad entre servicios o aplicaciones, los cuales tienen las funciones más comunes de un repositorio: buscar, exponer, colectar, enviar, almacenar, pedir, entregar y alertar. Entre estas funciones, se reconocen cinco combinaciones como actividades principales: Buscar/Exponer, Exponer, Enviar/Almacenar, Pedir/Entregar y Alertar/Exponer, que pueden verse con detalle más adelante.

En la Figura 2.14 se encuentran resaltadas las funciones anteriormente comentadas, en la que se muestra la arquitectura funcional entre un sistema e-learning, los repositorios digitales y los servicios de información. En la misma figura se muestran los roles, los componentes funcionales y los servicios que definen el espacio de interacción completo.

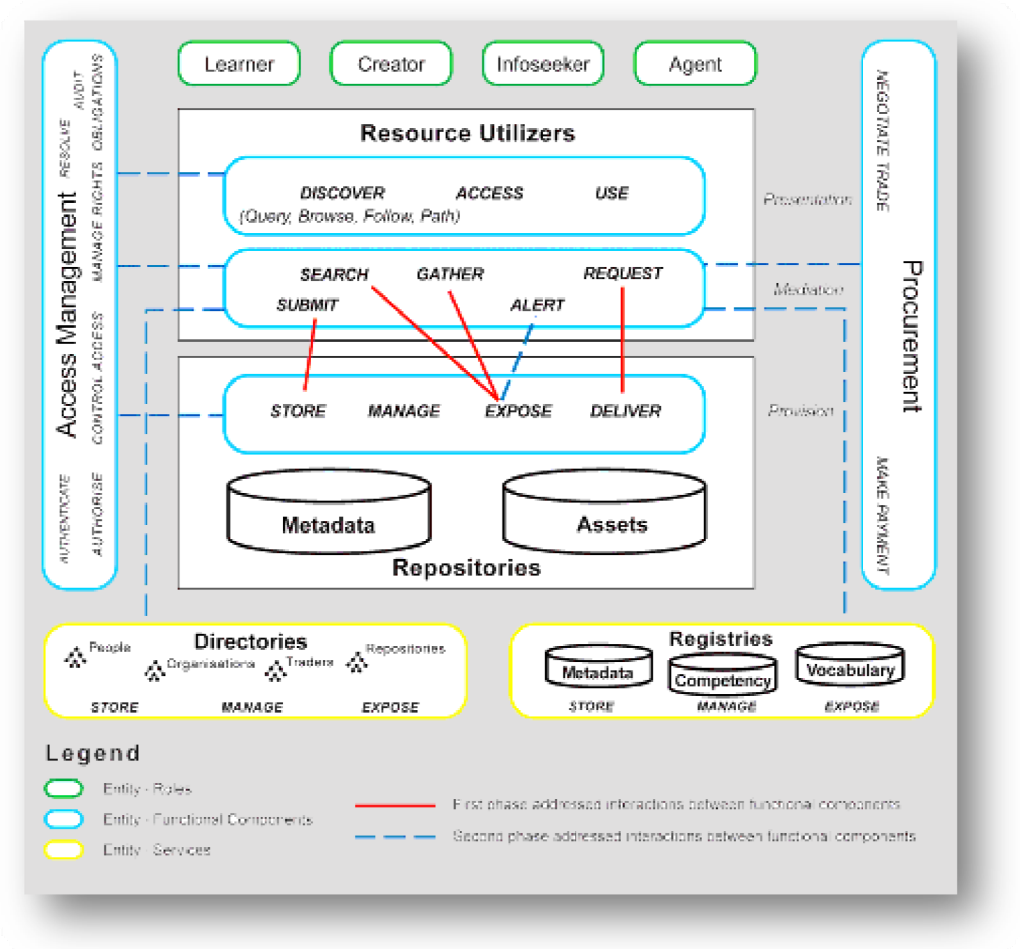


Figura 2.14 Arquitectura funcional de IMS DRI [IMS, 2003a]

La especificación considera que los repositorios digitales tienen diversidad de formatos en sus contenidos, diferentes sistemas, tecnologías y objetivos, por lo que no tiene requisitos para la operación interna de los repositorios y sus recomendaciones.

En la Tabla 2.7 se resumen las recomendaciones tecnológicas que la especificación hace para cada una de las funciones principales.

Función	Descripción	Recomendación tecnológica
Buscar/Exponer (Search/Expose)	Ejecuta la búsqueda de metadatos asociados con los contenidos que el repositorio expone.	XQuery para colecciones con metadatos en XML y Z39.50 para búsquedas en sistemas bibliotecarios.
Colectar/Exponer (Gather/Expose)	Define la solicitud de metadatos que el repositorio expone, la agregación de los metadatos para utilizarse en búsquedas subsecuentes y la agregación de metadatos para crear nuevos repositorios.	No hay recomendación específica, pero IMS DRI sugiere que OAI puede proveer una funcionalidad adecuada.
Enviar/Almacenar (Submit/Store)	Se enfoca a la manera en la que un objeto se mueve a un repositorio desde un sitio accesible por red y cómo el objeto será representado en el repositorio para su acceso.	Se recomienda el uso de paquetes IMS a través de SOAP.
Pedir/Entregar (Request/Deliver)	Permite, que una vez que el usuario a localizado los metadatos en la función Buscar, pueda solicitar al repositorio el acceso al recurso.	Recomendación general para utilizar HTTP y FTP para diferentes tipos de recursos. También IMS CP.
Alertar/Exponer (Alert/Expose)	Función clave, en la que a través de correo electrónico se notifica a los usuarios sobre eventos en el repositorio, pero no está contemplada todavía en esta versión de la especificación.	No hay recomendación específica ya que esta función sale del alcance de esta especificación.

Tabla 2.7 Recomendaciones tecnológicas para las funciones de un repositorio

En cuanto a los requisitos de conformidad para la interoperabilidad en esta especificación, se apunta que no los hay, ya que las soluciones para el intercambio de información entre repositorios se pueden realizar de distintas maneras y no es posible determinar de forma estricta estos requisitos. En su lugar, se da un listado de las

especificaciones y estándares que se recomienda seguir para elegir las aplicaciones y llevar a cabo una implementación que fomente la interoperabilidad.

2.3.3. CEN CWA Simple Query Interface

El Comité Europeo de Normalización (CEN), y más concretamente su comité técnico ISSS (*Information Society Standardization System*) comenzó en 2004 la especificación y experimentación de la especificación Simple Query Interface (SQI) [CEN, 2005]. Esta especificación está basada en el trabajo de un conjunto de Proyectos y Agencias Internacionales: ARIADNE, CELEBRATE, Edutella, Elena, EduSource, ProLearn, Universal/EducaNext y Zing. Este conjunto de instituciones agrupa a las más importantes cuando se habla de repositorios y e-Learning.

Esta agrupación significa una variedad de prioridades particulares así como de recursos existentes y tecnologías que hacen que los recursos estén disponibles. Un ejemplo claro de esto es el caso del proyecto Elena [Elena, 2009], el cual, explora las posibilidades de movimiento de "servicios de aprendizaje" en vez de "objetos de aprendizaje", es decir, eventos más que objetos, y muy específicos más que genéricos y reutilizables. Servicios como tutorías vía e-mail o lecturas por videoconferencia a través de agentes personales de aprendizaje. Todo esto contrasta con los repositorios de objetos de aprendizaje más convencionales como ARIADNE o EduSource.

En términos prácticos y como fondo, la necesidad de una nueva especificación surgió por la demanda existente de interoperabilidad entre sistemas; de forma que cualquier entorno pueda ejecutar una consulta sobre un conjunto amplio de sistemas distribuidos que no necesariamente compartan un lenguaje de consulta, un esquema de metainformación, o un protocolo de conexión. Por lo tanto se necesitaba un pequeño y simple criterio de especificación que fuera viable en cualquier entorno, esto es SQI.

Un protocolo importante como el Z39.50 y los estándares XQuery sugeridos por el IMS Digital Repositories Interoperability (DRI), son difíciles de implementar en varias de las herramientas mencionadas anteriormente.

Existen además algunas especificaciones que tratan el mismo tema, y no están relacionadas necesariamente con la comunidad educacional. La principal es Search/Retrieve for the Web (SRW), implementada como un servicio Web. Esto significa que transfiere información XML sobre HTTP. Un potencial problema con SRW es que solo

realiza consultas síncronas. Un sistema fuente lanza una consulta sobre el destino, y espera hasta que el destino le devuelva la respuesta(los resultados, si es posible).

En Schoolnet Europa (consorcio de Ministerios de Educación Europeos fundado en 1997 y que actualmente cuenta con 31 Ministerios [Schoolnet, 2009]) señalan que estos modelos pueden no trabajar demasiado bien para redes heterogéneas que involucran aplicaciones de consulta P2P como Edutella [2009]. En la práctica, una consulta síncrona como las definidas por SRW asume que la consulta proviene de una aplicación cliente simple que solo recibe y expone información de un servidor. Sin embargo la especificación SQI, aunque diseñada para ser simple tiene ambos modos: síncrono y asíncrono. Otro problema adicional de SRW es que, por ejemplo, algunos participantes de Elena o ProLearn no trabajan normalmente con XML o HTTP.

SQI, no es un lenguaje. No tiene o necesita un modelo de datos, o un lenguaje particular de consulta. Sin embargo, SQI es una API, con una lista coherente de comandos de programación. Solo tiene en cuenta el envío y la respuesta de consultas, pero no la estructura de las mismas. Esto significa que SQI es mínimo, y por eso relativamente sencillo de implementar en una gran variedad de sistemas.

SQI es una API para consultar repositorios de recursos de aprendizaje. Esta caracterizado por su sencillez y flexibilidad ya que permite ser desplegado fácilmente en una gran variedad de escenarios. SQI está basado en el concepto de gestión de sesión que permite diferenciar la autenticación de la gestión de consultas. Es neutral en cuanto a los lenguajes de consulta y el formato de resultados, soporta ambos modos de consulta, síncrono y asíncrono y puede ser implementado como un servicio con o sin estado.

Si se consideran dos repositorios que comparten al menos un lenguaje de consulta y un formato de metainformación, se necesitan los siguientes pasos para permitir que un repositorio (fuente de la consulta) consulte al otro (destino de la consulta) usando SQI:

- La fuente escoge uno de los lenguajes de consulta disponibles en el destino (por ejemplo, XQUERY - es posible pasar por alto este paso cuando el destino propone un lenguaje de consulta por defecto).
- La fuente selecciona uno de los formatos de resultados disponibles en el destino (por ejemplo, el marco de metainformación de objetos de

aprendizaje de IEEE, también aquí es posible pasar por alto este paso cuando el destino propone un formato de resultados por defecto).

- La fuente envía una consulta en el lenguaje de consulta seleccionado.
- Dependiendo del modo de consulta seleccionado, el destino provee el resultado de la consulta en el formato seleccionado como el valor de retorno de la llamada usada para enviar la consulta (modo síncrono) o llamando una o más veces al servicio que escucha los resultados de consultas, implementado por la fuente (modo asíncrono). El último modo es mucho más robusto y permite que SQI sea usado como la interfaz final de una búsqueda federada ya que no es necesario esperar el final de la consulta inicial antes de devolver los primeros resultados.

Especificación

La Web pone un gran número de recursos de aprendizaje al alcance de cualquiera con acceso a Internet. Sin embargo, muchos de estos recursos son difíciles de encontrar de una manera eficiente ya que se encuentran en sistemas de gestión del aprendizaje (contenido) cerrados y propietarios, en servidores de medios de comunicación y herramientas de colaboración online.

Tales sistemas son comúnmente llamados repositorios de objetos de aprendizaje y normalmente son parte de una Web educacional. Los repositorios de objetos de aprendizaje mantienen información de objetos de aprendizaje, es decir, metainformación para describir artefactos educacionales como cursos, tutoriales online, lecturas, libros de texto electrónicos, sesiones de tutoría, etc.

El principal problema que se produce entre estos repositorios de objetos de aprendizaje es debido a la falta de “entendimiento” entre ellos, lo que deriva en problemas de interoperabilidad. La interoperabilidad puede definirse como “la habilidad de dos o más sistemas o componentes para intercambiar información y usar la información que han intercambiado”. Para el usuario, la falta de interoperabilidad significa:

- Las aplicaciones y sus datos están aislados
- Se requieren datos de entrada redundantes

En cambio, la interoperabilidad asegura que los datos sean introducidos solo una vez en una aplicación y que puedan ser propagados automáticamente a otras aplicaciones. Como un escenario requiere un modelo semántico común, entonces los metadatos del repositorio A pueden ser también interpretados por el repositorio B. Adicionalmente, los sistemas interoperables están basados en protocolos comunes, con los que controlan las interacciones para hacer un intercambio de datos entre repositorios de manera eficiente.

Para conseguir interoperabilidad, se pueden utilizar diferentes tipos de protocolos. El marco de interoperabilidad en los repositorios de objetos de aprendizaje presentado en la Figura 2.15 distingue entre servicios de núcleo y servicios de aplicación. Los servicios de núcleo son necesarios, por ejemplo, para acordar en un procedimiento común la identificación unívoca de objetos de aprendizaje. Otros servicios de este tipo están relacionados con la autenticación de usuarios y repositorios, o con la creación y gestión de sesiones para interactuar entre aplicaciones.

Aplicaciones típicas que hacen a los repositorios interoperables son:

- El servicio de indexación, como un tipo de servicio de replicación, permite al repositorio A “poner” metainformación de objetos de aprendizaje en el repositorio B. Soporta el mantenimiento distribuido de la metainformación a través de operaciones de inserción, borrado o actualización.
- El servicio de obtención es un servicio donde el repositorio A “saca” metainformación del repositorio B.
- El servicio de consulta permite al repositorio A buscar en el repositorio B recursos de aprendizaje apropiados, de modo que la metainformación transferida encaje en una consulta especifica.
- El servicio de políticas de acceso asigna derechos de acceso a los objetos de aprendizaje almacenados en un repositorio remoto.
- El servicio de entrega interactúa con el repositorio donde el recurso de aprendizaje electrónico está almacenado y entrega el recurso al usuario final.

Los servicios de aplicaciones hacen uso de los servicios de núcleo. Por ejemplo, el servicio de núcleo de gestión de sesión puede ser requerido por el servicio de consulta.

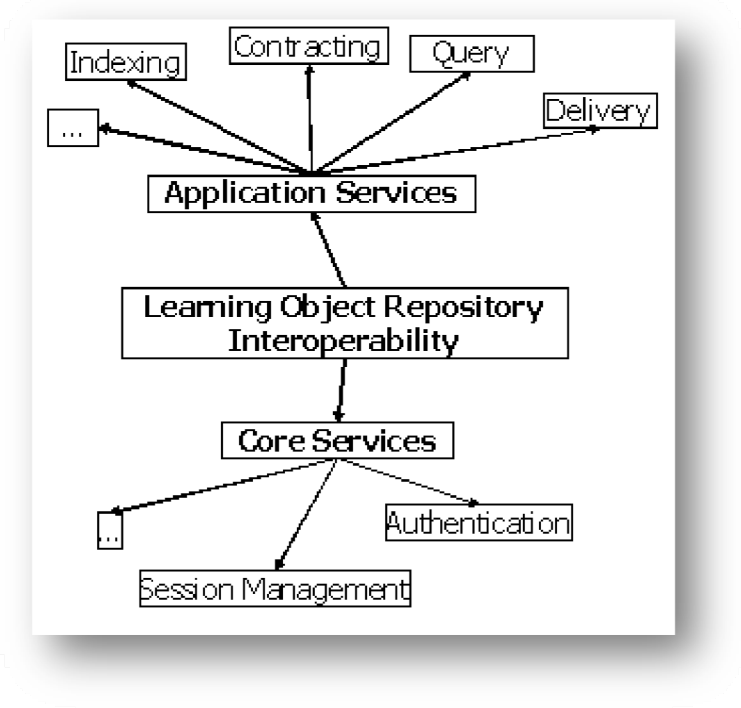


Figura 2.15 Marco de Interoperabilidad de un Repositorio de Objetos de Aprendizaje [CEN, 2005]

Ambos, servicios de núcleo y de aplicaciones, requieren una infraestructura común de mensajería, la cual permita a los repositorios interactuar. XML, RPC, Java RMI, y WSDL/SOAP son ejemplos de estos servicios de mensajería. Un servicio de mensajería está basado en una infraestructura común de red y protocolos de más bajo nivel como TCP/IP, HTTP, etc. La Figura 2.16 representa los diferentes protocolos involucrados en garantizar la interoperabilidad en los repositorios de objetos de aprendizaje.

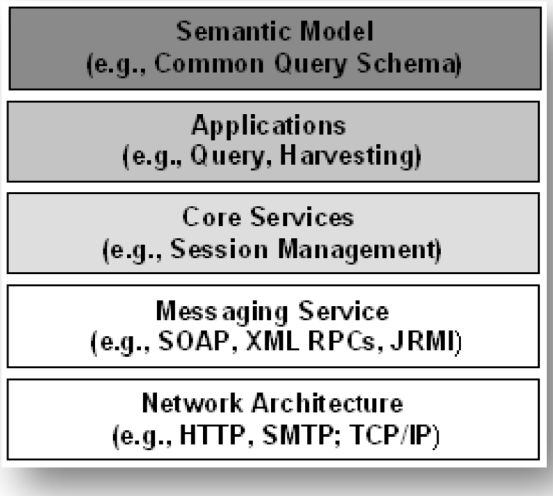


Figura 2.16 Pila de protocolos encargados de garantizar la interoperabilidad de un repositorio de objetos de aprendizaje

Los Sistemas de Gestión de Aprendizaje (LMS), Sistemas de Gestión del Contenido de Aprendizaje (LCMS), Pools de Conocimiento, son los tipos de tecnologías de la información para los cuales se han diseñado las interfaces propuestas en este apartado. Con su actual enfoque de interfaz de consulta, este documento tiene como objetivo a los arquitectos de redes educacionales, administradores de repositorios de recursos de aprendizaje, técnicos en la reutilización de objetos de aprendizaje, como también a los investigadores de servicios Web y de interoperabilidad entre sistemas.

Servicio de consultas

En el sentido de permitir encontrar recursos en repositorios interoperables, una interfaz de consulta, como la que se plantea en este apartado, es completamente necesaria. Aunque es posible referirse a la interoperabilidad de los repositorios de aprendizaje con un enfoque especial de la metainformación de los objetos de aprendizaje, esta interfaz de consulta también puede ser usada en otros dominios o escenarios de aplicaciones.

En los escenarios e-learning, los repositorios de aprendizaje pueden ser pares iguales, donde cada repositorio puede realizar consultas sobre otro. Para distinguir el sistema que consulta del consultado, el término “source” se introduce para etiquetar el sistema que emite la búsqueda (la fuente de la consulta); mientras que el término “target” etiqueta al sistema que es consultado (el destino de la consulta).

La metainformación puede ser almacenada usando diferentes mecanismos, como repositorios basados en archivos, Bases de Datos Relacionales, repositorios XML, o conjuntos de herramientas RDF, los cuales utilizan diferentes lenguajes de consulta constituyendo un entorno heterogéneo. Para hacer que los repositorios de aprendizaje sean interoperables, no solo se debe definir un interfaz común, también un lenguaje común de consulta y un formato de resultados común para las descripciones de los objetos de aprendizaje. Los aspectos de interoperabilidad como un esquema común de consulta y el formato de los resultados son parte del modelo semántico de una red educacional, como muestra la Figura 2.16.

El servicio de consulta manda una consulta en el lenguaje de consulta común al destino. Después los resultados de la consulta, representados en el formato común son

transportados a la fuente. En el nivel de implementación, la encapsulación puede ser necesaria para convertir una consulta de un lenguaje de consulta común X a un lenguaje de consulta local y transformar la consulta y los resultados de la consulta desde un formato propietario a uno común y viceversa.

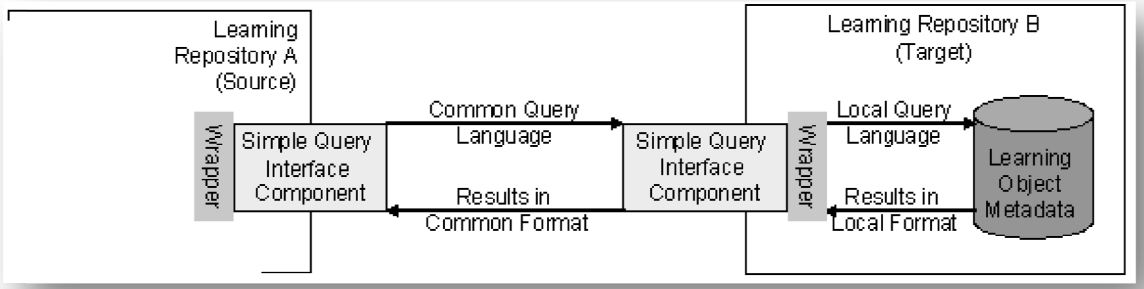


Figura 2.17 Comunicación a través de SQI entre dos componentes [CEN, 2005]

La Figura 2.17 ilustra un proceso de intercambio, donde el repositorio de aprendizaje A (la fuente) realiza una consulta al repositorio de aprendizaje B (el destino). Se asume que ambos sistemas tienen que estar de acuerdo de antemano sobre un lenguaje de consulta común. En el repositorio B, el componente de interfaz puede necesitar transferir la consulta desde un lenguaje de consulta común a uno local. También puede ser necesario realizar algunos mapeados desde el esquema común al propietario, antes de realizar la búsqueda. Esta tarea es realizada por un componente de encapsulación. Desde que la búsqueda obtiene resultados, el conjunto de resultados es remitido a la fuente, formateado de acuerdo al formato de resultados común.

Autenticación y gestión de la sesión

Las interfaces de aplicación realizan una abstracción de la autenticación y el control de acceso. Sin embargo, existe la necesidad de autenticar la fuente para permitir a un destino, por ejemplo, vincular políticas de consulta con el repositorio de la fuente. Por ejemplo:

- El repositorio A tiene permitido consultar el repositorio B sin ninguna limitación,

- El repositorio C solo tiene permitido recibir 1000 resultados de consulta por día desde el repositorio D como máximo.

Lo ideal es que la autenticación sea realizada solo una vez para una serie de interacciones. Para conseguir esto, se necesita devolver un elemento de sesión después de realizar con éxito una autenticación y así identificar al sistema en las subsiguientes comunicaciones.

Una vez que la sesión ha sido establecida se asume que la fuente tiene derecho a comunicarse con el destino. Para establecer una sesión se necesita un nombre de usuario y una contraseña u otro tipo de credencial. De esta forma, la identificación de un repositorio fuente previene a los repositorios destino a abrir sus sistemas a desconocidos, y permite políticas de consulta.

Una sesión es válida hasta que se destruye, por lo tanto, continúa activa después de que se haya ejecutado una consulta. Para destruir una sesión, se debe llamar al servicio *destroySession*. Como alternativa, el tiempo de sesión expira, cuando no tiene lugar ninguna comunicación durante 30 minutos (valor por defecto). Sin embargo, una sesión puede ser válida mucho más de 30 minutos, teniendo que ser destruida manualmente.

En cuanto al identificador de sesión se decide en el ámbito de la aplicación, como por ejemplo, un valor aleatorio de 128 bits. La especificación asume el uso de métodos seguros de autenticación, autorización y encriptación como por ejemplo SSL.

Especificación SQI

Esta especificación presenta un API de consulta, la cual puede implementarse sobre diferentes tipos de sistemas y entornos, aunque en esta Tesis se utilizará para aplicarla sobre repositorios de objetos de aprendizaje. El principal objetivo de diseño es mantener la especificación simple y fácil de implementar, de ahí su nombre, Interfaz Simple de Consulta. El esfuerzo colaborativo de combinar repositorios heterogéneos ha conducido a los siguientes requisitos:

- SQI es neutral en términos de formato de resultados y lenguaje de consulta. Los repositorios conectados vía SQI pueden ser de naturaleza heterogénea: por eso, SQI no asume formatos de resultados ni lenguajes de consulta.
- SQI soporta consultas sincronías y asíncronas con el objetivo de permitir la aplicación de la especificación SQI en casos de uso heterogéneos.
- SQI soporta implementaciones con o sin estado.
- SQI está basado en el concepto de gestión de sesión que separa aspectos de autenticación de la gestión de consultas.

El diseño de la API está basado en los siguientes principios de diseño:

- Principio de separación comando-consulta.
- Conjunto de comandos simple y extensible.

Con el objetivo de hacer uso completo de la funcionalidad de la implementación de SQI, dicha implementación debe complementarse con acuerdos sobre:

- El conjunto de atributos y vocabulario que puede usarse en la consulta.
- El lenguaje de consulta y su representación.
- La representación de la lista de objetos de aprendizaje que satisfacen la consulta.
- La representación de la metainformación de las instancias individuales de los objetos de aprendizaje.

Cualquier acuerdo entre dos o más repositorios es válido para SQI. Así los acuerdos pueden, por ejemplo, ser expresados con esquemas XML o esquemas RDF.

Aunque SQI no contribuye directamente a superar las diferencias entre varios paradigmas de gestión de la metainformación (Z39.50, enfoques basados en XML, comunidad RDF), apunta a ser una especificación independiente para todos los repositorios educacionales abiertos.

SQI puede desarrollarse en dos escenarios diferentes:

- **Escenario síncrono:** El destino devuelve los resultados de la consulta a la fuente. La recuperación de los resultados es iniciada por la fuente a través del envío de la consulta y a través de otros métodos se permite el acceso a los resultados de la consulta. Se puede ver en la Figura 2.18.
- **Escenario asíncrono:** La recuperación de los resultados la inicia el destino. Cuando se encuentra un número significativo de resultados coincidentes, dichos resultados son remitidos desde el destino a la fuente. Para soportar esta comunicación la fuente implementa un *listener* de resultados. La fuente debe ser capaz de identificar inequívocamente el envío de una consulta a un destino particular (incluso si la misma consulta es enviada a varios destinos). De lo contrario la fuente no es capaz de distinguir los resultados de búsqueda remitidos desde varios destinos y/o consultas previamente enviadas a dichos destino. Se puede ver en la Figura 2.19.

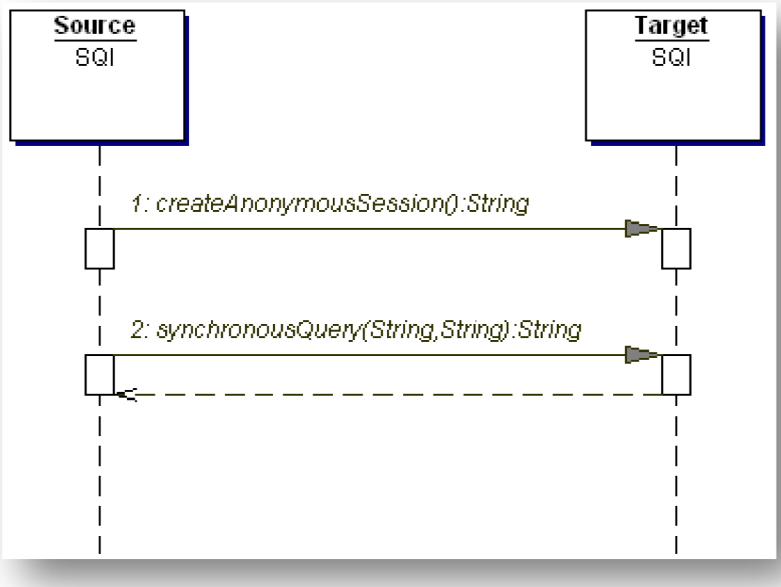


Figura 2.18 Consulta sobre un repositorio con el modo de consulta síncrono [CEN, 2005]

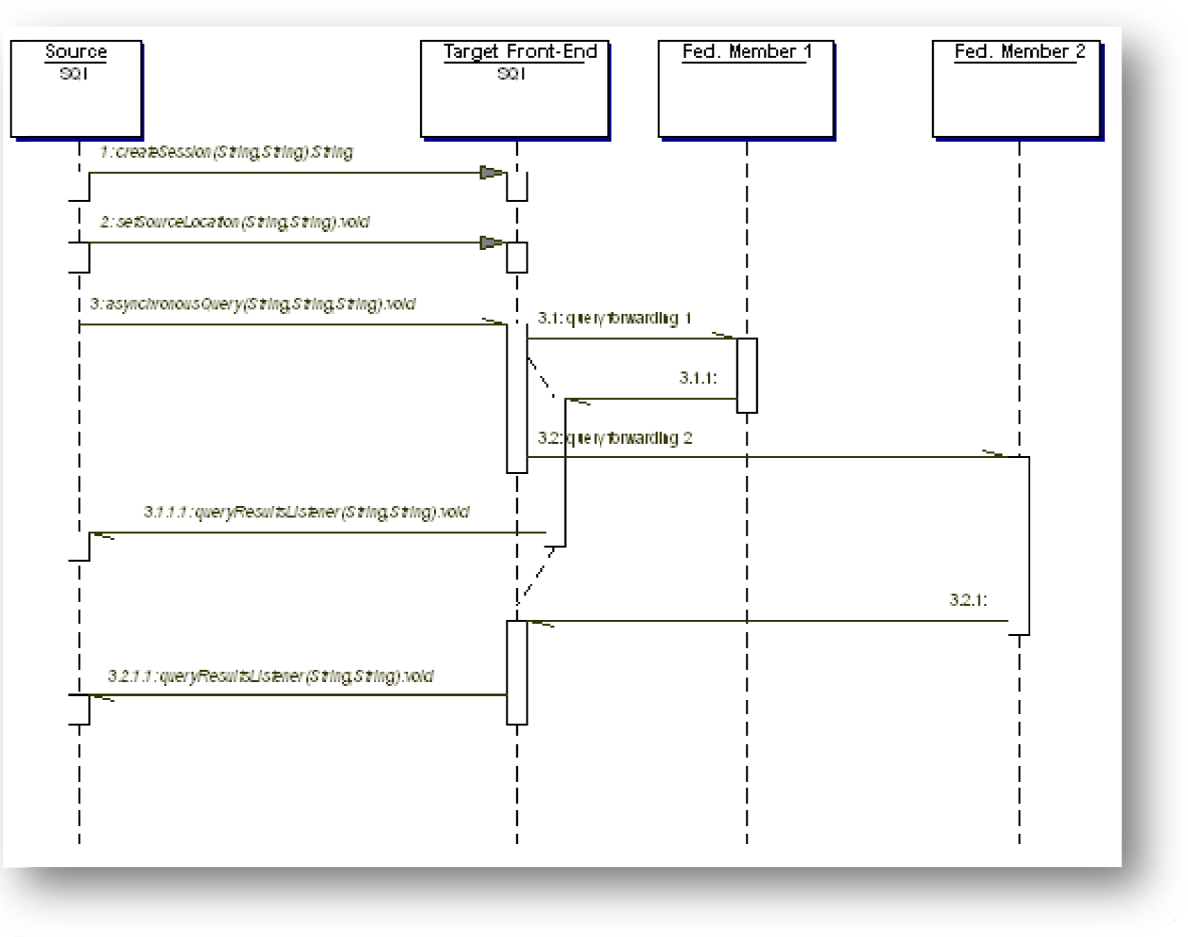


Figura 2.19 Búsqueda federada con el modo de consulta asíncrono [CEN, 2005]

Una interfaz de consulta operada en modo síncrono puede realizar múltiples consultas por sesión (incluso simultáneamente). En el caso de una interfaz de consulta operada en modo asíncrono, la fuente provee un identificador de consulta que permite emparejar resultados entrantes con la consulta enviada correspondiente (la fuente debe consultar varios destinos y cada destino deber contestar a la consulta, devolviendo más de un resultado a la fuente). También se puede dar el caso de que varias consultas puedan estar activas al mismo tiempo con una sesión en modo de consulta asíncrono.

El diseño de SQI está basado en el "principio de separación comando-consulta", como se indicó anteriormente. Este principio plantea que todos los métodos deben ser un comando que lleva a cabo una acción, o una consulta que devuelve datos, pero no ambos. Más formalmente, los métodos deben devolver un valor sólo si no causan efectos laterales. Esto conduce a un estilo de diseño que produce interfaces más claras y entendibles.

La separación comando-consulta (Command-Query Separation, CQS) es un principio de la programación orientada a objetos. Fue ideado por Bertrand Meyer como parte de su trabajo pionero, el lenguaje de programación Eiffel.

Para hacer que la interfaz sea fácilmente extensible, se minimiza el número de parámetros de los diferentes métodos más que el número de métodos adoptados. Variaciones de la interfaz (por ejemplo, la separación entre el esquema de consulta común y el formato de resultados común), pueden introducirse fácilmente añadiendo una función (por ejemplo, *getSupportedQuerySchema*), mientras que no se necesiten cambios en los métodos ya implementados. De este modo la compatibilidad se mantiene fácilmente.

Si ceñimos el estudio de SQI a la secuencia de acciones que se llevan a cabo en un entorno de consulta, el primer paso lo dará la fuente creando una conexión con el destino. Una vez que la sesión ha sido establecida, la interfaz de consulta en el destino esperará las peticiones de búsqueda.

Algunos métodos permiten la configuración de la interfaz en el destino. Parámetros de consulta como los que se citan a continuación, pueden ser establecidos con sus respectivos métodos:

- El lenguaje de consulta (*setQueryLanguage*).
- El número de resultados devueltos en un conjunto de resultados (*setResultsSetSize*).
- El número máximo de resultados de una consulta (*setMaxQueryResults*).
- La duración máxima en la ejecución de una consulta (*setMaxDuration*).
- El formato de los resultados (*setResultsFormat*).

Los parámetros que se establecen por medio de estos métodos continúan siendo validos durante toda la sesión o hasta que se vuelvan a cambiar. Si ninguno de estos métodos se invoca antes de que se envíe la primera consulta, se asumen los valores por defecto.

A continuación se muestra en la Tabla 2.8 el listado completo de métodos con los que cuenta SQI en relación con el establecimiento y liberación de la sesión, configuración de consultas y ejecución de las mismas.

	Implementado por el destino e invocado por el origen	Implementado por el origen e invocado por el destino
Gestión de la Sesión		
createSession	X	
createAnonymousSession	X	
destroySession	X	
Configuración de la Consulta		
setQueryLanguage	X	
setResultsFormat	X	
setMaxQueryResults	X	
setMaxDuration	X	
Consultas Síncronas		
setResultsSetSize	X	
synchronousQuery	X	
getTotalResultsCount	X	
Consultas Asíncronas		
asynchronousQuery	X	
setSourceLocation	X	
queryResultsListener		X

Tabla 2.8 Conjunto de métodos de la interfaz SQI

En la Tabla 2.8 se pueden identificar los siguientes grupos de métodos:

- **Gestión de la sesión**: En este grupo se encuentran los métodos encargados de crear y destruir las sesiones.
  - createSession: Este método permite crear una sesión autenticada (con usuario y contraseña normalmente, aunque se podrían utilizar otros métodos) con un destino.
  - createAnonymousSession: Este método permite crear una sesión anónima (sin autenticación) con un destino.
  - destroySession: Este método permite liberar una sesión anteriormente creada.
- **Configuración de la consulta**: En este grupo se encuentran los métodos encargados de configurar las futuras consultas que se realizarán sobre uno o varios destinos.

- setQueryLanguage: Permite especificar el tipo de lenguaje que se utilizará en la consulta al repositorio o sistema de búsqueda.
- setResultsFormat: Permite especificar el tipo de formato en el que se desea que estén formateados los resultados ofrecidos por el sistema de búsqueda.
- setMaxQueryResults: Permite especificar el máximo número de resultados que se desean recibir.
- setMaxDuration: Permite especificar el periodo de tiempo máximo (en milisegundos) que se desea que dure el proceso de búsqueda durante una consulta asíncrona.

- **Consultas síncronas**: En este grupo se encuentran los métodos encargados de realizar consultas síncronas sobre un destino, así como obtener sus resultados.
  - setResultsSetSize: Permite especificar el número máximo de elementos que se desean obtener durante una consulta simple.
  - sychronousQuery: Permite lanzar una consulta síncrona sobre un sistema. Del total de registros obtenidos en la consulta (setMaxQueryResults) se devolverán en cada petición tantos como se haya indicado a través del método setResultsSetSize, empezando por uno que se recibirá como parámetro en este método de consulta.
  - getTotalResultsCount: Permite obtener el número de resultados válidos que ha ofrecido una consulta.
- **Consultas asíncronas**: En este grupo se encuentran los métodos encargados de realizar consultas asíncronas sobre un destino, así como obtener sus resultados.
  - asynchronousQuery: Permite lanzar consultas asíncronas de búsqueda, de forma que el sistema que realiza la llamada podrá realizar otras tareas mientras se procesa la solicitud. En este caso será importante el valor asignado por el método setMaxDuration.
  - setSourceLocation: Este método deberá ser invocado antes de lanzar una consulta asíncrona. Será utilizado para especificar dónde está el sistema que lanza la consulta y que espera recibir los resultados.
  - queryResultsListener: Permite la devolución de los resultados al origen.

Si se continúa con la secuencia SQI, una vez establecida la sesión y configuradas las consultas, será momento entonces de ejecutarlas. La fuente puede enviar una consulta ya sea con el método *synchronousQuery*, o con a*synchronousQuery*. La consulta se

procesa en el destino y produce un conjunto de registros, denominado *ResultsSet*. La consulta se expresa en un lenguaje de consulta identificado a través de un parámetro de configuración de la consulta. En la consulta, se debe hacer referencia a un esquema común. En el modo síncrono, los resultados de la consulta son directamente devueltos por el método *synchronousQuery*. El método *getTotalResultsCount* devuelve el número total de registros de metainformación coincidentes en el proceso del destino. En el caso de una interfaz de consulta asíncrona, el método *queryResultsListener* lo invoca el destino para mandar los resultados de la consulta a la fuente.

Para indicar situaciones anómalas (por ejemplo, parámetros erróneos, o fallos llevando a cabo una operación), se incorporan las *SQIFault*, que pueden lanzar todos los métodos SQI. Un sistema de códigos de error permite documentar estas situaciones anómalas.

Query Languages

En este apartado se realiza una revisión de los diferentes lenguajes de consulta que acepta actualmente SQI y los que están siendo investigados o faltan por implementar en los diferentes sistemas SQI.

- **VSQL:** Se trata del lenguaje más sencillo para realizar consultas, y es aceptado por todas las plataformas o sistemas que implementan SQI. Este lenguaje es utilizado para enviar una lista de términos en una consulta. No acepta operadores lógicos ni expresiones de ningún tipo y su sintaxis es la siguiente:

  <simpleQuery><term>termino1</term>…<term>terminoN</term></simpleQuery>

  Esta consulta es interpretada como conjunción en la búsqueda, de la siguiente manera:

  termino1 AND…AND terminoN

  Como se puede apreciar el tipo de consultas actualmente aceptado en las comunicaciones SQI es muy limitado, aunque existen otros lenguajes de consulta que aún no están implementados, pero que como se explica a continuación dotarán a las consultas de mecanismos y posibilidades mucho más avanzadas, consiguiendo así que las búsquedas sean más eficientes y precisas.

- **ProLearn Query Language v1.0**: ProLearn Query Language (PLQL) es un lenguaje de consultas para repositorios de objetos de aprendizaje definido en el contexto de la red ProLearn. Como ya se ha comentado, SQI es un estándar de transporte de consultas y una de las características que lo identifica, es la independencia del lenguaje de consulta. Pero detrás de SQI existe la necesidad de disponer de un lenguaje de consulta específico, diseñado con el objetivo de recuperar objetos de aprendizaje de colecciones en repositorios de objetos de aprendizaje posiblemente heterogéneos. PLQL tiene como objetivo convertirse en este lenguaje de consulta específico.

  Fundamentalmente se trata de un formato de intercambio de consultas, usado en aplicaciones de enseñanza electrónica (o clientes PLQL) para realizar consultas en repositorios de objetos de aprendizaje (o servidores PLQL); donde cada objeto puede describirse mediante metainformación cumpliendo cualquiera de los estándares más populares como Dublín Core o LOM. Para el envío de la consulta y la recuperación de los resultados, la aplicación puede utilizar el protocolo SQI como cualquier otro estándar de interoperabilidad. En la parte del cliente PLQL se definen interfaces de usuario, las cuales pueden ser desde interfaces complejas hasta tipos simples basados en palabras clave. En la parte del servidor PLQL se construyen adaptadores PLQL para los repositorios locales.

  PLQL sugiere la combinación de búsqueda exacta, seleccionando objetos de aprendizaje a través de su metainformación, y búsqueda aproximada, recuperando los objetos mediante descripciones aproximadas en su contenido. De este modo una consulta PLQL puede contener tanto cláusulas exactas como cláusulas aproximadas, cada una de ellas con su sintaxis.

  PLQL también tiene en cuenta los niveles de complejidad que puedan tener o soportar los diferentes repositorios. Es por esto que uno de los principales aspectos del diseño del lenguaje es disponer de niveles progresivos, dependiendo del nivel de expresividad de las consultas, por lo que los repositorios más simples podrán utilizarán los niveles más bajos. La estructura del resultado devuelto por una consulta PLQL también está definido en niveles y se describe más adelante.

A continuación se definen los distintos niveles con los que cuenta PLQL. Por niveles, el nivel cero es muy básico y corresponde a la búsqueda simple por aproximación. Por el contrario el nivel cinco el más complejo y todavía no está definido, pero se asume que incluirá todas las características deseadas para el formato de intercambio de consultas. A

continuación se presentan las principales características de cada uno de estos tres niveles desarrollados hasta el momento:

- **NIVEL 0:** Esta capa permite expresar consultas aproximadas. El destino debe contener todos los términos de búsqueda especificados en la consulta, los cuales pueden estar tanto en las descripciones de metainformación, como los objetos de aprendizaje representados a través de estructuras adecuadas de recuperación de información (por ejemplo, índices). Cuando muchas cláusulas de aproximación están contenidas en la misma consulta, se consideran en conjunción, luego los objetos de aprendizaje tienen todas las palabras clave presentadas en la cláusula de aproximación para ser seleccionados; los objetos seleccionados son ordenados de acuerdo a la acumulación de relevancia de las palabras clave, la ordenación la lleva a cabo el motor de búsqueda del servidor.

  La capa cero ofrece la el mismo nivel de expresividad que VSQL, el lenguaje de consulta soportado por todos los destinos que implementan SQI, un ejemplo de VSQL sería:

  <simpleQuery>

  <term>Objeto de Aprendizaje</term>

  <term>Java</term>

  </simpleQuery>

  Consulta equivalente a:

  "Objeto de Aprendizaje" and "Java"

- **NIVEL 1:** En el nivel uno, además de las búsquedas aproximadas soportadas por el nivel 0, las consultas PLQL pueden expresar búsquedas exactas sobre campos de metainformación. Esto último se denota en el sentido de paths (rutas), entendiendo como path, simples concatenaciones de elementos (separados por puntos), empezando desde la raíz y sin omisiones; las expresiones y los paréntesis no están permitidos en este caso.

  El nivel 1 sólo soporta las siguientes raíces (en minúsculas): ‘dc’ (Dublin Core Metadata Element Sete), ‘lom’ (Dublin Core Metadata Element Set), y ‘mpeg’ (Moving Picture Experts Group). Por el contrario no soporta espacios de nombres genéricos.

Este nivel no entiende de tipos sin embargo, se permiten cadenas codificadas que representen diversos tipos de datos, dado que sus significados pueden entenderse haciendo una referencia a sus metaesquemas vía URI. De igual manera se permiten ciertas expresiones haciendo referencia también a sus metaesquemas.

Cuando se presentan muchas cláusulas exactas en la misma consulta, se consideran en conjunción y cuando cláusulas exactas y aproximadas están en una misma consulta, se asume que la búsqueda exacta tiene mayor prioridad que la búsqueda aproximada. La semántica de PLQL en el que caso de disponer de una cláusula exacta y una aproximada, aplica primero la cláusula exacta para conseguir un conjunto de resultados inicial, y después aplica la cláusula aproximada sobre este conjunto de resultados inicial produciendo el conjunto de resultados final.

Sin embargo las consultas exactas puede que no sean analizadas sintácticamente de manera correcta debido a la metainformación disponible en el servidor de almacenamiento. Cuando se produce esto, que varias cláusulas exactas no se analizan correctamente en el servidor, se debe devolver un código que indique que no se han ejecutado. En particular, si esto ocurre, el resultado de la búsqueda exacta es null; el repositorio debe operar sobre todo el conjunto de objetos de aprendizaje, como si la búsqueda exacta no se hubiese establecido.

Como variante de estas semánticas, a petición de la aplicación en tiempo de presentación de la consulta, el repositorio puede permitir el uso de valores constantes en las cláusulas exactas que no se han analizado correctamente, como palabras clave, para de este modo establecer una búsqueda aproximada basada en los términos indicados en las cláusulas exactas; se debe indicar con un código a la aplicación que se ha producido esta situación.

- **NIVEL 2:** Comparado con los niveles 0 y 1, el nivel 2 incrementa el nivel expresivo de las consultas soportadas, permitiendo la disyunción además de la conjunción; además, las cláusulas pueden utilizar predicados de comparación arbitrarios. Con la búsqueda aproximada, se utiliza el símbolo "=" para denotar el operador 'incluye' y el símbolo "exact" para denotar comparación de cadena exacta. El nivel 2 permite un conjunto limitado de estructuraciones de cláusulas exactas, soportando el uso de paréntesis en expresiones con rutas; en este sentido, es posible descender en una estructura jerárquica hasta nodos dados y construir condiciones basadas en propiedades de varios de los descendientes de un nodo.

2.3.4. CEN CWA Simple Publishing Interface

El Comité Europeo de Normalización (CEN) desarrolló, como se ha visto en el anterior apartado, la especificación SQI en 2004. Sin embargo esta especificación no determina nada acerca de cómo debería ser el proceso normal de publicación de contenidos en repositorios distribuidos. Por este motivo comienza en 2007 la publicación de otra interfaz a la que denominó SPI (Simple Publishing Interface).

En este caso se trata de una API para la publicación de datos y metadatos en un repositorio. Provee un protocolo simple y fácil de implementar y de integrar en aplicaciones ya existentes. Entre sus principales características se destaca que:

- SPI es neutral en términos de estándares de metainformación.
- SPI define una interfaz abstracta admitiendo interoperabilidad semántica. El punto de enlace de los servicios Web que detalla la norma proporciona una forma de enlazar esta interfaz con una implementación concreta y así proporcionar la interoperabilidad técnica.

SPI define los siguientes métodos:

- Create Identifier: Este método se usará para obtener un identificador para un nuevo recurso en el protocolo especificado. Este identificador será usado al invocar a cada uno de los cuatro métodos que se detallan a continuación.
- Submit/Delete Metadata Record: Métodos para insertar o eliminar descripciones de objetos respectivamente. Dichas descripciones irán detalladas como datos de tipo cadena de caracteres (String).
- Submit/Delete Resource: Métodos para subir o eliminar recursos respectivamente. El recurso como tal, será especificado en el método como un tipo *Base64Binary*, de forma que admitirá cualquier tipo de recurso.

2.3.5. IMS Digital Learning Services Standards

Se trata de una nueva suite de estándares involucrados en garantizar la interoperabilidad, ofreciendo cambios en sistemas empresariales, servicios Web y software basado en servicios entre otros. La integración de la empresa es uno de los principales desafíos que tiene la enseñanza superior de las TI, especialmente entre sistemas SIS (*Student Information System*) y LMS (*Learning Management System*). Esta suite la componen las siguientes especificaciones:

- IMS Common Cartridge – CC [IMS, 2008]: Para organizar y distribuir contenido educativo.
- IMS Learning Tools Interoperability – LTI [IMS, 2006]: Para la interoperabilidad entre aplicaciones, sistemas y mash-ups o aplicaciones Web híbridas.
- IMS Learning Information Services – LIS [IMS, 2007]: Trata sobre información referente al aprendizaje: privilegios y resultados.

Aunque la existente especificación IMS ES (*Enterprise Services*) V1.0 aporta algunas ayudas a este sentido, son necesarias otras funcionalidades que la especificación no puede soportar, sobre todo en temas semánticos.

La especificación IMS Enterprise Services [IMS, 2004a] es la definición de cómo los sistemas de aprendizaje son capaces de gestionar el intercambio de información concerniente a las personas, grupos y componentes del contexto de aprendizaje. Esta especificación se basa en los siguientes conceptos:

- **Interoperability**: Intercambio de información entre diferentes sistemas.
- **Service-oriented**: Intercambio de información desde el punto de vista de los servicios que son suministrados gracias a la colaboración de los sistemas. Esto adquiere la forma de servicios de gestión de personal, grupos y miembros.
- **Component-based**: Es el conjunto de servicios que serán suministrados para conformar un rango de servicios. Los servicios de gestión de personal, grupos y miembros pueden combinarse para proporcionar otros servicios a los cuales se añadirá más procesos en futuras versiones.
- **Layering**: Define el comportamiento y modelo de datos entre las diferentes capas de los servicios.
- **Multiple Bindings**: Su modelo de información se define empleando el lenguaje UML, lo cual permite contrastar de una manera fiable el modelo de

información en una amplia variedad de marcos (entornos). Los marcos de mayor importancia son los lenguajes de descripción de servicios Web (WSDL).

- **Adoption**: Este estándar se basa en las especificaciones originales. Aun existiendo múltiples y significativos cambios entre ellos, la base del modelo de datos ha mantenido las estructuras de las personas, grupos o miembros, de la base original.

En lo referente al modelo abstracto de esta especificación, se puede indicar que está formado por una estructura como la que se muestra a través de la Figura 2.20.

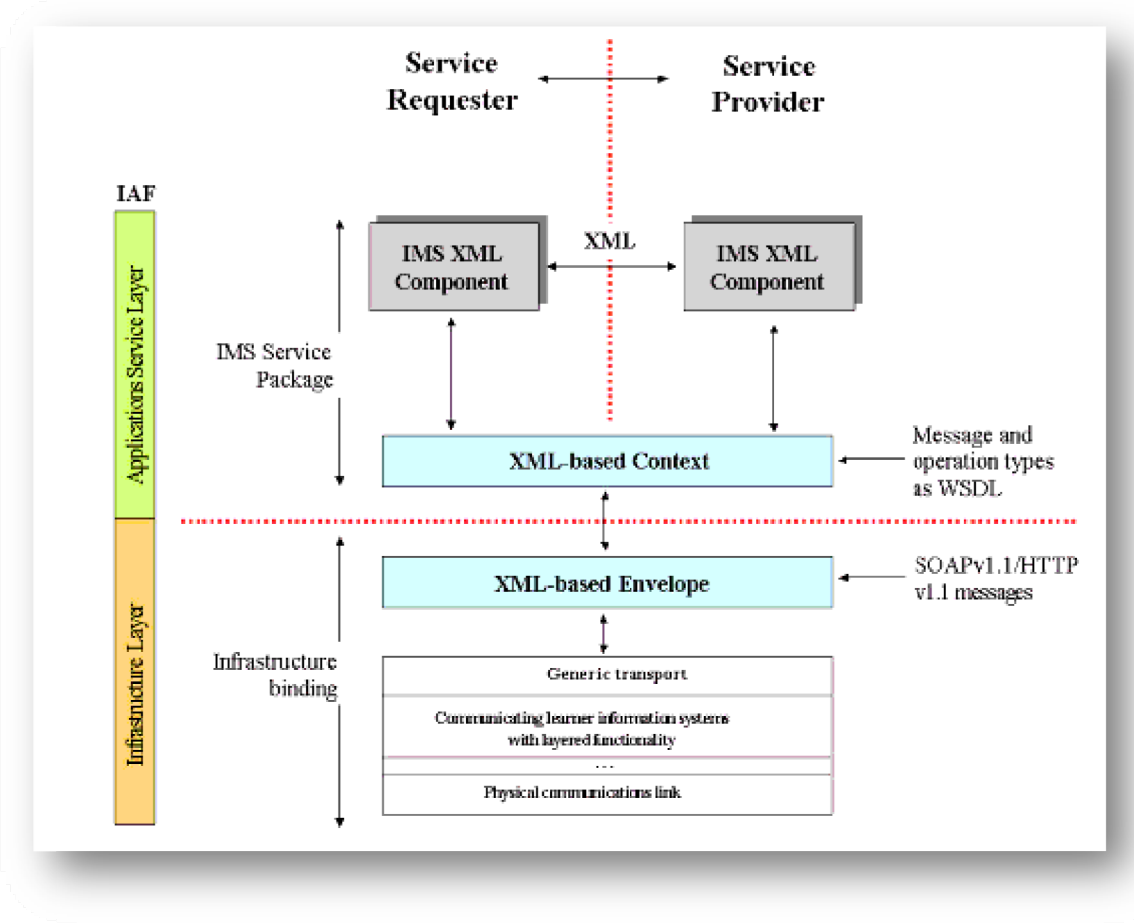


Figura 2.20 Modelo de la Arquitectura Enterpise Services [IMS, 2004a]

Como se muestra en la Figura 2.20, el alcance del modelo *Enterprise Services* quedaría representado por la parte superior del modelo a partir de la línea de puntos horizontal. Antes de comentar las diferentes partes de la figura, es importante indicar que con este gráfico IMS no pretende hacer una representación de cómo debería estar construido un sistema empresarial. La especificación IMS Enterprise Services contiene un modelo abstracto basado en mensajería, encargado de ser la base para el intercambio de

información entre los distintos sistemas. La comunicación entre la capa de infraestructura y la capa de servicios de aplicación se realizaría a través de un sistema de mensajería basado en SOAP, llamado IMS Service Package.

El intercambio de información entre el sistema que realiza la petición y el que proporciona la respuesta, se realizaría, a través de un intercambio de información tipada utilizando para ello XML. Será necesario simplemente indicar los datos a intercambiar entre los dos sistemas, las operaciones que pueden ser invocadas y la secuencia de operaciones permitidas.

IMS también da ciertas especificaciones que deberían cumplir tanto los sistemas invocadores de peticiones, como los que van a procesarlas para retornar la información solicitada.

IMS Common Cartridge

Esta reciente especificación (data de Julio de 2008) surge para aportar los siguientes beneficios:

- Mayores posibilidades de elección de contenidos: Permite colecciones de recursos de aprendizaje de diversos tipos y fuentes.
- Reduce problemas de bloqueos entre el vendedor de contenidos y la plataforma: Establece paquetes de cursos en formatos nativos aprobados por los editores, así como soporta una amplia variedad de formatos de contenidos, eliminando esos bloqueos en las plataformas.
- Mayores opciones de evaluación: apoya explícitamente el estándar de mayor utilización para el intercambio de normas y evaluación (IMS QTI) proporcionando un estándar de calificación y seguimiento que no requiere de la complejidad o la carga de un CBT (Computer Based Training – Formación basada en ordenador).
- Aumenta la flexibilidad, el intercambio y la reutilización: encaja en el contexto educativo de los instructores, de forma que permite ensamblar lecciones formadas por varios recursos, pudiendo ser estos de diversos tipos, pudiendo publicarlas, de forma que se aporta reutilización de contenidos fáciles de crear compartir y mejorar.

- Proporciona un empaquetado flexible a través de referencias a URL's con contenido Web: Basado en la especificación IMS Content Packaging V1.2, IMS Common Cartridge permite incorporar contenido "virtual" de otras organizaciones a través de URL's, mejorando con ello el tamaño de los paquetes e incrementando al mismo tiempo la flexibilidad.
- Soporta la colaboración y la Web 2.0 (a través de futuras provisiones de IMS LTI): Incluye una especificación para el intercambio de discusiones online o foros. Ha sido diseñado para permitir la incorporación futura de IMS LTI de forma que se permita el lanzamiento de servicios Web así como el intercambio de datos de aplicaciones de aprendizaje distribuidas y sistemas dentro de IMS Common Cartridge.
- Soporta contenido protegido a través de la autorización de recursos: Los paquetes o partes de los paquetes pueden ser protegidos a través de un estándar o protocolo de autorización abierto.
- Permite sencillas migraciones de SCORM 2004: IMS Common Cartridge y SCORM 2004 están ambos basados en IMS Content Packaging permitiendo el uso de herramientas para convertir desde SCORM a IMS Common Cartridge.
- Respaldado por una comunidad que proporciona herramientas para la implementación y la conformidad: la alianza Common Cartridge dispone de código abierto y código de la comunidad para automatizar el testeo de los paquetes, el testeo de las plataformas de aprendizaje, las referencias de los servicios de autorización, futuras incorporaciones de LTI y mucho más.

Se podría resumir viendo las ventajas anteriormente expuestas que la especificación IMS Common Cartridge está formada por:

- IEEE LOM (metadatos)
- IMS Content Packaging v1.2 (empaquetado de contenidos)
- IMS Question & Test Interoperability v1.2.1 (cuestionarios de evaluación)
- IMS Authorization Web Service v1.0 (autorización de acceso)

Con el Common Cartridge, IMS ha optado por simplificar lo máximo posible y dejar de lado la mayoría de características opcionales y extensiones. Así, para los metadatos sólo se usan los quince elementos de Dublin Core (mapeados a los elementos correspondientes de LOM) y en cuanto a QTI sólo se contemplan los seis tipos de preguntas más comunes. Se han añadido, en cambio, nuevas características cuando se ha estimado necesario:

- Un nuevo tipo de recurso que sirve para iniciar un fórum de debate.
- Un protocolo de autorización (IMS Authorization Web Service) que permite al editor de un paquete controlar el acceso a sus contenidos.

Por lo tanto si se siguiera esta especificación, el contenido del paquete quedaría como muestra la Figura 2.21.

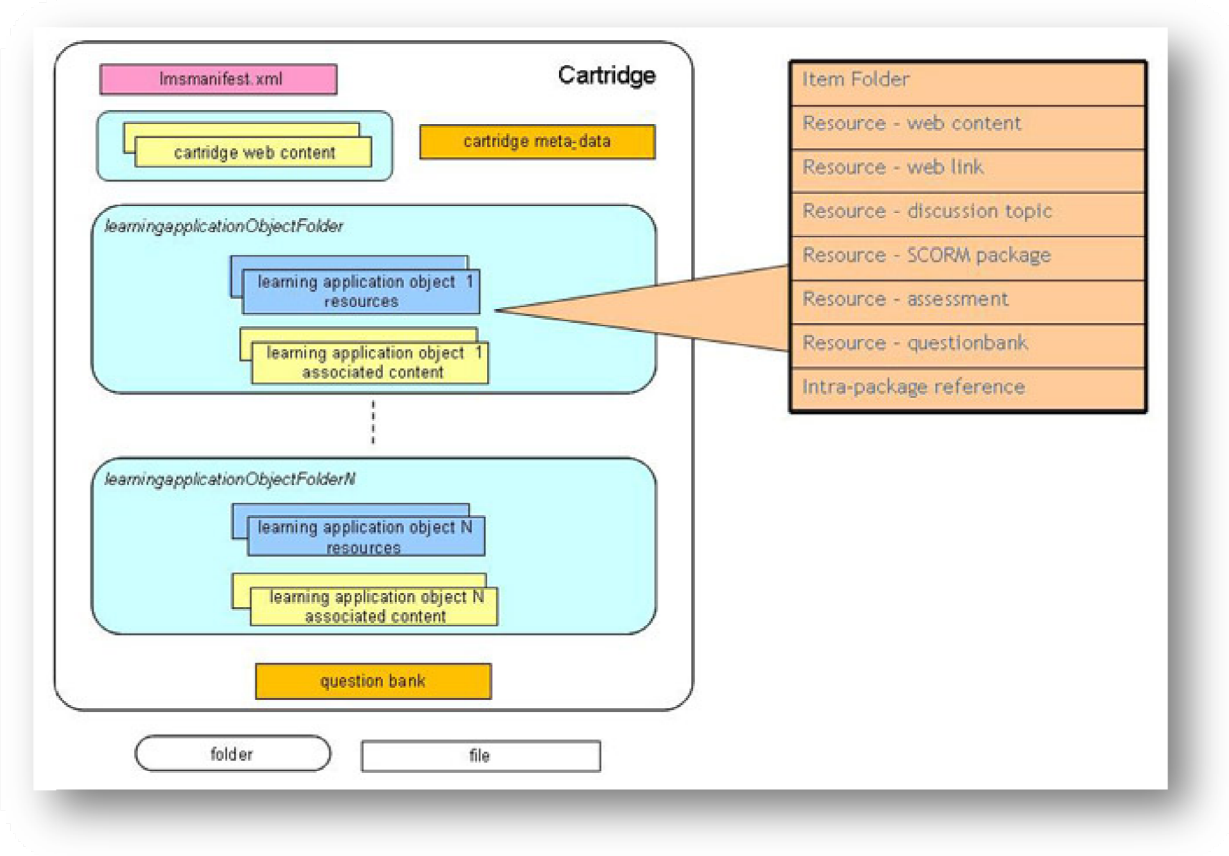


Figura 2.21 Estructura de un paquete utilizando la especificación IMS Common Cartridge [IMS, 2008]

Según se muestra en la Figura 2.21 se puede ver:

- imsmanifest.xml: Se trata del fichero que almacena la metainformación del objeto. Como se puede ver se trata de un fichero con formato XML.
- Web Content Resources: Se trata de contenido como ficheros HTML, imágenes, sonidos, videos, etc. Generalmente este tipo de contenidos no requerirá procesamiento extra por parte de los LMS, pudiendo éste estar organizado en subdirectorios o referenciar a objetos de aprendizaje de la aplicación.

- Learning Application Object Resources: Aquí se encuentran los recursos en formato de objetos de aprendizaje. Estos recursos podrán ser paquetes SCORM, cuestionarios, contenido Web, enlaces a otros contenidos, etc. Generalmente se parsean en entrada y se transforman en estructuras de datos internas en el LMS.
- Question Bank: Son listados de preguntas de alguno de los siguientes tipos:
  - Respuesta múltiple.
  - Verdadero/Falso.
  - Ensayo
  - Rellenar huecos
  - Coincidencias de patrones

Por último se presenta en la Tabla 2.9 una comparación entre SCORM e IMS Common Cartridge.

Característica	SCORM	IMS Common Cartridge
Estándar de empaquetado	IMS Content Packaging	IMS Content Packaging
Estándar de metadatos	IEEE LOM	Dublin Core vía IEEE LOM
Estándar de secuenciación	IMS Simple Sequencing	La secuenciación no es un requerimiento, sin embargo asume IMS Learning Design e IMS Simple Sequencing
Estándar de rastreo	IEEE derivado de AICC	QTI y Learning Tools Interoperability
Estándar de integración con la Web 2.0 y otras herramientas de aprendizaje	Nada	IMS Learning Tools Interoperability
Estándar de autorización de contenido	Nada	IMS Authorization Web Service
Soporte para herramientas de colaboración (tipo foros)	Nada	IMS Forum Initation
Soporte para estándares de currículo	Nada	IMS Reusable Competencies IMS Vocabulary Description
Soporte para reportar salidas	Nada	IMS Learning Information Services IMS Learner Information Package IMS ePorfolio
Soporte para accesibilidad	Nada	IMS Access for All

Tabla 2.9 Comparativa entre SCORM e IMS Content Cartridge

IMS Learning Tools Interoperability

El grupo de trabajo que ha desarrollado estas herramientas de interoperabilidad para entornos de aprendizaje tiene como objetivo principal el mejorar la interacción entre los sistemas de gestión del aprendizaje y las herramientas del aprendizaje.

Las directrices de esta especificación han tenido un alcance limitado y están destinadas a lograr una conexión simple entre el LMS y cualquier herramienta relacionada con los entornos e-learning. LTI V2.0 ampliará las directrices originales para gestionar las herramientas de configuración, las de presentación, los problemas de accesibilidad y el acceso a los datos del LMS como calendarios, calificaciones o contenidos.

Si se produjera un amplio apoyo de los proveedores de herramientas y los proveedores de LMS, se obtendrían los siguientes beneficios en términos de costes, soporte y seguridad:

- Reducir los costes de desarrollo de los proveedores de herramientas para desarrollar integraciones personalizadas con muchos LMS.
- Reducir los gastos de soporte para las universidades, proveedores de LMS y proveedores de herramientas procedentes de una interfaz definida entre el LMS y las herramientas: de esta forma será más factible que desarrollar una nueva versión de un LMS por ejemplo.
- Proteger el LMS de herramientas de integración propietarias: con LTI, las características de las herramientas están expuestas a través del descriptor y ningún código se ejecutará en el LMS.

Su impacto se caracterizará por:

- Menor coste y mayor tiempo de salida al mercado para hacer una herramienta de interoperación con muchos CMS/VLE.
- Más facilidad y seguridad en los modelos de despliegue.
- Mejores prácticas.
- Permite el desarrollo de herramientas de aprendizaje comerciales: más herramientas y mayor facilidad de acceso.

IMS Learning Information Services

La última especificación que se encuentra dentro de lo que se conoce como *IMS Digital Learning Services* recibe el nombre de *IMS Learning Information Services*. Se trata de un estándar encargado de apoyar las interacciones y el intercambio de datos entre los sistemas de aprendizaje y los administradores, estudiantes o los sistemas de recursos, incluyendo el intercambio de listas, perfiles de estudiantes, objetivos de competencias de aprendizaje así como el reporte de salidas.

Esta especificación se conocía anteriormente con el nombre de *IMS Enterprise Services V1.0* y en su versión 2.0 pasó a conocerse con el nombre de *IMS Learning Information Services*. Ya se comentó al principio de este apartado las principales características de la especificación IMS Enterprise Services; a continuación se presenta en la Tabla 2.10 una comparación entre ambas:

V1 (IMS Enterprise Services)	V2 (IMS Learning Information Services)
Personas, grupo s y miembros solamente	Añade nuevas características a través de IMS Learner Information Profile
No se define un formato estándar de representación de los cursos	Añade modelos de información y servicios de gestión de cursos
No se define un formato estándar de representación de la jerarquía de los cursos	Los modelos de información de gestión de cursos soportan la jerarquía de cursos
No se define un formato estándar de representación de múltiples secciones de cursos dentro de otros cursos	Las asociaciones de secciones permiten a las secciones de cursos múltiples ser expresadas como conjuntos de asociaciones para LMS
No hay representación de reportes de menor nivel	Añade servicios de reportes separados para operar con reportes de más bajo nivel

Tabla 2.10 Comparación entre V1 y V2

2.3.6. IMS Resource List Interoperability

IMS elaboró en Julio de 2004 la norma IMS RLI (Resource List Interoperability [IMS, 2004b]). En términos generales, esta norma determina la forma óptima de organizar, describir e intercambiar listas de recursos de un curso, como lo son las bibliografías. La norma se basa en un servicio abstracto de comportamiento y en un modelo de datos que describe en términos generales un recurso a nivel de un ítem, una colección de estos recursos (una lista), y los comportamientos asociados con un servicio de administración de RL (Resource List) o RLM (Resource List Manager). Cabe resaltar que la especificación no interviene en cómo se almacenan los recursos, sólo interviene en la interoperabilidad entre sistemas con paquetes de datos. Por otra parte, considerando que nunca se llegará a un sistema de descripción de recursos bibliográficos único, se propone el mapeo de LOM hacia los sistemas de citas más comunes entre bibliotecas y publicaciones.

El modelo de datos comprende un conjunto mínimo de elementos para citar publicaciones impresas. Se apoya en los estándares existentes para especificar e intercambiar dichos metadatos:

- Para metadatos utiliza LOM, a través de IMS LRM, combinado con ISO 690-2 para referencias bibliográficas a documentos electrónicos.
- Como esquemas de localización propone la utilización de OpenURL [2009], DOI [2009] o PURL [2009].
- Para el empaquetamiento de las listas y su transferencia entre sistemas se utiliza IMS CP (Content Packaging).

La especificación define como destinatarios de esta especificación a estudiantes, diseñadores instruccionales, bibliotecarios y repositorios de contenidos, entre otros, que tendrán la posibilidad de construir, seleccionar, utilizar y transmitir listas de recursos.

IMS RLI e IMS DRI tienen grandes similitudes, ambos están enfocados en la manipulación de recursos y metadatos, tanto de colecciones de objetos como de información, así como de paquetes de objetos de aprendizaje.

En la Figura 2.22 se muestra el modelo de la arquitectura del servicio de Administración de Listas de Recursos, que hace uso de un protocolo de mensajes para el intercambio de datos entre los sistemas administradores de listas. El alcance de la

especificación se resalta en el centro, puede observarse que se refiere a la interoperabilidad entre los datos y el modelo de comportamientos en el que se incluyen las operaciones que el RLM puede ejecutar. Estas operaciones están basadas en un modelo conocido como CRUD (Create/Read/Update/Delete) y se detallan en la Tabla 2.11; también pueden ubicarse en la Figura 2.22 las operaciones centrales entre los servicios de IMS RLI.

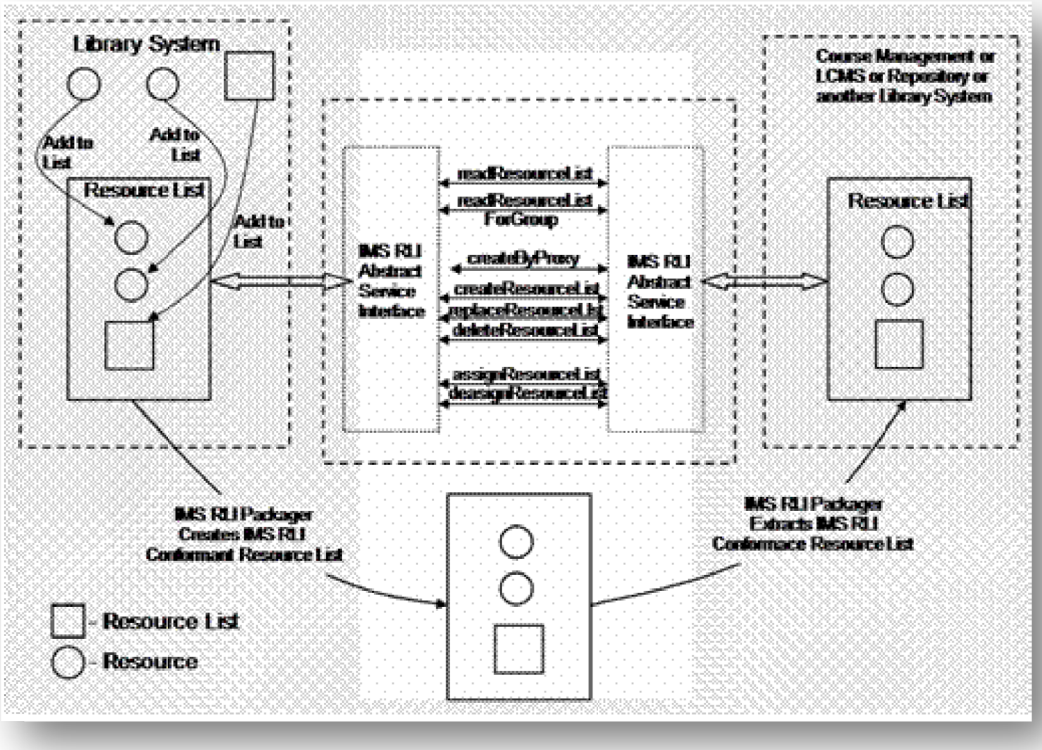


Figura 2.22 Arquitectura RLI [IMS, 2004b]

A continuación se muestra en la Tabla 2.11 el conjunto de métodos con los que cuenta esta especificación:

Operación	Descripción
createResourceList	Pide la creación de una lista de recursos en el sistema destino, donde el sistema origen es responsable de alojar el identificador de la lista de recursos.
createByProxyResourceList	Solicita la creación de una lista de recursos en el sistema destino, donde el sistema origen es responsable de alojar el identificador de la lista de recursos.
readResourceList	Lee todo el contenido de la lista de recursos identificada. El destino debe regresar todo el contenido que tenga de la lista identificada.
readResourceListforGroup	Lee todo el contenido de un conjunto de listas de recursos asociadas en un grupo.
replaceResourceList	Escribe nuevo contenido en el registro de una lista. La información contenida anteriormente se destruye.
deleteResourceList	Solicita el borrado de una lista. La lista y cualquier relación con grupos de listas quedan eliminadas.
assignResourceList	Solicita asociar una lista a un grupo de listas.
deassignResourceList	Solicita eliminar la asociación de una lista en un grupo.

Tabla 2.11 Operaciones de un Administrador de Listas de Recursos

2.3.7. IMS Vocabulary Definition Exchange

La especificación IMS VDEX (Vocabulary Definition Exchange [IMS, 2004c]) data de Febrero de 2004. Esta norma define un formato basado en XML para el intercambio de listas de valores de diferentes tipos, que son usadas como fuente de los vocabularios que se usan para etiquetar metadatos. En este sentido se consideran dos categorías distintas de vocabularios, diferenciadas por la clave usada para identificar un concepto:

- Vocabularios donde la clave es algún tipo de token, el cual referencia de forma efectiva algún término del lenguaje humano.
- Vocabularios donde la clave es un término del lenguaje humano.

Los tipos de datos usados en LOM y en la mayoría de las especificaciones IMS son términos tokenizados. Sin embargo también hace uso de términos del lenguaje humano, en el caso de las clasificaciones. Así la especificación soporta la descripción de las formas más extendidas de definir valores para etiquetar metadatos:

- La descripción de vocabularios/términos controlados que vienen expresados como pares de fuente-valor.
- La descripción de vocabularios jerárquicos o taxonomías.
- Tesauros.

Para cada uno de estos tipos de describir un vocabulario, existe un perfil de IMS VDEX concreto, según especifica [Sarasa et al., 2008].

2.3.8. IMS General Web Services

La base de los Servicios Web Generales [IMS, 2005] promueve la interoperabilidad de las especificaciones de los servicios Web basados en software y plataformas de diferentes vendedores. Se centra en la especificación de un conjunto de servicios Web y en los problemas más comunes encontrados en su implementación. Trata de dirigir la interoperabilidad en la capa de aplicación, en particular, la descripción de los comportamientos expuestos vía servicios Web.

Esta especificación puede ser extendida gracias a la adopción de uno o más perfiles, como los relacionados con 'Addressing' (transporte neutral del direccionamiento de los servicios Web), 'Attachment' (envío de documentos no XML a través de mensajería SOAP) o 'Security' (intercambio de datos dotados de mecanismos seguros).

En principio los enlaces basados en SOAP soportan gran cantidad de modelos de mensajería, ya que el modelo de información de la especificación IMS está definido independientemente de la naturaleza del mensaje (determinado en el enlazado descrito en el fichero WSDL), sin embargo actualmente sólo hay un modelo soportado, y este es el modelo síncrono, en el que el invocador del servicio se quedará bloqueado hasta recibir la respuesta del mismo.

El resto de modelos serán añadidos cuando las necesidades de las aplicaciones lo demanden. Hay tres métodos gracias a los cuales la capacidad funcional puede ser extendida:

- Añadir nuevos mensajes SOAP: El añadir nuevas transacciones de negocio y el uso de nuevos modelos de mensajería requerirá la creación de nuevos mensajes SOAP.
- Extensión de la cabecera SOAP: La actual especificación usa la cabecera SOAP para albergar la información del estado del intercambio de datos producido entre aplicaciones. Sería recomendable que se crearan extensiones propias para dicha cabecera SOAP.
- Extensiones de los datos contenidos en el cuerpo SOAP: El cuerpo SOAP contiene la instancia XML que será usada para representar los parámetros definidos por las operaciones de intercambio en la especificación. De igual forma se necesitaría añadir nuevos parámetros o extender la estructura XML de los actuales.

La especificación IMS GWS promueve la interoperabilidad a través de diferentes plataformas o sistemas software, gracias a la implementación basada en servicios. Según se puede ver en el modelo presentado en la Figura 2.23:

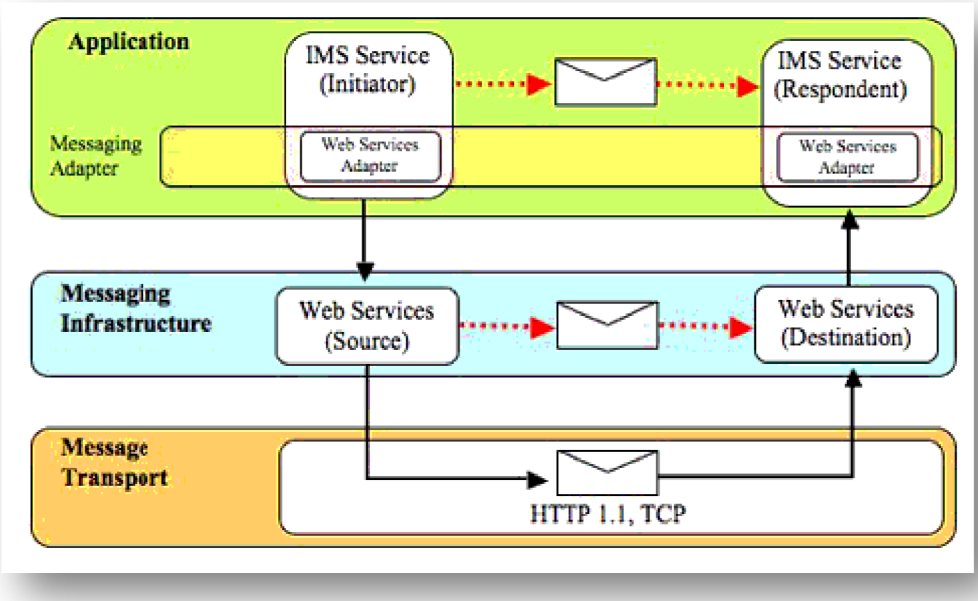


Figura 2.23 Modelo de intercambio de información en IMS GWS [IMS, 2005]

Dicha especificación no está centrada en generar una arquitectura "plug-and-play" para garantizar la completa interoperabilidad a través de servicios Web, ya que asume que con la interoperabilidad de los protocolos de los niveles inferiores sería suficiente. Como se puede ver en la imagen mostrada, los niveles inferiores son los primeros responsables de la infraestructura de intercambio de datos a través de mensajería. Una vez que el modelo del sistema de información define las reglas de enlazado, estas serán aplicadas a crear toda la coreografía de intercambio de mensajes. Una de las principales ventajas de utilizar estas reglas de enlazado, es que el adaptador de servicios necesario en ambas partes (emisor/receptor), tiene unas características similares, lo que permitirá generar un adaptador común de servicios Web o un framework, lo cual facilitará en gran medida el trabajo a desempeñar.

Uno de los principales perfiles que se pueden destacar es el denominado "GWS Attachments Profile", el cual extiende dicha especificación para soportar el intercambio de información no XML a través de mensajes SOAP, gracias a la cual permitirá

intercambiar ficheros doc, zip, pdf, etc. Esta información será incorporada al mensaje SOAP usando MTOM (*Message Transmission Optimization Mechanism*), el cual combina la eficiencia de transporte que utiliza SWA (SOAP con Attachment) y la flexibilidad de la codificación Base64. MTOM usa el mecanismo XOP (XML-binary Optimization Packaging) para de una manera eficiente colocar contenido no XML dentro de paquetes MIME.

2.4. ARQUITECTURAS

En este apartado se presentan las principales arquitecturas que se han diseñado hasta la fecha, las cuales sirven de base para el desarrollo de los principales sistemas e-learning que serán detallados en el siguiente capítulo.

IEEE [2001] ha desarrollado un estándar de definición de arquitecturas de e-learning con los siguientes objetivos:

- Proporcionar un marco general que ayude a entender los existentes y futuros sistemas.
- Promover la interoperabilidad y portabilidad identificando las interfaces críticas.
- Incorporar un horizonte tecnológico de, al menos, 5 o 10 años adaptable a nuevas tecnologías.

El estudio del actual campo de las aplicaciones de e-learning supone un reto por estar en constante crecimiento, por la cantidad de posibilidades que debe ofrecer y por su necesidad de flexibilidad y adaptación a nuevas herramientas (Figura 2.24).

Figura 2.24 Nuevas herramientas de e-learning

En todo caso se ha constatado que casi todos los sistemas e-learning deben estar soportados por una arquitectura que cumpla con las siguientes características (sintetizadas ya por CISCO en [Cisco, 2001]):

- **Abierta**. Debe soportar la interoperabilidad entre distintos proveedores de soluciones y estar basada en los estándares de organizaciones como AICC, IMS, ADL e IEEE.
- **Escalable**. Sus funciones deben poder ser ampliadas cuando sea necesario.
- **Global**. Debe poder ser utilizada en cualquier lugar del mundo y en cualquier momento con igual facilidad.
- **Integrada**. Debe integrarse con distintas infraestructuras de red y otras aplicaciones de seguridad, recursos humanos, etc.
- **Flexible**. Debe poder adaptarse a nuevos requisitos y procesos, nuevas tecnologías y nuevos proveedores de soluciones.
- **Adaptable**. En un mundo en constante desarrollo, debe ser de rápida y fácil implantación en organismos, empresas y entidades educativas.

Otros autores o entidades añaden distintas justificaciones para el diseño de arquitecturas e-learning. Por ejemplo, Rosenberg [2001] se basa en la necesidad de flexibilidad y adaptabilidad de estos sistemas a los cambios del entorno, de las tecnologías y de los contenidos; la empresa de creación de contenidos SkillSoft [2009], se basa en la necesidad de integrar tecnología, contenidos y servicios; por su parte, Microsoft, ya en 1999, se basaba en la necesidad de obtener sistemas integrados, controlables, fáciles de usar, adaptables a las nuevas tecnologías y escalables en función de las necesidades de crecimiento [Microsoft, 2009].

Se presentan a continuación las arquitecturas más relevantes que se han desarrollado en los últimos años; de forma sintética se describirán únicamente sus principales características.

2.4.1. Arquitectura LTSA de IEEE

IEEE ha realizado un gran esfuerzo para crear un estándar para el desarrollo de arquitecturas de sistemas de e-learning. Su última versión es la IEEE P1484.1/D9 [IEEE, 2001], en este documento establece una arquitectura de alto nivel para sistemas e-learning y sus componentes llamada LTSA (*Learning Technology Systems Architecture*).

En su arquitectura, IEEE especifica cinco niveles, aunque solo establece como obligatorio en este estándar el nivel 3 (*System Components*). En la Figura 2.25 se pueden ver gráficamente estos niveles.

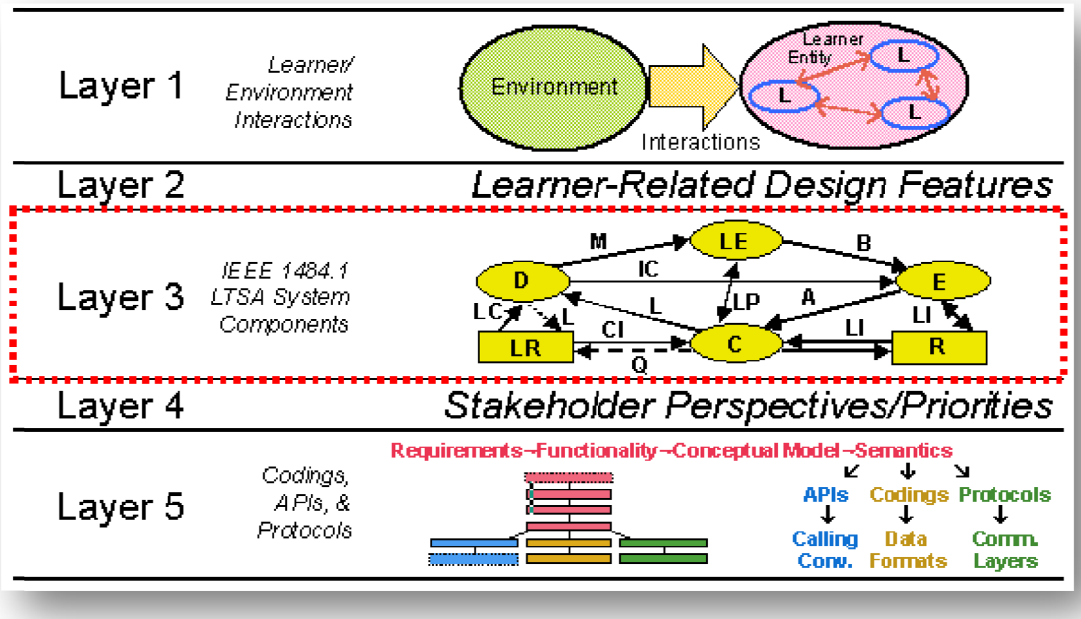


Figura 2.25 Arquitectura LTSA de IEEE

- **Nivel 1. Learner and Environment Interactions.** Es el nivel más genérico: el alumno tiene conocimientos nuevos o diferentes después de una experiencia educativa. Hay una interrelación entre el alumno y el sistema e-learning. Se refiere a la adquisición, transferencia, intercambio, descubrimiento, etc., de conocimientos o información a través de la interacción con el entorno.
- **Nivel 2. Learner-Related Design Features.** Relacionado con los efectos de la naturaleza humana del usuario sobre el diseño de los sistemas de aprendizaje informáticos.

- **Nivel 3. System Components.** Describe los componentes básicos de la arquitectura.
- **Nivel 4. Implementation Perspectives and Priorities.** Describe cómo una gran variedad de organismos, industrias, empresas y entes educativos analizan los sistemas de e-learning, en cuanto a: verificación y validación de los principales componentes; importancia de cada componente en función de las distintas perspectivas y prioridad entre los niveles.
- **Nivel 5. Operational Components and Interoperability.** Proporciona una visión global de cómo componentes e interfaces que permiten la interoperabilidad (codificación, APIs y protocolos) puede estar relacionados con la arquitectura de sistemas de e-learning.

El desarrollo del estándar se centra en el único nivel normativo existente: el nivel tres representado en la Figura 2.26.

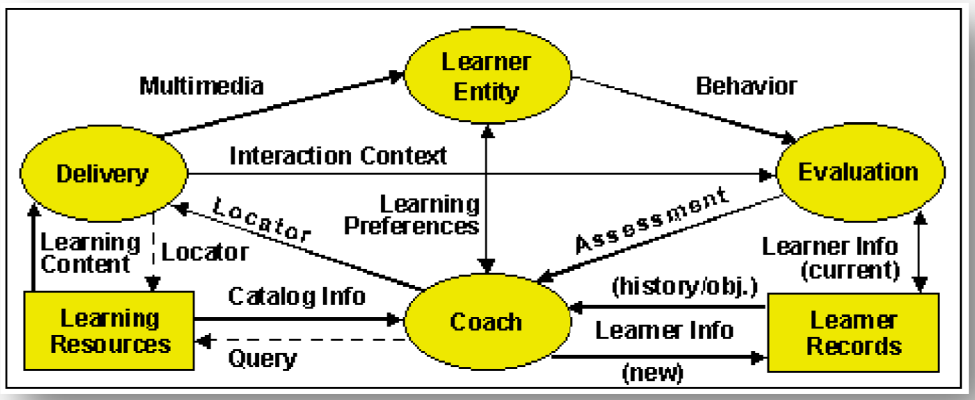


Figura 2.26 Nivel 3- System Components

El Sistema de Componentes del LTSA identifica los interfaces críticos de interoperabilidad para sistemas de e-learning. El propósito de este estándar es describir y entender los componentes generales identificando funciones generales. Las implementaciones reales de sistemas e-learning pueden no corresponder componente a componente con este estándar, pero conceptualmente deben realizar las mismas funciones. Por ejemplo, por motivos comerciales, las funciones de evaluación, entrega y entrenamiento pueden agruparse en la misma herramienta aunque conceptualmente sean componentes separados.

En la Figura 2.26 se pueden ver tres tipos de componentes:

- **Procesos** (con forma de elipse): Alumno, Evaluación, Entrenamiento y Entrega.
- **Almacenajes** (rectángulos): Registro de Alumnos y Catálogo de Recursos de Aprendizaje.
- **Flujos** (flechas): Preferencias de Aprendizaje, Comportamiento, Información de Conocimientos, Información del Alumno, Consulta, Información del Catálogo, Localizador, Contenidos, Multimedia y Contexto de Interacción.

De forma resumida, la operativa de los sistemas que implementan esta arquitectura es la siguiente:

- Los estilos, estrategias y métodos de aprendizaje los definen los alumnos mediante preferencias de aprendizaje.
- Los alumnos son observados y evaluados a través de sus interacciones con el sistema.
- La evaluación produce información sobre el alumno y su conocimiento.
- La información sobre el alumno se almacena en la base de datos histórica de alumnos.
- El proceso de entrenamiento revisa la información sobre el alumno y su conocimiento para establecer objetivos de aprendizaje futuros.
- El proceso de entrenamiento busca los recursos de aprendizaje necesarios para cada contenido.
- El proceso de entrenamiento extrae los localizadores del catálogo de recursos de aprendizaje y los pasa al proceso de entrega.
- El proceso de entrega extrae los contenidos de aprendizaje y los transforma en una presentación interactiva multimedia.

Cada componente se detalla a continuación. Los procesos se describen definiendo entradas, funcionalidad y salidas. Los almacenes se describen en función de los datos que contienen y los métodos de almacenamiento, búsqueda y actualización. Los flujos se describen mediante el tipo de información que fluye y el tipo de conectividad (unidireccional o bidireccional, estática o dinámica).

- **Alumno:** Entidad que representa al alumno. Puede ser una persona, un grupo de personas vistas como alumnos individuales o un grupo de personas vistas

como alumnos que trabajan de forma colaborativa. Sus relaciones con el resto de entidades son:

- Entrada: recibe las presentaciones multimedia.
- Salida: da datos sobre su comportamiento.
- Entrada/Salida: define, junto con el Proceso de Entrenamiento, sus preferencias de aprendizaje.

- **Evaluación:** Proceso con el que se mide al Alumno. Sus relaciones con el resto de entidades son:
  - Entrada: recibe el conocimiento observable del Alumno y el contexto de interacción con la Entrega.
  - Salida: envía información sobre el conocimiento del Alumno al Proceso de Entrenamiento.
  - Entrada/Salida: se puede recuperar y almacenar información referida a la situación actual del Alumno en el registro de alumnos.
- **Entrenamiento:** Proceso en el que se buscan y seleccionan los contenidos de aprendizaje que deben entregarse en función de datos recibidos sobre el alumno y los recursos de aprendizaje disponibles. Sus relaciones con el resto de entidades son:
  - Entrada: recibe los datos sobre el conocimiento del Alumno obtenidos del proceso de Evaluación, los datos pasados, presentes y futuros del Alumno del registro de alumnos y la información resultado de sus búsquedas en el catálogo de recursos de aprendizaje.
  - Salida: envía datos sobre el Alumno al registro de Alumnos, envía al catálogo de recursos de aprendizaje las búsquedas con las que encontrar los recursos apropiados y envía el resultado de estas búsquedas, en forma de localizadores, al proceso de Entrega.
  - Entrada/Salida: recibe y negocia las preferencias de aprendizaje del Alumno.
- **Entrega:** Proceso en el que se transforma la información contenida en los recursos de aprendizaje que el proceso de Entrenamiento ha decidido utilizar. Esta transformación produce los contenidos educativos que se presentan al Alumno, que pueden estar en formatos muy variados: presentaciones y preguntas, sistemas inteligentes de tutorización, videoconferencias, modelos conceptuales, etc. Sus relaciones con el resto de entidades son:
  - Entrada: recibe, desde el proceso de Entrenamiento, localizadores que deben ser recuperados y los recursos de aprendizaje correspondientes desde el catalogo.
  - Salida: envía localizadores al catálogo para recuperar recursos de aprendizaje, presenta los contenidos relativos a estos recursos en

forma multimedia al Alumno y envía al proceso de Evaluación el contexto de interacción.

- **Registro de Alumnos:** Constituye la base de datos de Alumnos, con datos pasados, presentes o futuros sobre los objetivos, logros, actividades, etc. del alumno. Los procesos de Evaluación y de Entrenamiento pueden recuperar y almacenar datos en esta base de datos.
- **Catalogo de Recursos de Aprendizaje:** Constituye la base de datos de recursos de aprendizaje; incluye presentaciones, tutorías, herramientas, experimentos de laboratorio y cualquier otro tipo de material educativo. El proceso de Entrenamiento puede hacer búsquedas y recuperar información sobre lo buscado en esta base de datos. El proceso de Entrega, a través de los localizadores obtenidos por el proceso de Entrenamiento, puede obtener los contenidos que desee del catálogo de recursos de aprendizaje.
- **Preferencias de Aprendizaje:** Flujo bidireccional entre el Alumno y el proceso de Entrenamiento, en el que se incluyen datos relativos a las preferencias culturales del alumno y sus limitaciones físicas o mentales. En estos datos también pueden influir entes externos con autoridad, como padres, profesores, entes educativos, etc.
- **Comportamiento:** Flujo desde el Alumno al proceso de Evaluación en el que se refieren todas las actividades que ha realizado el alumno: respuestas escritas (o directamente con voz), opciones seleccionadas, etc.
- **Información de Conocimientos:** Flujo de información que el proceso de Entrenamiento recibe del proceso de Evaluación con datos sobre la situación actual del alumno.
- **Información del Alumno:** Flujo bidireccional (recuperación de datos o almacenamiento de datos) entre el registro de alumnos y los procesos de Evaluación o Entrenamiento. En este flujo de información se pasan datos pasados, presentes o futuros sobre los objetivos, logros, actividades, preferencias, etc. del alumno.
- **Consulta:** Flujo desde el proceso de Entrenamiento hacia el Catálogo de recursos de aprendizaje; contiene los criterios de búsqueda necesarios para poder localizar los recursos de aprendizaje que necesita.
- **Información del Catalogo:** Flujo que el proceso de Entrenamiento obtiene del catálogo de recursos de aprendizaje como resultado de sus búsquedas en el catálogo; contiene los datos que describen los recursos de aprendizaje obtenidos (metadatos) y sus localizadores (identificativos que apuntan al recurso).
- **Localizador:** Flujo de información desde el proceso de Entrenamiento hacia el proceso de Entrega (pidiéndole que ponga a disposición el contenido

correspondiente) o desde el proceso de Entrega al catálogo de recursos de aprendizaje (solicitándole que le envíe el recurso de aprendizaje que identifica el localizador). La información que contiene es un identificativo o enlace a un recurso de aprendizaje, por ejemplo una dirección URL, etc.

- **Contenidos:** Flujo de información desde el catálogo de recursos de aprendizaje al proceso de Entrega, en el que se envía el recurso solicitado.
- **Multimedia:** Flujo de información desde el proceso de Entrega al Alumno, en el que se le presentan los contenidos educativos en diversas formas: gráfico, texto, video, audio, etc.
- **Contexto De Interacción:** Flujo de información desde el proceso de Entrega hasta el proceso de Evaluación, que proporciona la información necesaria al proceso de Evaluación para interpretar los datos de comportamiento. Esta información incluye datos sobre el entorno de aprendizaje que ha utilizado el Alumno.

2.4.2. Arquitectura de ARIADNE

Las herramientas que facilita el proyecto ARIADNE [2009] para el desarrollo de aplicaciones de aprendizaje se sostienen sobre la siguiente estructura (Figura 2.27).

Los elementos más importantes de la arquitectura son:

- **Gestor de KPS**: Forma parte de la capa de datos, en la cual se almacenan los LO ("*Learning Objects*", objetos de aprendizaje) y los Metadatos que los describen.
- **SILO** (*Search and Index Learning Objects*): Es una herramienta con la que se busca ofrecer una forma simple de acceso al KPS.
- **WEBLE** (*Web-Based Learning Environment*): Proporciona herramientas para la gestión de los cursos y acceso al KPS.
- **KPS Client**: Es la herramienta de acceso al KPS.

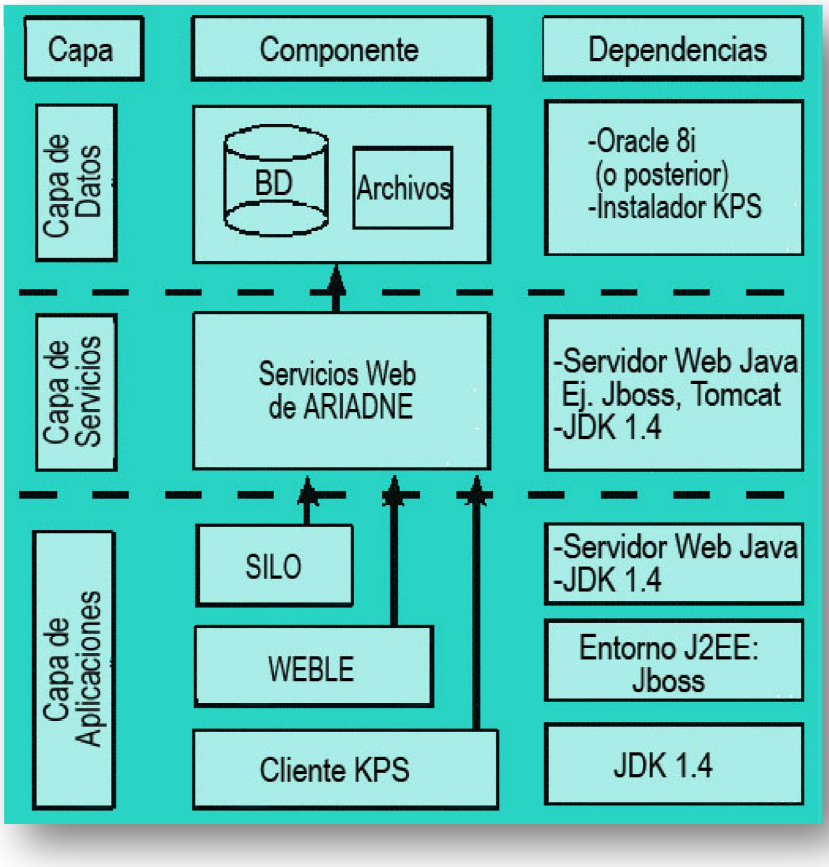


Figura 2.27 Arquitectura del sistema ARIADNE [2006]

En esta arquitectura se pueden observar dos tipos de dependencias diferentes:

- **Dependencias entre los componentes ARIADNE**: Estas dependencias se representan con flechas negras. Por ejemplo, las dependencias del componente SILO con los servicios Web. Sin embargo, esto no quiere decir que SILO no pueda instalarse sin haber instalado previamente los servicios Web. Actualmente existen múltiples nodos en la red ARIADNE que albergan estos servicios. Esto quiere decir (desde el punto de vista práctico) que, por ejemplo, se puede instalar un servidor Tomcat, instalar SILO y configurarlo para emplear los servicios Web que Tomcat proporciona.
- **Dependencias de los componentes ARIADNE y software de terceras partes**: El conjunto de recuadros de la parte derecha del esquema de la Figura 2.27 representan el conjunto de software necesario para instalar los componentes ARIADNE. El componente WEBLE, por ejemplo, necesita ser instalado en un servidor Web que soporte J2EE; SILO tan sólo necesita un servidor Web Java como puede ser Tomcat.

Se verá con más detalle sus características en el capítulo tres dedicado al planteamiento del problema.

2.4.3. Arquitectura de CISCO

Cisco ha desarrollado un modelo de arquitectura para sistemas e-learning, caracterizado por ser un sistema abierto, escalable, global, integrado y flexible [Cisco, 2001]. Su visión es mucho más comercial que la de IEEE, como corresponde a su carácter de empresa privada, y va más encaminada a una implementación concreta, la suya propia. En todo caso, se va a realizar una presentación breve por el interés evidente de su nivel intermedio.

La arquitectura completa se divide en tres niveles tal como se ve en la Figura 2.28.

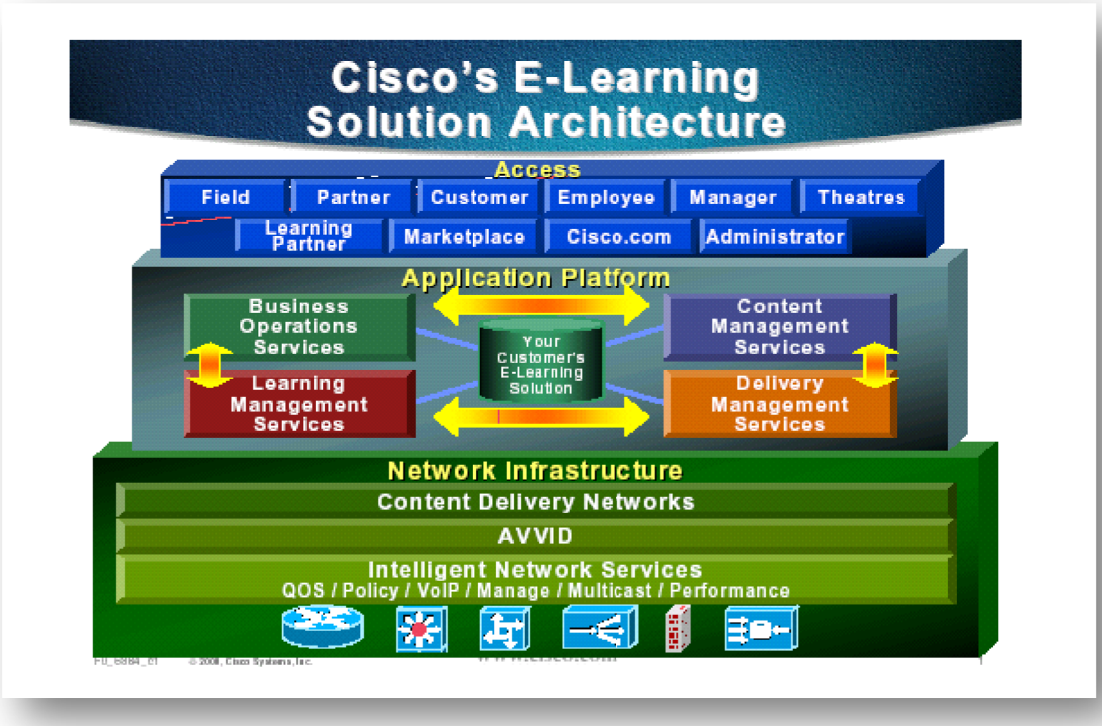


Figura 2.28 Arquitectura de Cisco Systems [CISCO, 2001]

Cada uno de los niveles tiene la siguiente funcionalidad:

- **Nivel 1. The Underlying Network Infrastructure:** Este es el último nivel o nivel de red en el que se apoya el nivel intermedio. Según [Cisco, 2000], la arquitectura de red propuesta tiene por objeto proporcionar disponibilidad, seguridad y buenos resultados en tiempo y calidad de acceso; integrando tecnologías de:

- IP Multicasting, para transmitir datos multimedia desde distintos servidores a muchos clientes diferentes a la vez y en cualquier momento.
- Quality of Service (QoS), para proteger la transmisión de datos crítico mientras se realizan tareas de e-learning muy costosas en cuanto a recursos de red, como transmisión de videos, clases virtuales, etc.
- Seguridad, para proteger los datos y aplicaciones de accesos indebidos: gestión de *password*, encriptación de datos a transmitir, utilización masiva de recursos y tecnología VPN (*Virtual Private Network*) que gestiona la seguridad de redas privadas dentro de la red general.
- Mecanismos de entrega y distribución de contenidos:
  - CDN (*Content Delivery Network*), que utilizando la tecnología de red SODA (*Self-Organizing Distributed Architecture*) permite a los usuarios recibir localmente datos procedentes de servidores de intranet y extranet optimizando los accesos.
  - CDM (*Content Distribution Manager*), que maneja la distribución de los contenidos seleccionando los servidores por proximidad física y disponibilidad y según las restricciones establecidas por el administrador de la red.

Es un nivel excesivamente cercano a las infraestructuras físicas y que carece de interés a los efectos de esta tesis, si bien la importancia de sus propuestas es más que patente en entornos que dependen en gran medida del correcto funcionamiento de la tecnología.

- **Nivel 2. The Application Blueprint:** Este es el nivel intermedio que realiza la gestión propia de e-learning: contenidos, estudiantes, cursos, etc. Es un modelo modular donde las tareas y responsabilidades se dividen en áreas funcionales relacionadas e integradas. Según se desarrolla en [Cisco, 2001], estos módulos son los siguientes:
  - **Operaciones de Negocio-BOS** (*Business Operations Services*). Está formado por un conjunto de herramientas y aplicaciones integradas que soportan los siguientes servicios:
    - Gestión de la efectividad del aprendizaje y de la experiencia del alumno.
    - Soporte a las consultas *on-line*, las llamadas gratuitas de apoyo y la distribución por e-mail.
    - Servicio de ayuda *on-line* en las aplicaciones.
    - Gestión de la seguridad de acceso aplicaciones y contenidos.
    - Generación de informes de progreso, utilización y resultados.

- Análisis de necesidades: *Gap Analysis* mide la diferencia entre los conocimientos necesarios de un estudiante según sus competencias y sus conocimientos reales; y *Cost Model Analysis* que, extrapolando datos de los resultados anteriores, determina la clase de contenidos que deben ser creados en otros módulos.

- **Gestión de Contenidos-CMS** (*Content Management Service*). Permite verificar, registrar utilizando metadatos, ensamblar, manejar y publicar contenidos. Está formado por los siguientes submódulos:
  - *Workflow Application*. Aplicación de flujo de datos de gestión de contenidos, en la que es posible personalizar el flujo de datos según el grupo de trabajo que la utilice.
  - *Authoring Tool Integration Service*. Permite que distintos tipos de contenidos: texto, test, gráficos, etc. se integren en cualquier nivel jerárquico de la estructura de contenidos.
  - *Registry Services*. Almacena la ubicación, los metadatos descriptivos y la estructura de metadatos asociada de cada contenido. Físicamente pueden almacenarse en el *Content Storage System* o en otra ubicación segura.
  - *Object Mining Service*. Localiza contenidos mediante distintos criterios de búsqueda. Aplicación muy útil para poder aplicar reusabilidad.
  - *Assembler*. Permite, utilizando los resultados del módulo anterior, crear plantillas que son registradas y almacenadas. Permite ensamblar contenidos utilizando estas plantillas como base.
  - *Content Storage Services*. Proporciona la gestión de almacenamiento físico de los contenidos: manejo de versiones, bloqueos, histórico, informes, etc.
  - *Publishing Services*. Cuando un curso está listo, proporciona los datos para su puesta a disposición que hará el módulo DMS.
- **Gestión de la Demanda-DMS** (*Demand Management Services*). Se encarga de poner a disposición de los alumnos los cursos terminados. Está formado por un conjunto de herramientas y aplicaciones integradas que soportan los siguientes servicios:
  - Presentar al alumno los contenidos de un curso en función de una parametrización personalizada.
  - Manejar la distribución de contenidos en función de las reglas establecidas por el administrador.

- Controlar los contenidos que se distribuyen y en qué forma. Estos datos se pasan al módulo LMS que es el que los utiliza para futuras peticiones e histórico.

- **Gestión del Aprendizaje-LMS** (*Learning Management Services*). Es el módulo al que acceden los alumnos y que lleva el control de sus acciones. Está formado por los siguientes submódulos:
  - *Personalization*. Crea planes de enseñanza personalizados en función del perfil del usuario y de las preferencias personales.
  - *Search/Browse*. Permite realizar búsquedas y consultas en el catálogo de cursos.
  - *Registration*. En unión al módulo *Search/Browse* permite registrarse en un curso y gestionar las listas de espera, notificaciones, cambios de programación, etc.
  - *Learner Tracking*. Contiene la historia del aprendizaje de cada alumno: cursos realizados y situación en los cursos actuales. Estos datos le permiten proponer cursos futuros en función del perfil de cada alumno.
  - *E-Commerce*. Junto con el módulo *Search/Browse* permite realizar el pago de los cursos proporcionando varios métodos de pago y gestión de datos financieros.
  - *Assessment*. Analiza para cada estudiante su *pre-assessment* (diferencia entre lo que sabe y lo que debería saber para una determinada competencia) y su *post-assessment* (confirmación de su conocimiento real). Con estos datos personaliza el material de estudio e informa comprensivamente del currículum del alumno, para seguimiento personal o de la comunidad educativa.
  - *Manager's Toolkit*. Maneja la función de gestor educativo que puede aprobar o denegar altas en cursos, añadir o quitar alumnos, seguir el progreso del aprendizaje, etc.
  - *Survey*. Recoge datos sobre la satisfacción de los usuarios para enviárselos al módulo BOS.
  - *Resource Management*. Programa y asigna la utilización de recursos: equipos informáticos, profesores, clases, eventos virtuales, etc.

- **Nivel 3. The Access Layer:** Este es el nivel más alto o nivel de acceso a la aplicación, donde el tipo de persona o entidad que accede puede ser muy variado. Este nivel proporciona la posibilidad de definir diferentes portales de acceso personalizados.

2.5. ARQUITECTURAS ORIENTADAS A SERVICIOS

Los sistemas de información son cada vez más complejos y se necesitan sistemas más exigentes que requieren cooperación; y cada vez son menos comunes las soluciones centralizadas y aparecen nuevos métodos de interacción (comunicaciones). El objetivo de las organizaciones es reducir los costes y maximizar la utilización de la tecnología existente; al mismo tiempo las empresas intentan ser más competitivas y avanzar en sus prioridades estratégicas.

Existen dos líneas para conseguir estos propósitos: la heterogeneidad y el cambio. Muchas empresas tienen diferentes sistemas, aplicaciones y arquitecturas de diferentes tecnologías y algunas bastante antiguas. Integrar productos de diferentes proveedores y de diferentes tecnologías es una ardua tarea, por eso cada vez más las aplicaciones están orientadas al servicio. Esto se puede ver en los gráficos del informe del CBDI Forum [CBDI, 2006b], que se muestra en la Figura 2.29.

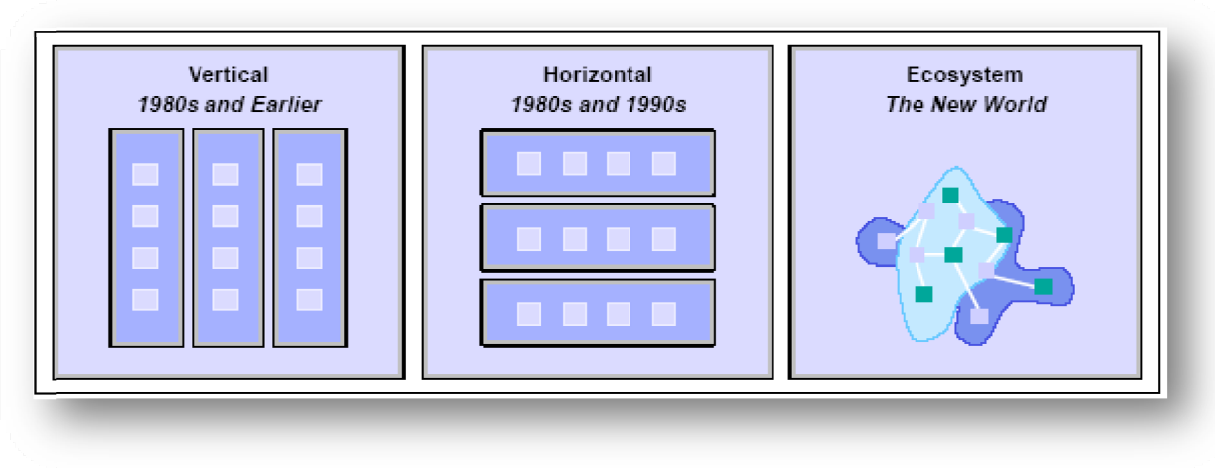


Figura 2.29 Evolución de los procesos de negocio

Por otra parte, en la Figura 2.30 se puede apreciar la evolución hacia las arquitecturas orientadas a servicio.

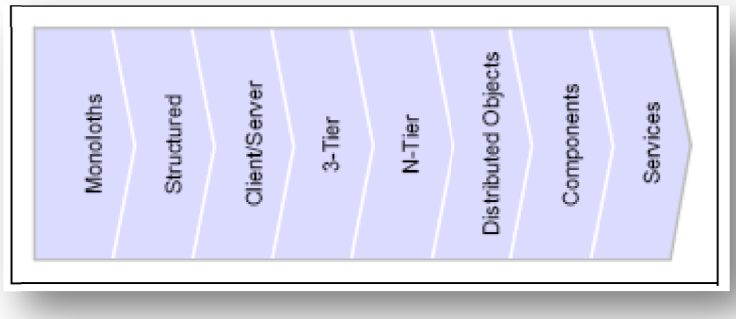


Figura 2.30 Evolución de las arquitecturas de sistemas software

Durante años la industria de la tecnología ha estado luchando para solucionar problemas como la heterogeneidad, la interoperabilidad (para que la solución escogida o la aplicación construida sea independiente de la plataforma y tecnología utilizada), incluso en la forma de recoger los requisitos, o que las aplicaciones sean independientes del protocolo de comunicación. Han ido apareciendo distintas formas de plantear la construcción de aplicaciones, sin lugar a duda una de las más importantes para resolver los problemas anteriores ha sido el enfoque de la orientación a objetos, aplicado tanto al análisis, diseño, como programación de aplicaciones, planteando el desarrollo de software como "un problema de dominio y una solución lógica desde la perspectiva de los objetos (cosas, conceptos o entidades)" [Larman, 2001]. Jacobson et al. [1992] definen estos objetos como "características con un número de operaciones y un estado que recuerda los efectos de las operaciones sobre ellos".

En el análisis orientado a objetos, tales objetos son identificados y descritos en el dominio del problema; y en el diseño orientado a objetos son transformados en objetos de software que serán implementados en un lenguaje orientado a objetos. De esta forma se reduce el esfuerzo de análisis y diseño de escenarios complejos y se facilita la reutilización.

A partir de la orientación a objetos han ido apareciendo otros "paradigmas" como la orientación a aspectos [POA, 2009], o la orientación a servicios; un servicio es generalmente implementado como una entidad software que puede ser accedida como una única instancia e interactúa con otras aplicaciones y otros servicios a través de un modelo de comunicación basado en mensajes. Antes de proceder a la descripción del concepto de arquitectura orientada a servicio, conviene aportar algunas de las definiciones que se podrían realizar del término servicio Web:

- Una aplicación modular que se autodescribe, que puede ser publicada, localizada, invocada o usada desde cualquier parte de la Web y que está basada en estándares abiertos como XML, UDDI, SOAP o WSDL. [Newcomer, 2002].
- Una aplicación accesible a otras aplicaciones a través de la Web. [Pelechano, 2005].
- Un sistema software identificado por una URI (*Uniform Resource Identifier*), cuyos interfaces públicos y enlaces se definen y describen utilizando XML. Su definición puede ser descubierta por otros sistemas software. Estos sistemas pueden interactuar con el servicio Web de la forma prescrita por su definición, usando mensajes basados en XML a través de protocolos y estándares de Internet [W3C, 2004a] [W3C, 2004b].
- Una interfaz que describe un conjunto de operaciones, accesibles a través de la red mediante mensajería XML estándar. [IBM, 2008]
- Cualquier aplicación accesible por HTTP/HTTPS, con la que se puede interactuar usando mensajes SOAP, se registra en un registro UDDI y tiene una descripción WSDL [HP, 2009].
- Un proveedor de información o capacidades expuestas en una red a través de interfaces, protocolos consistentes y estándares [MS, 2009].

En cuanto a la terminología referente a los servicios Web, es necesario conocer:

- **Servicios:** Entidades lógicas con una funcionalidad definida y las reglas establecidas por una o más interfaces que son publicadas. Los servicios Web se proponen como una alternativa para facilitar la intercomunicación entre diferentes arquitecturas de componentes, ofreciendo una visión de dichas arquitecturas, basada en servicios, totalmente compatible con Internet.
- **Proveedor de servicio (*Service provider*):** La entidad de software que implementa la especificación del servicio.
- **Consumidor del servicio (*Service consumer (o requestor)*):** La entidad de software que llama al proveedor de servicio. Tradicionalmente llamado "cliente". Un consumidor de servicio puede ser una aplicación u otro servicio.
- **Servicio localizador (*Service locator*):** Un tipo específico de proveedor de servicio que actúa como un registro y permite la búsqueda de interfaces y localizaciones de servicios.
- **Servicio intermediario (*Service broker*):** Un tipo específico de proveedor de servicio que puede pasar las peticiones de un servicio a uno o más proveedores de servicios adicionales.

Existen muchas definiciones de arquitectura, y también existen muchas interpretaciones de estas definiciones. La arquitectura de un sistema, en general, describe su estructura, a través de los siguientes aspectos:

- Los componentes que intervienen como bloques básicos del sistema.
- Conectores que describen los mecanismos de comunicación con otros sistemas y los mecanismos de interconexión entre los propios componentes de la arquitectura.
- Los flujos que muestran como una aplicación utiliza los componentes y los conectores para llevar a cabo su objetivo.

Se utiliza el acrónimo SOA para hacer referencia a la arquitectura orientada a servicio, el término SOA son las siglas inglesas de *Service Oriented Architecture*.

La Arquitectura Orientada a Servicios presenta una ventaja para la construcción de sistemas distribuidos, ya que contempla, la funcionalidad de las aplicaciones como servicios que pueden utilizar otras aplicaciones y otros servicios.

Una Arquitectura Orientada a Servicio está compuesta por elementos funcionales y elementos relacionados con la calidad de servicio, como se puede ver en la Figura 2.31 que se describen a continuación.

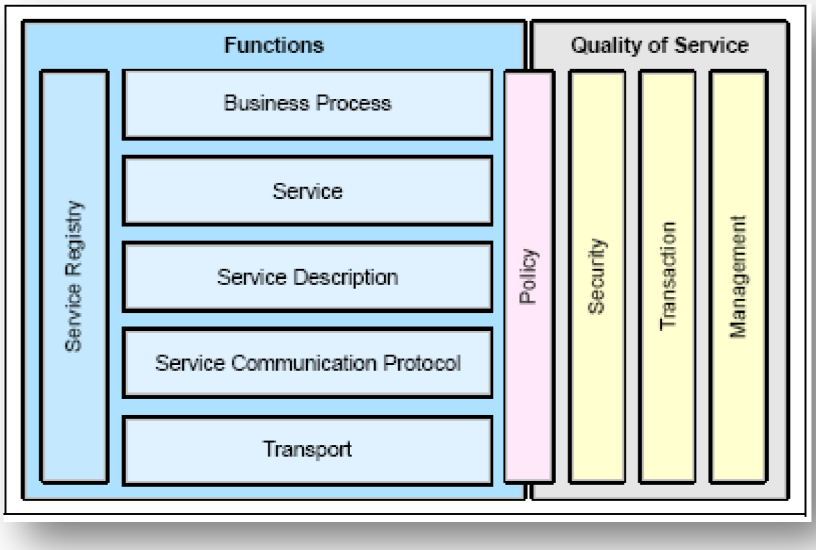


Figura 2.31 Elementos de una Arquitectura Orientada a Servicio

- **Aspectos funcionales:**
    - **Transporte:** Es el mecanismo usado para llevar peticiones de servicios desde el consumidor al proveedor del servicio, y las respuestas desde el proveedor del servicio al consumidor del servicio.
    - **Protocolo de comunicación del servicio:** Es el mecanismo de comunicación establecido entre el proveedor del servicio y el consumidor del servicio.
    - **Descripción del servicio:** Es el esquema establecido para describir qué es el servicio, cómo debe invocarse y que datos son requeridos para la invocación.
    - **Servicio:** Describe un servicio que está disponible para utilizarse.
    - **Proceso de negocio:** Es una colección de servicios, invocados de una manera particular, en una determinada secuencia y con unas reglas particulares para llevar a cabo la funcionalidad de negocio requerida. Un proceso de negocio puede estar compuesto por servicios de diferente naturaleza e incluso en distintas localizaciones.
    - **Registro de Servicio:** Es el repositorio de servicios y las descripciones que son usados por los proveedores de servicio para publicarlos, y para que los consumidores del servicio puedan invocarlos. El registro del servicio puede aportar la funcionalidad a los servicios que necesiten un repositorio centralizado.
- **Aspectos de la calidad del servicio:**
    - **Política:** Es un conjunto de condiciones o reglas sobre las cuales un proveedor del servicio hace un servicio disponible a los consumidores.
    - **Seguridad:** Es el conjunto de reglas que pueden ser aplicadas para la identificación, autorización y el control de acceso a los consumidores de servicios.
    - **Transacción:** Es el conjunto de atributos que pueden ser aplicados a un grupo de servicios para conseguir un resultado consistente. Por ejemplo si un grupo de tres servicios tienen que terminar para completar la función, todos tienen que estar completados y haber terminado su ejecución.
    - **Gestión:** Es el conjunto de atributos que pueden ser aplicados para manejar a los proveedores del servicio o a los consumidores.

En la Figura 2.32 se pueden ver los elementos de colaboración existentes en una Arquitectura Orientada a Servicio:

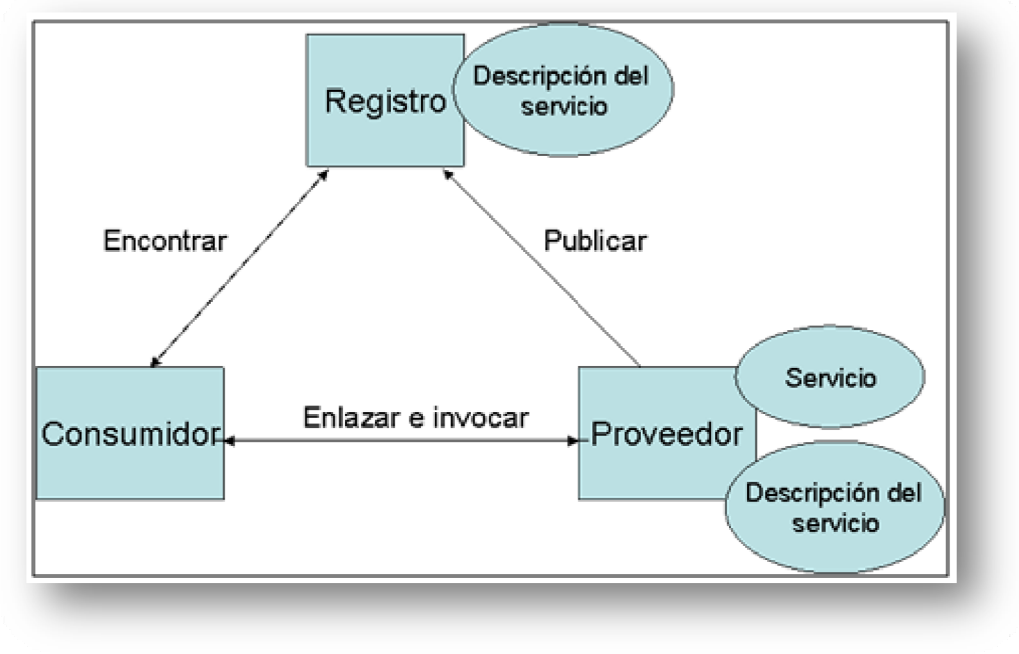


Figura 2.32 Elementos de colaboración en una Arquitectura SOA

Los roles de una Arquitectura Orientada a Servicio son:

- **Consumidor de servicio:** El consumidor de servicio es una aplicación, un módulo de software u otro servicio que requiere un servicio. Inicia la búsqueda en el registro de servicio, enlaza con el servicio a través del transporte y ejecuta la función del servicio de acuerdo con las reglas establecidas.
- **Proveedor de servicios:** El proveedor de servicios es una entidad que se puede acceder a través de la red y que acepta y ejecuta peticiones de los consumidores. Publica las interfaces de los servicios en el registro de servicios para que los consumidores puedan descubrirlos y puedan acceder a ellos.
- **Registro de servicios:** Un registro de servicios es el que permite que los servicios puedan ser descubiertos. Contiene un repositorio con los registros que están disponibles y permite la búsqueda de los proveedores de los servicios a través de las interfaces que han sido establecidas y que son de interés para los consumidores.

Las operaciones dentro de una arquitectura orientada a servicios son:

- **Publicación:** Para que los servicios puedan estar accesibles, un servicio tiene que tener una descripción, que debe ser publicada para que pueda ser descubierta e invocada por un consumidor de servicio.

- **Localización:** Un consumidor del servicio puede localizar un servicio realizando una búsqueda sobre el registro de servicios que cumpla algún criterio.
- **Enlazar e invocar:** Después de recoger la descripción del servicio, el consumidor del servicio puede invocar el servicio de acuerdo con la información de la propia descripción del servicio.

En la Figura 2.33 se puede ver más claramente el entorno y forma de colaboración:

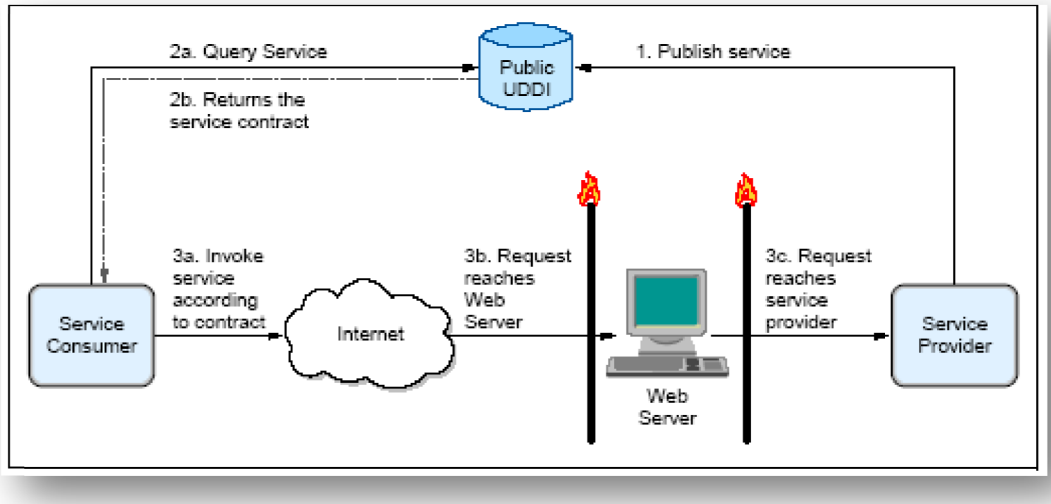


Figura 2.33 Colaboración en una Arquitectura SOA

Cómo ya se comentó en apartados anteriores, para poder realizar estas operaciones, existen distintas tecnologías; así, para realizar la descripción del servicio que especifica la forma de acceder al consumidor y poder interactuar con el proveedor se utiliza WSDL (*Web Services Description Lenguaje*) [W3C, 2001a], esta especificación establece el qué, el dónde y el cómo se accede a un determinado servicio. Y para poder hacer la localización, la integración y poder enlazar e invocar a un Servicio Web se utiliza UDDI (*Universal Description, Discovery and Integration*) [UDDI, 2009].

Para que el funcionamiento de una arquitectura orientada a servicio sea dinámica, tiene que cumplir las siguientes características [McGovern et al., 2001]:

- Los servicios tienen que ser modulares.

- Los servicios tienen que soportar la interoperabilidad.
- Los servicios tienen que tener la descripción perfectamente establecida.
- Los servicios tienen que ser transparentes a la localización.
- Los servicios tienen que ser independientes del lenguaje de implementación.
- Los servicios tienen que ser transparentes al protocolo de comunicación.
- Los servicios tienen que ser independientes del Sistema Operativo y del hardware utilizado.

Como ya se ha indicado, los procesos de negocio de una organización se van creando teniendo en cuenta la habilidad para cambiar rápidamente, la heterogeneidad y sobre todo la necesidad de reducción de costes. Para recordar la competitividad existente entre las distintas organizaciones, deben poder crear procesos de negocio y ofrecer servicios a sus clientes de forma que se puedan adaptar rápidamente a factores internos como adquisiciones, fusiones y reestructuraciones, o factores externos como son los requerimientos del mercado y de los clientes. Para las compañías es necesario llevar un equilibrio entre el coste, la efectividad, la calidad de sus procesos y la flexibilidad de la infraestructura tecnológica de que disponen.

Con una arquitectura orientada a servicio (SOA) se pueden alcanzar algunos beneficios para ayudar a las distintas organizaciones a conseguir unos procesos de negocios exitosos y dinámicos:

- **Solventar los problemas existentes:** Utilizando una arquitectura SOA se establece una capa de abstracción que permite a una organización continuar ofreciendo una innovación en tecnología, encapsulando los problemas en servicios que ofrecen como funciones de negocio. Las organizaciones pueden continuar obteniendo valor utilizando recursos externos a través de los servicios en lugar de invertir y reconstruir las soluciones existentes.
- **Fácil de integrar y gestionar la complejidad:** El punto de integración en una arquitectura SOA es la especificación del servicio y no la implementación. De esta forma aporta la transparencia de la implementación y minimiza el impacto cuando la infraestructura o la implementación existente cambia. Mediante la especificación del servicio con las funciones implementadas o los problemas que resuelve, incluso en distintos sistemas, la integración resulta sencilla.
- **Acelerar la puesta en producción de los sistemas (*time-to-market*):** La habilidad de componer nuevos servicios haciendo uso de los ya existentes

aporta una ventaja significativa a una compañía que necesita responder de una forma rápida a la demanda de negocio que le requieren. El ciclo de desarrollo se puede llegar a reducir, aunque también es importante saber establecer bien una base de requisitos funcionales y de pruebas.

- **Reducir costes y aumentar la reutilización:** Realizando una base de servicios que se pueden exponer y se pueden reutilizar, incluso si los servicios los ofrecen otras compañías, resultan mucho menos costosos los desarrollos, se necesita menos tiempo de codificación, se aumenta la reutilización en los desarrollos, existe menos duplicación de recursos, y por lo tanto, se aumenta el potencial y se reducen los costes.
- **Estar preparados para el cambio:** La arquitectura SOA permite a las organizaciones estar preparadas para el futuro. Los procesos de negocio se pueden crear mucho más fácilmente si se piensan y se crean en base a servicios gestionados y se dejan las interfaces de interconexión totalmente definidas, permitirá a las compañías seguir creciendo sin necesidad de invertir de nuevo en los mismos desarrollos, y además permite a los sistemas existentes seguir creciendo.

Con esto se puede comprobar que utilizando las arquitecturas SOA se aporta gran valor, pero también hay que recordar que migrar a una arquitectura orientada a servicio no es una tarea trivial. Más que una migración de toda una organización a una arquitectura SOA, se recomienda migrar el conjunto de funciones o artefactos que necesitan interoperar con otras organizaciones, y empezar a utilizar SOA para futuros desarrollos.

También se pueden ver los beneficios de una Arquitectura SOA desde el punto de vista empresarial; algunas de ellas son las siguientes:

- **Eficiencia:** Transforma los procesos de negocio en servicios compartidos con un menor coste de mantenimiento, mayor ROI (*Return Of Investment*).
- **Capacidad de respuesta:** Rápida adaptación y despliegue de servicios, clave para responder a las demandas de clientes, colaboradores y empleados.
- **Adaptabilidad:** Facilita la adopción de cambios añadiendo flexibilidad y reduciendo el esfuerzo.

2.5.1. Conclusiones de las Arquitecturas Orientadas a Servicios

Dentro de un proceso de negocio puede ser necesario utilizar funcionalidades de distintos sistemas y distintas localizaciones; para completar estas funcionalidades se utilizan los Servicios Web, que también tienen que ser identificados durante el proceso de análisis de una arquitectura orientada a servicio. Cada servicio tiene que estar bien definido mediante una interfaz (WSDL) para que pueda ser publicado, localizado e invocado. Dependiendo del proceso de negocio, el servicio puede ser publicado para que otras empresas puedan utilizarlo, o internamente para ser utilizado en los procesos de negocio internos de una determinada compañía.

Los servicios Web son una tecnología altamente adaptable a las necesidades de implementación de una Arquitectura Orientada a Servicio. En esencia, los servicios Web son la implementación de una especificación bien definida de una funcionalidad, es decir, son aplicaciones modulares que aportan una lógica del proceso de negocio como servicio que puede ser publicado, localizado e invocado en Internet [IBM, 2008]. Basados en los estándares XML [W3C, 1998], los servicios Web pueden ser desarrollados usando cualquier lenguaje de programación, cualquier protocolo y cualquier plataforma. Los servicios Web pueden ser localizados y utilizados en cualquier momento, desde cualquier localización y usando cualquier protocolo y plataforma.

Pero es importante remarcar que los servicios Web no son la única tecnología que se puede usar para implementar una Arquitectura Orientada a Servicio. Existen ejemplos de organizaciones que utilizan con éxito una Arquitectura Orientada a Servicios donde utilizan, además de los servicios Web, otras tecnologías de intercambio de mensajes y de acceso a funciones remotas haciendo uso del estándar XML, como puede ser el protocolo XML RPC [Winer, 1999], que funciona exactamente igual que el protocolo RPC (*Remote Procedure Call*), a través de un túnel HTTP.

2.6. LA SEMÁNTICA Y SU APLICABILIDAD EN ENTORNOS E-LEARNING

El término semántica [2009] se refiere a los aspectos del significado o interpretación del significado de un determinado símbolo, palabra, lenguaje o representación formal. En principio cualquier medio de expresión (lenguaje formal o natural) admite una correspondencia entre expresiones de símbolos o palabras y situaciones o conjuntos de cosas que se encuentran en el mundo físico o abstracto que puede ser descrito por dicho medio de expresión.

La semántica puede estudiarse desde diferentes perspectivas:

- **Semántica lingüística**: trata de la codificación y descodificación de los contenidos semánticos en las estructuras lingüísticas.
- **Semántica lógica**: desarrolla una serie de problemas lógicos de significación, estudia la relación entre el signo lingüístico y la realidad. Las condiciones necesarias para que un signo pueda aplicarse a un objeto, y las reglas que aseguran una significación exacta.
- **Semántica en ciencias cognitivas**: intenta explicar por qué nos comunicamos, y cuál es el mecanismo psíquico que se establece entre hablante y oyente durante este proceso.

La lingüística es la disciplina donde originalmente se introdujo el concepto de semántica. La semántica lingüística es el estudio del significado de las expresiones del lenguaje. La semántica lingüística contrasta con otros dos aspectos que intervienen en una expresión con significado: la sintaxis y la pragmática.

La semántica es el estudio del significado atribuible a expresiones sintácticamente bien formadas. La sintaxis estudia sólo las reglas y principios sobre cómo construir expresiones interpretables semánticamente a partir de expresiones más simples, pero en sí misma no permite atribuir significados. La semántica examina el modo en que los significados se atribuían a las palabras, sus modificaciones a través del tiempo y aún sus cambios por nuevos significados. La lexicografía es otra parte de la semántica que trata de describir el significado de las palabras de un idioma en un momento dado, y suele exhibir su resultado en la confección de diccionarios.

Por otro lado, la pragmática se refiere a cómo las circunstancias y el contexto ayudan a decidir entre alternativas de uso o interpretación; gracias a la pragmática el lenguaje puede ser usado con fines humorísticos o irónicos. Además la pragmática reduce la ambigüedad de las expresiones, seleccionando sólo un conjunto adecuado de interpretaciones en un determinado contexto.

La lógica de predicados de primer orden es el tipo de sistema lógico-matemático más sencillo donde aparece el concepto de interpretación semántica. Dicha lógica está formada por:

- Un conjunto de signos (conectivas, paréntesis, cuantificadores,...).
- Un conjunto de variables y constantes.
- Un conjunto de predicados sobre las variables.
- Un conjunto de reglas de buena formación de expresiones a partir de expresiones sencillas.

En la lógica de primer orden el conjunto de variables y constantes juega un papel similar al lexicón de las lenguas naturales, ya que bajo una interpretación semántica son los elementos que admiten referentes. A su vez, el conjunto de reglas de buena formación de expresiones hace el papel de la sintaxis en las lenguas naturales. Para interpretar semánticamente las expresiones formales de un sistema lógico de primer orden se necesita definir un modelo o conjunto estructurado sobre el que interpretar los enunciados formales del sistema lógico. Un modelo de acuerdo con la teoría de modelos es un conjunto con cierta estructura junto con una regla de interpretación que permite asignar a cada variable o constante un elemento del conjunto y cada predicado en el que intervienen un conjunto de variables puede ser juzgado como cierto o falso sobre el conjunto en el que se interpretan las proposiciones del sistema lógico formal.

En lógica matemática se suelen dividir los axiomas en dos tipos:

- **Axiomas lógicos**: que definen básicamente las reglas de deducción y están formados por tautologías. Básicamente son válidos para cualquier tipo de sistema formal razonable.
- **Axiomas matemáticos**: que aseveran la existencia de cierto tipo de conjuntos y objetos con verdadero contenido semántico. Gracias a ello es posible introducir conceptos nuevos y probar las relaciones entre ellos.

Así, si se tiene un conjunto de axiomas que define la teoría de grupos, cualquier grupo matemático es un modelo en el que las proposiciones y axiomas de dicha teoría reciben interpretación y resultan en proposiciones ciertas sobre ese modelo.

La semántica en ciencias cognitivas tiene que ver con la combinación de signos y con cómo la mente atribuye relaciones permanentes entre estas combinaciones de signos y otros hechos no relacionados por naturaleza con estos símbolos. También es muy especial, ya que es la manera de introducir significados dados de uno mismo. Por ejemplo la noción que existe de silla en la que la misma tiene 4 patas, respaldo, etc. Las hay de más o menos patas pero se trata de deslizamiento de sentidos, que se construye en la mente a partir del caso central o prototipo.

2.6.1. La semántica aplicada a la Web

La Web semántica se basa en la idea de añadir metadatos semánticos a la Web. Esas informaciones adicionales, que describen el contenido, el significado y la relación de los datos, se deben proporcionar de manera formal, para que así sea posible evaluarlas automáticamente por máquinas de procesamiento. El objetivo es mejorar Internet ampliando la interoperabilidad entre los sistemas informáticos y reducir la necesaria mediación de operadores humanos [W3C, 2001b].

El precursor de la idea, Tim Berners-Lee, intentó desde el principio incluir informaciones semánticas en su creación (la Web), pero por diferentes causas no fue posible. Por ese motivo introdujo el concepto de semántica con la intención de recuperar dicha omisión.

En la actualidad, la Web está basada principalmente en documentos escritos en HTML, un lenguaje de marcas que sirve principalmente para crear hipertexto en Internet. El lenguaje HTML es válido para adecuar el aspecto visual de un documento e incluir objetos multimedia en el texto (imágenes, esquemas de diálogo, etc.). Pero ofrece pocas posibilidades para categorizar los elementos que configuran el texto más allá de las típicas funciones estructurales, como sucede con otros lenguajes de maquetación (tipo LaTeX).

HTML permite mediante una herramienta de visualización (como un navegador o un agente de usuario) mostrar por ejemplo un catálogo de objetos en venta. El código HTML de este catálogo puede explicitar aspectos como “el título del documento” es “Venta de Equipos”; pero no hay forma de precisar dentro del código HTML si el producto VGN-FS315S es un “portátil”, con un “precio de venta al público” de “1200 €”, o si es otro tipo de producto de consumo o cualquier periférico. Lo único que HTML permite es alinear el precio en la misma fila que el nombre del producto. No hay forma de indicar “esto es un catálogo”, “portátil SONY VAIO” o “1200 €” es el precio. Tampoco hay forma de relacionar ambos datos para describir un elemento específico en oposición a otros similares en el mismo catálogo.

La Web Semántica se ocuparía de resolver estas deficiencias. Para ello dispone de tecnologías de descripción de los contenidos, como RDF y OWL, además de XML, el

lenguaje de marcas diseñado para describir los datos. Estas tecnologías se combinan para aportar descripciones explícitas de los recursos de la Web (ya sean estos catálogos, formularios, mapas u otro tipo de objeto documental). De esta forma el contenido queda desvelado, como los datos de una base de datos accesibles por Web, o las etiquetas inmersas en el documento (normalmente en XHTML, o directamente en XML, y las instrucciones de visualización definidas en una hoja de estilos aparte). Esas etiquetas permiten que los gestores de contenidos interpreten los documentos y realicen procesos inteligentes de captura y tratamiento de información.

2.6.2. Componentes de la Web semántica

Los principales componentes de la Web Semántica son los metalenguajes y los estándares de representación XML [W3C, 1998], XML Schema [W3C, 2004e], RDF [W3C, 1999], RDF Schema [W3C, 2004f] y OWL [W3C, 2004c]. A continuación se realiza una breve descripción de todos ellos:

- **XML**: se trata de un lenguaje extensible de marcas que permite la generación de documentos estructurados aportando una sintaxis superficial, sin embargo no les dota de ninguna restricción sobre el significado de los conceptos que en él se detallan. Ya se realizó una descripción más detallada dentro del apartado 2.5.
- **XML Schema**: es un lenguaje para definir la estructura de los documentos XML.
- **RDF**: es un modelo de datos para los recursos y las relaciones que se puedan establecer entre ellos. Aporta una semántica básica para este modelo de datos que puede representarse mediante XML.
- **RDF Schema**: es un vocabulario para describir las propiedades y las clases de los recursos RDF, con una semántica para establecer jerarquías de generalización entre dichas propiedades y clases.
- **OWL**: añade más vocabulario para describir propiedades y clases: tales como relaciones entre clases (p.ej. disyunción), cardinalidad (por ejemplo "únicamente uno"), igualdad, tipologías de propiedades más complejas, caracterización de propiedades (por ejemplo simetría) o clases enumeradas.

La usabilidad y aprovechamiento de la Web y sus recursos interconectados puede aumentar con la web semántica gracias a:

- Los documentos etiquetados con información semántica (compárese ésta con la etiqueta <meta> de HTML, usada para facilitar el trabajo de los robots de búsqueda). Se pretende que esta información sea interpretada por el ordenador con una capacidad comparable a la del lector humano. El etiquetado puede incluir metadatos descriptivos de otros aspectos documentales o protocolarios.
- Vocabularios comunes de metadatos (Ontología (Informática)) y mapas entre vocabularios que permitan a quienes elaboran los documentos disponer de nociones claras sobre cómo deben etiquetarlos para que los agentes automáticos puedan usar la información contenida en los metadatos (p.ej. el

metadato author tenga el significado de "autor de la página" y no el del "autor del objeto descrito en la página").

- Agentes automáticos que realicen tareas para los usuarios de estos metadatos de la Web Semántica
- Servicios Web (a menudo con agentes propios) que provean de información a los agentes (por ejemplo un servicio de garantías a quien un agente pudiera consultar sobre si un comercio electrónico tiene un historial de mal servicio o de generar correo basura).

A continuación se proporciona una descripción mucho más precisa de algunos de los componentes anteriormente citados ya que serán de vital importancia en el desarrollo posterior de este trabajo.

RDF (Resource Description Framework)

RDF (Resource Description Framework [W3C, 1999]) nació en el seno de la W3C como consecuencia de la necesidad de tener un lenguaje para describir los recursos existentes en la Web, o como sugiere su nombre, debido a la necesidad de tener un lenguaje para expresar metainformación. El origen de RDF se debe a Ramanathan V. Guha cuando trabajaba en Apple Computer en su forma inicial conocida como MCF, más tarde continuada durante su etapa en Netscape Communications Corporation.

Este modelo se basa en la idea de convertir las declaraciones de los recursos en expresiones con la forma sujeto-predicado-objeto (conocidas en términos RDF como tripletes). El sujeto es el recurso, es decir aquello que se está describiendo. El predicado es la propiedad o relación que se desea establecer acerca del recurso. Por último, el objeto es el valor de la propiedad o el otro recurso con el que se establece la relación.

Debido a la similitud entre los conceptos “información” y “metainformación”, se entiende que el uso de RDF se puede extender no sólo a describir recursos, como indican sus siglas, sino que se puede aplicar a la descripción de cualquier objeto que pueda ser identificado. Cuando un objeto sea identificado, éste podrá ser determinado en términos de propiedad/valor, debido a lo cual RDF puede utilizarse para describir objetos que se

encuentren en la Web, fuera de ella, e incluso elementos abstractos sin representación física.

Así RDF surgió como un lenguaje genérico para la descripción de elementos (cualquier entidad que pueda ser identificada) a través de metainformación; la cual podrá incluir las propiedades de dicho elemento, así como las relaciones que se puedan dar con el resto de ellos, elaborando representaciones más complejas, pero siempre procesables por un sistema informático diseñado para tal efecto. A esa descripción de los diferentes elementos, junto con sus relaciones establecidas (contexto), se denominará conocimiento; resumiendo, RDF es por lo tanto un lenguaje para la descripción de conocimiento.

En líneas generales, RDF permitirá establecer una serie de afirmaciones que constarán de sujeto, verbo y predicado. Como se ha comentado anteriormente, estas afirmaciones podrán estar relacionadas, por lo que el predicado de una de ellas podrá ser sujeto de otra, formando con ello redes semánticas o de conocimiento.

La especificación de desarrollo RDF está definida en los siguientes documentos:

- RDF/XML Syntax Specification. Encargado de la descripción de la sintaxis utilizada para expresar el modelo RDF así como sus esquemas.
- RDF Vocabulary Description Language 1.0: RDF Shema. En este documento se describen los esquemas RDF (RDFS).
- RDF Primer. Se trata de un documento introductorio a RDF, que pretende dar una visión general para introducir al lector en el estándar. Este documento no forma parte de lo que sería la especificación formal.
- Resource Description Framework (RDF). Concepts and Abstract Syntax. En este documento se procede a realizar una explicación del modelo abstracto en el que está basado RDF.
- RDF Semantics. Se especifica en este documento una semántica precisa para el vocabulario definido por RDF, así como las reglas de inferencia tanto para RDF como para los esquemas RDFS.
- RDF Test Cases. Se trata de un conjunto de test que recogen las reglas gramaticales de RDF.

A continuación se muestra en la Figura 2.34 un ejemplo de fichero RDF en el que se puede ver la definición de una clase llamada “Países” dentro de la cual se pueden distinguir los tipos “Italia”, “Inglaterra”, “Alemania”, “Francia” o “América”.

Figura 2.34 Ejemplo de fichero RDF

OWL (Ontology Web Language)

El Lenguaje de Ontologías Web (OWL [W3C, 2004c]) está diseñado para ser usado en aplicaciones que necesitan procesar el contenido de la información en lugar de únicamente representar información para los humanos. OWL facilita un mejor mecanismo de interpretabilidad de contenido Web que los mecanismos admitidos por XML, RDF, y esquema RDF (RDF-S) proporcionando vocabulario adicional junto con una semántica formal. OWL tiene tres sublenguajes, con un nivel de expresividad creciente: OWL Lite, OWL DL, y OWL Full.

La Web semántica es una visión del futuro de la Web donde la información está dando un significado explícito, permitiendo que las máquinas puedan procesar automáticamente e integrar la información disponible en la Web. La Web semántica se basará en la capacidad de XML para definir esquemas de etiquetas a medida y en la aproximación flexible de RDF para representar datos. El primer nivel requerido por

encima de RDF para la Web semántica es un lenguaje de ontologías que pueda describir formalmente el significado de la terminología usada en los documentos Web. Si se espera que las máquinas hagan tareas útiles de razonamiento sobre estos documentos, el lenguaje debe ir más allá de las semánticas básicas del RDF Schema.

OWL proporciona tres lenguajes, cada uno con nivel de expresividad mayor que el anterior, diseñados para ser usados por comunidades específicas de desarrolladores y usuarios:

- OWL Lite está diseñado para aquellos usuarios que necesitan principalmente una clasificación jerárquica y restricciones simples. Por ejemplo, a la vez que admite restricciones de cardinalidad, sólo permite establecer valores cardinales de 0 ó 1. Debería ser más sencillo proporcionar herramientas de soporte a OWL Lite que a sus parientes con mayor nivel de expresividad, y OWL Lite proporciona una ruta rápida de migración para tesauros y otras taxonomías. OWL Lite tiene también una menor complejidad formal que OWL DL.
- OWL DL está diseñado para aquellos usuarios que quieren la máxima expresividad conservando completitud computacional (se garantiza que todas las conclusiones sean computables), y resolubilidad (todos los cálculos se resolverán en un tiempo finito). OWL DL incluye todas las construcciones del lenguaje de OWL, pero sólo pueden ser usados bajo ciertas restricciones. OWL DL es denominado de esta forma debido a su correspondencia con la lógica de descripción (Description Logics), un campo de investigación que estudia la lógica que compone la base formal de OWL.
- OWL Full está dirigido a usuarios que quieren la máxima expresividad y libertad sintáctica de RDF sin garantías computacionales. Por ejemplo, en OWL Full una clase puede ser considerada simultáneamente como una colección de clases individuales y como una clase individual propiamente dicha. Su principal problema radica en la poca probabilidad existente de cara a que cualquier software de razonamiento sea capaz de obtener un razonamiento completo para cada característica de OWL Full.

Cada uno de estos sublenguajes es una extensión de su predecesor más simple, respecto a lo que puede ser expresado legamente y a la validación de sus conclusiones. El siguiente grupo de relaciones se mantienen, pero las relaciones inversas no se mantienen.

- Cada ontología legal de OWL Lite es una ontología legal de OWL DL.
- Cada ontología legal de OWL DL es una ontología legal de OWL Full.
- Cada conclusión válida de OWL Lite es una conclusión válida de OWL DL.
- Cada conclusión válida de OWL DL es una conclusión válida de OWL Full.

Los desarrolladores de ontologías que adoptan OWL deberían considerar cuál es el sublenguaje que mejor se adapta a sus necesidades. La elección entre OWL Lite y OWL DL depende de las necesidades de los usuarios sobre la expresividad de las construcciones, proporcionando OWL DL las más expresivas. La elección entre OWL DL y OWL Full depende principalmente de las necesidades de los usuarios sobre los recursos de metamodelado del esquema RDF (por ejemplo, definir clases de clases, o definir propiedades de clases). Cuando se usa OWL Full en comparación con OWL DL, el soporte en el razonamiento es menos predecible, ya que no existen en este momento implementaciones completas de OWL Full.

OWL Full puede ser considerada como una extensión de RDF, mientras que OWL Lite y OWL DL pueden ser consideradas como extensiones de una visión restringida de RDF. Cada documento OWL (Lite, DL, Full) es un documento RDF, y cada documento RDF es un documento de OWL Full, pero sólo algunos documentos RDF serán legalmente documentos OWL Lite u OWL DL. Por este motivo, se ha de tener cuidado cuando un usuario quiera migrar un documento de RDF a OWL. Cuando se considere que la expresividad de OWL DL u OWL Lite es adecuada, han de tomarse precauciones para asegurar que el documento RDF original cumple con las restricciones adicionales impuestas por OWL DL y OWL Lite.

Principales trabajos relacionados

A continuación se presenta en la Tabla 2.12 una lista con los principales trabajos elaborados al respecto, indicando para cada uno de ellos su localización y principales características. La mayor parte de trabajos están relacionados con ontologías, es decir, representaciones RDF de conocimiento semántico, aunque también se han encontrado algunos de ellos relacionados con la descripción semántica de los servicios Web, sustituyendo al clásico documento WSDL por otro que le permitirá describir de una forma más eficiente todo el funcionamiento y características de los diferentes servicios, aportando así una mayor sencillez y precisión a las búsquedas.

Nombre	Características	Localización
CORESE	Una herramienta RDF basada en grafos conceptuales.	Enlace
DOME	Se encarga de crear herramientas de gestión de ontologías que proporcionen soluciones integrales de resolución de problemas.	Enlace
JENA	Es un formato no propietario que ofrece un marco de recursos Java para construir aplicaciones semánticas. Ofrece un entorno para RDF, esquemas RDF y OWL e incluye un motor basado en reglas de inferencia.	Enlace
KAON	Es un gestor de ontologías de código abierto. Incluye un conjunto de herramientas para crear y gestionar ontologías y otras herramientas para construir aplicaciones basadas en ontologías.	Enlace
KIM	Es una plataforma que permite la anotación semántica y que soporta RDF, RDFS y OWL Lite.	Enlace
KOWARY	Herramienta escrita en Java y de código abierto que soporta RDF y OWL.	Enlace
KPONTOLOGY	Es una biblioteca para gestionar ontologías que permite usar diferentes ontologías. Permite utilizar los lenguajes RDF, OWL y ODE.	Enlace
MINDSWAP	Se trata de un editor de ontologías hipermedia basado en OWL.	Enlace
ONTOEDIT	Es una herramienta que permite construir ontologías usando significaciones gráficas.	Enlace
ONTOLINGUA	Provee, en un entorno colaborativo, un buscador, generador y modificador de ontologías.	Enlace
ONTOMAT	Herramienta de código abierto que soporta marcado OWL.	Enlace
ONTOPIA	Ofrece un conjunto de	Enlace

	herramientas para desarrollar y mantener ontologías.	
ONTOSHARE	Es una ontología para la Web basada en RDF y pensada para compartir conocimiento en un ambiente distribuido.	Enlace
ONTOWEAVER	Permite el diseño y desarrollo de sitios Web basándose en ontologías.	Enlace
ONTOWEBBER	Sistema de gestión de sitios Web basado en ontologías.	Enlace
POWL	Plataforma que utiliza RDFS/OWL.	Enlace
OPENCYC	Base de conocimiento general con más de 6.000 conceptos.	Enlace
OWLIM	Es un repositorio semántico para la capa de almacenamiento e inferencia de la base de datos RDF de Sesame.	Enlace
PROTÉGÉ	Editor de ontologías de código abierto para construir ontologías sobre RDFS, OWL y XML Schema. Es uno de los editores más utilizados	Enlace
PROTO ONTOLOGY	Plataforma que sirve para construir ontologías, la anotación semántica, la indización y la recuperación de información.	Enlace
SESAME	Es una base de datos RDF de fuente abierta con soportes de inferencia y consulta para esquemas RDF. Originalmente fue desarrollado por Aduna como un prototipo de desarrollo para el proyecto de la Unión Europea On-To-Knoledge.	Enlace
SUMO	Se trata de otro dominio de ontologías.	Enlace
SWOOP	Editor de ontologías hipermedia.	Enlace
TAP KNOWLEDGE BASE	Sistema de integración de servicios Web dentro de bases de conocimiento.	Enlace
TM-BUILDER	Es un constructor de ontologías basado en Topic Maps.	Enlace
WEBODE	Permite desarrollar ontologías	Enlace

	sobre ingeniería.	
WEBONTO	Consiste en un applet Java que permite navegar y editar modelos de conocimiento sobre la Web.	Enlace
WORDNET	Se trata de un sistema de referencia léxico cuyo diseño está inspirado en la memoria humana (léxica).	Enlace
WSMO STUDIO	Se trata de un editor de ontologías para modelado de servicios de la Web Semántica.	Enlace

Tabla 2.12 Tabla resumen con los principales trabajos relacionados con ontologías

2.6.3. La Web semántica aplicada a entornos e-learning

La mayor cantidad de búsquedas de información que realizan los actuales browsers, de forma más o menos acertada, es realizada mediante ciertas palabras clave incorporadas en el código HTML de las páginas Web dispersas en Internet. En los últimos años, se están realizando anotaciones de datos introducidas dentro de este código HTML, siguiendo algún esquema de anotación común, normalmente basado en XML. La idea es que los datos puedan ser utilizados y "comprendidos" por los ordenadores, sin necesidad de supervisión humana, de forma que los agentes Web, puedan ser diseñados para tratar la información incorporada en las páginas de manera semiautomática. Se trata de convertir la información en conocimiento, referenciando datos dentro de las páginas Web, a metadatos con un esquema común consensuado sobre algún dominio. Para que esta tarea sea efectiva, se necesita que el conocimiento esté representado de forma que sea legible por los ordenadores, esté consensuado, y sea reutilizable. Las ontologías proporcionan la vía para representar este conocimiento.

Dadas estas premisas, ¿por qué no aplicar estas técnicas a los sistemas de formación a distancia? De esta forma se podrían mejorar los resultados obtenidos por los procedimientos de búsqueda que potencian la reutilización del material educativo, y con ello favorecer enormemente la usabilidad de este tipo de arquitecturas. Para ello será necesario que se añadan técnicas semánticas en tres aspectos:

- Objetos de Aprendizaje
- Repositorios
- Servicios Web

Semántica aplicada a los Objetos de Aprendizaje

La definición de ontologías relacionadas con estrategias de enseñanza-aprendizaje es de utilidad porque permite especificar dentro del objeto de aprendizaje, información relevante para el procesamiento de dicho objeto, desde el punto de vista pedagógico. Esto favorece la personalización de la enseñanza basada en las preferencias, el estilo de aprendizaje del estudiante, así como el diseño particular del mismo. Otra clase de ontologías que se necesitan, son las relacionadas con la estructura física del objeto de aprendizaje, para que éste pueda ser utilizado e interpretado en diferentes sistemas de enseñanza, como indican las especificaciones IMS o SCORM, estableciendo un mecanismo de empaquetado de objetos de aprendizaje basado en XML.

Desde el punto de vista tecnológico, las metodologías lingüísticas ayudan a mejorar la utilización de la Lengua en los sistemas informáticos, asimilando, analizando, seleccionando y presentando la información con el objetivo que las máquinas lleguen a "entender" el lenguaje natural, y de esta forma, se satisfagan las necesidades de información de los usuarios con mayor precisión, y se contribuya a superar el problema de exceso de información. Teniendo en cuenta que tales usuarios pueden ser los estudiantes que reciben formación a través de sistemas e-learning, parece evidente que entre ambos campos se pueden establecer diferentes interrelaciones.

Un ejemplo de sistema que cumple estas características puede ser LUISA [2009] (Learning Content Management System Using Innovative Semantic Web Services Architecture). Se trata de un proyecto de investigación financiado por la Comisión Europea que agrupa instituciones del ámbito de la semántica, empresas y universidades líderes en el sector del eLearning. Entre los seis socios participantes se encuentra la Universidad de Alcalá de Henares, que aporta a LUISA su experiencia para manipular las estructuras semánticas de datos (ontologías y metadatos) dirigidas a la descripción de competencias y contenidos en repositorios de objetos de aprendizaje [AYS, 2007].

Otros ejemplos de sistemas podrían ser MuseoSuomi (un sistema de información cultural), ORAVA (portal Web semántico), HealthFinland (un sistema de información médica semántica), Temp-O-Map (un sistema de mediciones térmicas), Promottori (sistema de búsqueda de imágenes), etc. todos ellos del Semantic Computing Research Group (SeCo) de la Universidad de Helsinki [SeCo, 2009].

Semántica aplicada a los repositorios de Objetos de Aprendizaje

Para muchos autores, la idea de repositorio es intrínseca a los objetos de aprendizaje. Los objetos de aprendizaje, como entidades compuestas de las más diversas tecnologías multimedia, no tendrían ninguna cabida, ni significado real, si no se les concibe albergados en dichos repositorios. Los repositorios, por lo tanto, serán grandes almacenes de material digital, en todas sus variantes, pero incorporando mecanismos mucho más complejos que simplemente el hecho de qué es necesario almacenar y cómo se almacenará. Es decir, su propósito no es simplemente almacenar y distribuir dichos objetos, sino permitir que los mismos sean compartidos por distintos estudiantes y, sobre todo, facilitar su reutilización en diferentes desarrollos.

Se pueden identificar dos tipos diferentes de repositorios digitales de objetos de aprendizaje:

- Aquellos que contienen tanto los objetos de aprendizaje, con su contenido de información, como los metadatos que los describen.
- Aquellos que contienen sólo los metadatos de los objetos de aprendizaje, mientras que los objetos, con su contenido de información, se encuentran almacenados en otra ubicación en la que el repositorio puede localizarlos, a partir de la información contenida en los metadatos, y mediante el empleo de una herramienta adecuada para ello.

Además de los recursos lingüísticos tradicionales, y al igual que se ha comentado en el apartado anterior, también en el caso de los repositorios de objetos de aprendizaje, se le podrán aportar sistemas de búsqueda semánticos, que ayuden al usuario a encontrar más fácilmente aquellos recursos educativos que realmente se adapten a sus necesidades. Evidentemente, el elemento fundamental de esta nueva forma de organizar la información del repositorio, serán las ontologías.

En cuanto a la posible utilidad de las técnicas lingüísticas en un repositorio basado en ontologías, se sugieren las siguientes [Ortiz et al., 2006]:

- Utilizar estas técnicas para crear la ontología del repositorio a partir de información textual, especialmente para extraer los conceptos relacionados con los dominios de conocimiento con los que están relacionados los objetos de aprendizaje almacenados.
- Combinar ambas tecnologías en los procesos de búsqueda de objetos, utilizando el conocimiento del dominio representado en la ontología para limitar los resultados de la búsqueda, aplicando la ingeniería lingüística para extraer información de un texto que coincida con esa ontología, es decir, interpretando sólo expresiones que tengan sentido dentro del marco interpretativo de la ontología.

En los últimos años han aparecido muchos estudios que intentan solventar los problemas de reutilización de los objetos de aprendizaje a través de la introducción de semántica en su descripción [Sicilia et al., 2005]. Según [Soto et al, 2007], actualmente la calidad de los registros de metadatos depende, entre otros, de los siguientes factores:

- La información proporcionada en los metadatos depende de la bondad del creador del registro y del tiempo necesario para añadir dicha información.
- Las capacidades de edición o herramientas proporcionadas por el repositorio.
- El nivel de conocimiento del creador del registro sobre los estándares de metadatos de objetos de aprendizaje.
- El modelo conceptual del repositorio: qué entiende el creador del registro que es un objeto de aprendizaje, y qué estructura de información de metadatos debe tener.

Aceptando las especificaciones de metadatos y los estándares, los objetos de aprendizaje almacenados son más interoperables y reutilizables, si bien aún existen varios inconvenientes reseñables en los registros de metadatos de los repositorios:

- La información definida por los estándares (tales como IEEE LOM) no está orientada a ser procesada por agentes externos. Este hecho, dificulta la programación de aplicaciones con capacidad de interactuar formalmente con el repositorio.
- Los estándares actuales son de propósito descriptivo. Proporcionan información sobre los contenidos o el formato del objeto de aprendizaje, pero no disponen de una semántica de ejecución para los LMS ni de un modelo formal que aporte significado dentro del contenido de los registros de metadatos.

A continuación se detalla uno de los repositorios que se diseñó para solventar todos estos inconvenientes, se conoce con el nombre de SLOR (Semantic Learning Object Repository [Soto et al., 2007]). Se destaca de este repositorio que ha sido específicamente diseñado para la creación y administración de metadatos de los objetos de aprendizaje con propósitos de integración e intercambio con otros sistemas. Esta propuesta aporta mejores y nuevas funcionalidades sobre los repositorios actuales, gracias a la posibilidad que la ontología subyacente ofrece para ejecutar inferencias sobre el conocimiento albergado en los registros del repositorio.

Sus funcionalidades están agrupadas en módulos según el principio de escalabilidad:

- La función de creación de un objeto de aprendizaje permite incluir un nuevo registro de metadatos según el modelo conceptual previamente establecido por el creador a través de la interfaz. El método de creación obtiene una

referencia a un registro de metadatos, creando una instancia de la clase correspondiente, y permite establecer las propiedades del mismo según la definición del concepto elegido en la ontología subyacente, lo que puede ocasionar variaciones entre modelos.

- La búsqueda semántica permite solicitar instancias de las distintas conceptualizaciones del modelo ontológico, por ejemplo, recuperar todos los objetos digitales, o todos los objetos con propósito educacional. Por otro lado, los registros de SLOR almacenan información enlazada a conceptos de otras ontologías. Este escenario proporciona un profundo nivel de búsqueda que permite construir consultas complejas. Todas estas restricciones son almacenadas en una lista antes de invocar el método de búsqueda.

A continuación se muestra en la Figura 2.35 cómo sería una búsqueda de objetos de aprendizaje en el repositorio SLOR. Se puede ver en la parte inferior de la imagen cómo se relacionan los términos citados en la parte superior, en base a la ontología utilizada.

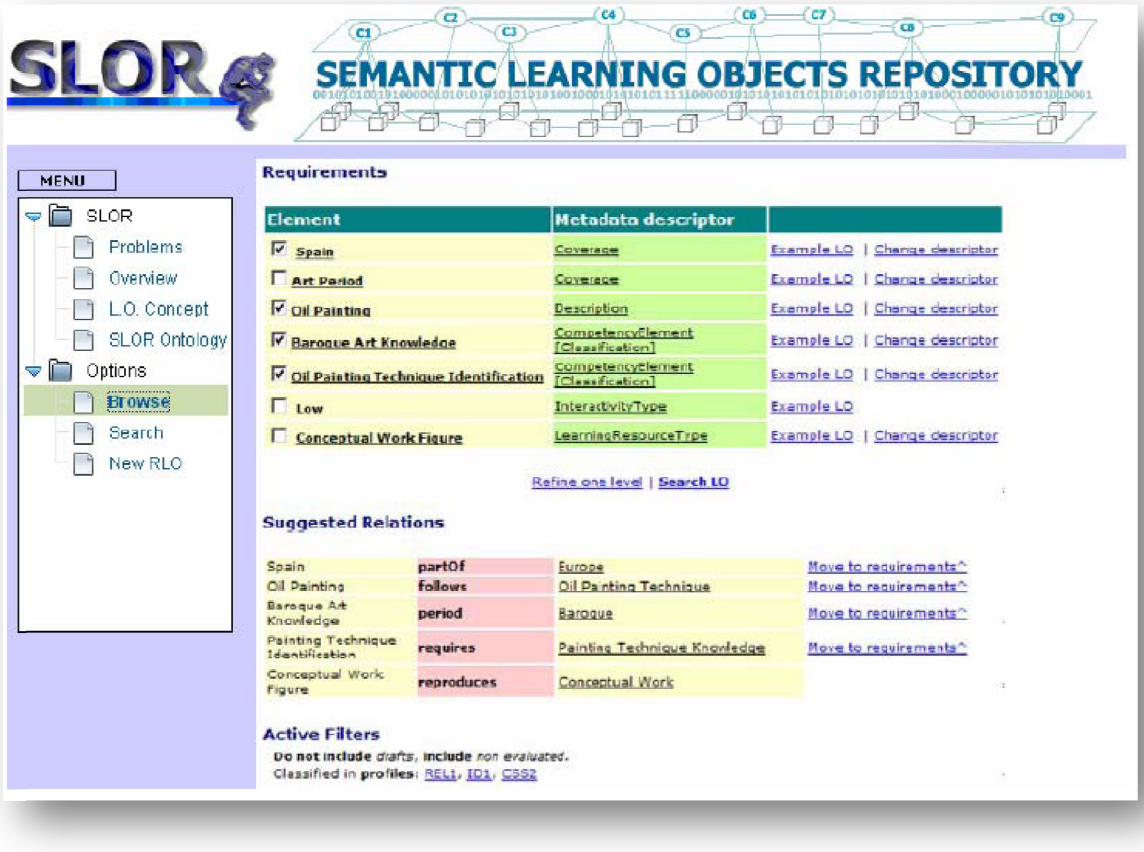


Figura 2.35 Búsqueda de Objetos de Aprendizaje en el repositorio SLOR

Semántica aplicada a los servicios Web

Otro de los nexos de unión que podemos realizar en este trabajo, es el de los servicios Web semánticos, a través de la siguiente relación:

- Servicios Web (Web de aplicaciones) + Web Semántica (Web de datos) = Servicios Web Semánticos (Web de datos y aplicaciones).

Los servicios Web constituyen un mecanismo de reutilización de componentes software, distribuidos en diferentes servidores Web, que realizan una funcionalidad o actividad de negocio, que se ofrece a las aplicaciones que los invocan a través de una interfaz bien definida. Los servicios Web, pueden registrarse y publicarse en la Web, y hacen uso de protocolos abiertos y estándares de Internet, como HTTP, XML, UDDI, SOAP y WSDL, que solucionan el problema de la interoperabilidad entre las diferentes tecnologías y plataformas software utilizadas en la parte cliente desde donde se invocan, y en los servidores en los que se ubican.

Los servicios Web semánticos son una línea importante dentro de la Web semántica, que propone describir no sólo la información de acceso a un servicio, sino definir vocabularios de funcionalidad y procedimientos para describirlos. Tal descripción abarca aspectos como entradas, salidas, procesos, condiciones necesarias para que se puedan ejecutar, los efectos que producen y la información para localizarlos. Se habla de servicios Web semánticos porque se agrega semántica explicita a la descripción de los mismos, por medio de la adición de metadatos, utilizando para ello ontologías. Aunque la especificación actual de servicios Web contiene metadatos en su descripción, a través del fichero WSDL (Web Service Description Language), éstos no son considerados como semánticos puesto que no están relacionados con ontologías.

En contraste, los servicios Web semánticos, contemplan la cooperación y comunicación entre servicios. De esta forma, es posible definir servicios semánticos, que describan la manera de interactuar con otros servicios, en ausencia de elementos externos, utilizando únicamente los elementos existentes para definir este tipo de servicios.

Los servicios Web semánticos van un paso más allá en su intención de extender la Web, desde un conjunto distribuido de fuentes de información a un conjunto distribuido de servicios, proporcionando un grado mayor de dinamismo y automatización en términos de capacidad de proceso por parte de la máquina y del entendimiento hombre-máquina, convirtiéndose de ese modo en el siguiente paso natural en el desarrollo de las tecnologías de Web Semántica. Las ontologías facilitan la semántica para el proceso por parte de las máquinas que, encima de los actuales Servicios Web materializan y posibilitan la idea de los Servicios Web Semánticos. Los SWS se definen como "servicios Web, desacoplados y anotados semánticamente, con una semántica de ejecución concreta, que pueden ser publicados, descubiertos, seleccionados, compuestos, intermediados y ejecutados a través de la Web, siguiendo un paradigma de dirección por tarea, portando su interfaz de interacción mediante orquestación y/o coreografía". La definición establece diferentes aspectos de los Servicios Web que requieren una clarificación ulterior:

- **Desacople**: La inteligencia interna de los procesos de negocio en las aplicaciones debe ser ocultada a un posible acceso público, diferenciando entre la parte pública (interfaz) y la privada (implementación) de los servicios. La parte pública de un servicio debería usar protocolos de intercambio de mensajes para comunicarse, facilitando de ese modo la disgregación, tanto como sea posible.
- **Anotados semánticamente**: Los servicios Web semánticos se describen por medio de sus interfaces y sus capacidades. En ambos casos, tanto el interfaz como la capacidad del servicio deben ser semánticamente anotados para permitir la interoperabilidad y su uso dinámico. Estos metadatos se especifican en un lenguaje formal para permitir su total procesado por las máquinas.
- **Semántica de ejecución**: La interacción entre servicios requiere una semántica formal de ejecución para especificar de manera unívoca el comportamiento que hay detrás de su ejecución.
- **Mediación**: En un entorno escalable, las comunicaciones deberían permitir que cualquiera pudiera hablar con quien quisiera. La mediación es el elemento primordial que facilita dicha comunicación, haciendo que las barreras existentes en términos de datos, protocolos de mensajería e incluso procesos de negocio se diluyan, habilitando mecanismos de traducción entre éstos.
- **Coreografía vs. Orquestación**: Cuando servicios que cooperan para alcanzar algún tipo de funcionalidad se comunican entre ellos, podemos clasificar el tipo de interacción que se produce entre ellos en dos categorías: orquestación y coreografía, dependiendo del nivel de control que unos tienen

sobre otros cuando se comunican. Mientras que en la coreografía una parte controla el proceso de interacción definiendo cómo comunicarse con el servicio para usar su funcionalidad, en una interacción orquestada el nivel de control de todas las partes en la comunicación es la misma, alcanzando la funcionalidad deseada mediante la cooperación de los distintos servicios.

Los servicios Web semánticos, lo mismo que la Web semántica con respecto a la Web actual, representan una extensión a la actual tecnología de servicios Web. La principal diferencia con respecto a los servicios Web tradicionales es que los SWS han sido enriquecidos con metadatos. Dicho enriquecimiento permite profundizar en los conceptos que subyacen en el procedimiento de uso de los servicios Web semánticos, verbigracia: publicación, descubrimiento, selección, composición, mediación, ejecución, monitorización, compensación, sustitución y auditoria. Dichos conceptos, realzados con la información proporcionada por la anotación semántica permite un mayor dinamismo y un mayor grado de automatización.

Actualmente la tecnología no ha cubierto el hueco que hay entre lo que la Web ofrece y los requisitos técnicos de los servicios Web semánticos. La Web semántica en general y las ontologías en particular son los medios necesarios para salvar dicho hueco y materializar una Web mucho más dinámica. En la actualidad, existen diferentes tecnologías para el desarrollo de servicios Web tanto semánticos como no semánticos.

Una de las mejores alternativas, en el ámbito de los no semánticos, es el Lenguaje para la Ejecución de Procesos de Negocio para Servicios Web (BPEL4WS: Bussines Process Execution for Web Services [OASIS, 2009]). BPEL4WS se centra en proveer un lenguaje robusto para describir interacciones de procesos, flujos de control, flujos de datos, excepciones y condiciones de error, que permiten componer servicios Web a partir de sus documentos de descripción WSDL. BPEL4WS no incluye una relación estricta con ontologías y por tanto la composición no puede ser considerada como un servicio Web semántico. Una de las implementaciones prácticas que tenemos al efecto, es ActiveBEPEL [2009].

En cuanto a las tecnologías para la composición de servicios Web semánticos, se puede destacar por ejemplo el Lenguaje para Ontologías de Servicios Web (OWL-S Ontology Web Language – Services [W3C, 2004d]) y la Ontología para Modelar Servicios Web (WSMO: Web Services Modeling Ontology [W3C, 2005a]).

- OWL-S[W3C, 2004d]: Se trata de una ontología basada en OWL, que permite agregar semántica a la descripción de los servicios, y realizar composiciones a partir de un conjunto de constructores de control, permitiendo describir las interacciones entre servicios, definiendo condiciones, ciclos repetitivos, secuencias y otros comportamientos posibles. OWL-S define tres partes funcionales que son: Perfil del Servicio, Modelo del Servicio y Acceso al Servicio.
- WSMO [W3C, 2005a]: El objetivo de esta iniciativa es resolver el problema de integración mediante una tecnología coherente que habilite el uso e implementación de los servicios Web semánticos. WSMO únicamente define el modelo conceptual para describir este nuevo tipo de servicios Web, sin embargo, se apoya en otras dos iniciativas para su representación sintáctica y su ejecución, como son el Lenguaje para Modelar Servicios Web (WSML: Web Services Modeling Language [W3C, 2005c]), encargado de la descripción sintáctica de los servicios, siguiendo el modelo establecido por WSMO. Por otro lado, el Entorno de Ejecución para el Modelado de Servicios Web (WSMX: Web Services Modeling Execution Enviroment [W3C, 2005b]), responsable de soportar la ejecución de los servicios Web semánticos descritos con WSML y basados en el paradigma definido por WSMO.

2.7. CONCLUSIONES

De todo lo expuesto anteriormente se destacan las características de los sistemas de aprendizaje, tanto los LMS como los LCMS, y sus diferencias; la definición de objeto de aprendizaje y de su evolución en un objeto de aprendizaje reutilizable; y la posibilidad de utilizar repositorios digitales para almacenar objetos de aprendizaje y hacer que sean compartidos y por tanto reutilizables.

Como se comentó anteriormente, uno de los mayores problemas a los que se enfrenta hoy en día la industria del e-learning es la interoperabilidad entre distintos sistemas de aprendizaje que quieren compartir su contenido en forma de objeto de aprendizaje reutilizable.

Para que esta interoperabilidad sea posible, es necesario que los estándares se apliquen desde la creación de los objetos de aprendizaje hasta la construcción y utilización de las herramientas que dan soporte al e-learning. Sin embargo, no hay ningún estándar que garantice completamente los objetivos de accesibilidad, interoperabilidad, durabilidad y reutilización de los materiales docentes basados en las redes.

Actualmente, el uso de estándares en e-learning se enfrenta a muchos y variados desafíos, entre los que se destacan los siguientes:

- Los estándares deben abarcar todo el proceso, pasando por la captación de contenidos, almacenamiento de objetos, trasformación de objetos mediante herramientas y distribución en forma de lecciones o cursos.
- Avances tecnológicos incompatibles o poco vinculados entre sí.
- El formato de los metadatos varía en cada plataforma.
- Especificaciones que después de su implementación no satisfacen todas sus expectativas.
- Necesidad de adaptación de otras formas de educación a distancia a aplicaciones basadas en Web.
- Formas de implementaciones muy diferentes (audio, realidad virtual, video, gráficos, CD-ROM, Internet, redes internas, texto, etc.).
- Es necesario certificar cada contenido en cada plataforma individual.
- Especificaciones muy genéricas y de poco valor en casos complejos o excesivamente particulares [Martínez et al., 2001].

- La industria ha estado creciendo sin tener una idea muy clara de los componentes del e-learning y sus interacciones, por lo que la necesidad de definir una arquitectura global es crítica para la evolución de los estándares.

Como se comentó en apartados anteriores no existe un estándar global para e-learning, sino una serie de grupos que desarrollan especificaciones que la industria trata de seguir aún cuando no hayan sido todavía adoptados formalmente como estándares. Parece que el mercado está convergiendo hacia las especificaciones de ADL-SCORM, el cual integra distintos esfuerzos realizados por organismos como AICC, IEEE e IMS. Sin embargo, SCORM no cubre todos los aspectos, como por ejemplo la interoperabilidad entre distintos sistemas de aprendizaje.

Por otra parte, los trabajos propuestos por el grupo IMS son realmente importantes en el ámbito del e-learning, principalmente por su:

- **Amplia visión**. Cubren todos los aspectos relacionados con la teleformación.
- **Dinamismo**. Incorporan novedades de una manera casi inmediata sin tener que esperar a realizar tareas tan completas como puedan ser las que desarrollan los grupos que proponen estándares y no especificaciones, como el IEEE.
- **Permanente actualización**. A pesar de lo anterior cuando, con posterioridad, la especificación se convierte en un estándar, IMS incorpora las modificaciones de modo inmediato por estar en permanente contacto y seguimiento con los otros grupos (de hecho forman parte de ellos).
- **Claridad**. Las especificaciones que proponen resultan siempre claras en su concepción y estructura, en forma de documentos XML fáciles de utilizar.

De las diversas especificaciones y estándares que se han propuesto, las más interesantes para este trabajo son:

- La especificación de IMS sobre repositorios digitales y servicios empresariales, con especial atención a los protocolos de interoperabilidad.
- Especificaciones que abogan por garantizar la interoperabilidad entre sistemas, como las mencionadas anteriormente CEN SQI y CEN SPI.
- Por lo que respecta a actividades relacionadas con datos y metadatos, se sigue la propuesta de IMS, que es fundamentalmente la que recoge IEEE en

su grupo de trabajo P1484.12, que apareció como estándar LOM *Learning Objects Metadata* y que ISO está en proceso de adopción como estándar ISO.

- Para composición de estos objetos de aprendizaje así como todas sus relaciones con el resto de arquitecturas, se destaca IMS Digital Learning Services Standars. De esta forma IMS agrupa en un solo estándar todo lo referente a composición de objetos de aprendizaje (IMS CC), herramientas de interoperabilidad (IMS LTI) y por último todo lo referente a privilegios y resultados relacionados con el aprendizaje (IMS LIS).
- Para terminología general estará basado en el Glosario propuesto por IEEE a través de su grupo P1484.3. En la actualidad el CEN (Comité Europeo de Normalización) está trabajando en la adecuación de los glosarios existentes (todos en Inglés) a las diferentes lenguas nacionales que forman parte de la unión europea.

Con respecto a las propuestas arquitecturales, aunque se han incluido pocos ejemplos, puede decirse que IEEE, CISCO (apartado 2.4) e IMS "*Abstract Framework*" (explicado en el apartado 2.3.1), son verdaderas arquitecturas y no ya implementaciones concretas de una arquitectura.

Para la definición de la arquitectura planteada en la tesis se han tenido en cuenta los trabajos del grupo de IEEE "P1484.1 *Architecture and Reference Model*" en sus especificaciones de Nivel 3, sin seguirlos de modo exhaustivo, ya que abarca elementos que no nos conciernen de modo directo. En esta especificación se habla de los componentes, y sus relaciones, que deberían figurar en todo sistema de teleformación. Se ha intentado ser fiel a sus planteamientos, criterios y propuestas en esta tesis.

Dentro del modelo de arquitectura planteado por CISCO, se ha centrado el desarrollo de este trabajo en el estudio del nivel 2 "*The Application Blueprint*", y dentro de este en el apartado de gestión de contenidos.

La arquitectura definida por IMS en su *Abstract Framework* es una referencia clara para los propósitos de esta tesis, ya que está basada en una arquitectura orientada a servicios, que es precisamente el enfoque seguido en la definición que se propone en la tesis. Si además de eso se añade que es el modelo de arquitectura que utiliza IMS para la definición de sus especificaciones, es necesario tomarla como un modelo a aplicar.

Como aspectos comunes a todas las arquitecturas presentadas se puede destacar que todas están claramente modularizadas en cuanto a configuración y funcionalidades, y que, en general, están divididas en capas o niveles. En la arquitectura presentada también se utilizará este estilo de representación, pero adaptado a las especificaciones de las arquitecturas orientadas a servicios.

Por último comentar que durante el último apartado de este capítulo se introdujeron diferentes tecnologías que se podrían implementar en un sistema de reutilización de objetos de aprendizaje como el que se detalla en esta Tesis (de hecho, ya se han realizado bastantes implementaciones al respecto). Gracias a estas implementaciones, será factible la elaboración de un sistema que realice una búsqueda exhaustiva y efectiva de material educativo completamente reutilizable y basado en estándares.

3. PLANTEAMIENTO DEL PROBLEMA

En este capítulo se analiza en primer lugar la evolución que han sufrido los sistemas de aprendizaje desde su origen para, posteriormente, detectar todas las necesidades demandadas por sus usuarios (alumnos, administradores, profesores, etc.), y cómo los diferentes entornos se van adaptando en la medida de lo posible para cubrir estas carencias. Todas estas necesidades se intentarán paliar a través del uso de diversas especificaciones (ya explicadas en el anterior capítulo), de forma que se estandarice en la medida de lo posible la mayor parte de la arquitectura que se propone en el siguiente capítulo.

También durante este capítulo se realizará un estudio de los principales repositorios y arquitecturas creadas hasta la fecha, indicando para todas ellas las principales características con las que cuentan, así como sus principales inconvenientes, sobre todo a la hora de interactuar con otras y garantizar así un proceso completo de reutilización de objetos de aprendizaje; de este estudio surge precisamente el fundamento para la realización de esta Tesis.

3.1. INTRODUCCIÓN

En la actualidad existen dos elementos clave para el correcto desarrollo de sistemas e-learning: la reutilización de objetos de aprendizaje y la interoperabilidad entre los repositorios de objetos de aprendizaje. Para conseguir la completa reutilización de un objeto de aprendizaje (LO: *Learning Object*) se debe, en primer lugar, desarrollarlo conforme a estándares de e-learning, como los establecidos por IMS, ADL (SCORM) o IEEE (LOM) explicados en el capítulo anterior. De esta forma se asegura que, a través de su empaquetamiento y descripción mediante metadatos, pueda ser integrado en cualquier plataforma e-learning compatible con estos estándares. Es más, existe el ejemplo de la organización IMS y de su especificación Digital Learning Services Standards, que data de Mayo de 2008, en la cual se determina cómo debería ir empaquetado un objeto de aprendizaje (IMS CC [IMS, 2008]), cómo deberían de interoperar las diferentes aplicaciones y sistemas que los usan (IMS LTI [IMS, 2006]) y cómo se debería tratar la información referente a los privilegios y resultados del aprendizaje (IMS LIS [IMS, 2007]). En segundo lugar, se ha de publicar en un repositorio que garantice su localización automática por parte de los diferentes tipos de usuarios que puedan verse implicados en un proceso educativo. Pero para que esta reutilización sea completa, estos recursos deben poderse descubrir desde otros repositorios o plataformas de formación distribuidos a través de Internet, en lo que se conoce como búsquedas federadas.

Esta Tesis parte de un trabajo anterior del autor y de otros profesores del Departamento de Ciencias de la Computación de la Universidad de Alcalá donde se realizaron los estudios de Doctorado. Inicialmente se planteó una arquitectura que posibilitara la reutilización de objetos de aprendizaje a través de la búsqueda federada de los mismos en repositorios distribuidos, paliando de esta forma todos los inconvenientes de los repositorios actuales. Todo este estudio fue plasmado en la tesis doctoral de Salvador Otón [Otón, 2006]. Al mismo tiempo, se desarrolló un sistema que, basándose en la arquitectura anteriormente descrita, ponía de manifiesto todos los supuestos que en ella se enunciaban. Este sistema fue presentado como proyecto fin de carrera de los estudios de Ingeniería en Informática del autor [Ortiz, 2006].

Una vez realizado este sistema, y tras ver las nuevas necesidades que iban surgiendo (sobre todo en cuanto a interoperabilidad se refiere), se decidió modificarlo para adaptarlo a especificaciones como SQI, que abogan por garantizar la interoperabilidad entre sistemas a través del uso de una interfaz sencilla de consulta. Este será el sistema

inicial del que parte el estudio de esta Tesis. Sin embargo, surgen nuevas necesidades, que conformaran el cometido de esta Tesis, así se destacan las siguientes:

- Proponer una extensión de SQI para la composición semántica de objetos de aprendizaje.
- De la necesidad anterior, surge por lo tanto la necesidad de proponer una extensión a LOM para lo mismo.
- Ampliar el buscador del sistema inicial para poder hacer consultas con composición/agregación de objetos, de forma que un usuario solicite un determinado curso y el nuevo sistema busque objetos de aprendizaje en diferentes repositorios distribuidos y conforme un curso completo a través de la composición de estos objetos.
- Ofrecer una utilidad para envolver cualquier repositorio SQI/LOM, como a los que accede ahora el buscador del sistema inicial, para que soporte las extensiones propuestas.

En los siguientes apartados de este capítulo se irá detallando cuál es la situación actual en cuanto a la interoperabilidad de los actuales sistemas de aprendizaje, para determinar cómo se pueden publicar y descubrir sus contenidos didácticos y cómo son capaces de intercambiarlos con otros sistemas o repositorios, y de esa forma determinar cuáles serán las necesidades y problemáticas detectadas, que se tratarán de solventar con el presente trabajo. Por otro lado, se indicarán cuáles son las principales características del sistema y la arquitectura desarrolladas con anterioridad, el cual supone el punto de partida de la Tesis.

3.2. EVOLUCIÓN DE LOS SISTEMAS DE APRENDIZAJE

Los primeros sistemas de aprendizaje eran más cerrados y limitados que los actuales y se ejecutaban sobre ordenadores aislados. Se trataba de cursos de autoestudio y prácticamente lo único que cambiaba con respecto a un libro era el soporte (de papel a soporte magnético). Estos fueron los principios de lo que hoy se conoce como Aprendizaje Asistido por Ordenador (*Computer Based Training* – CBT); hay términos alternativos a este y que vienen a significar lo mismo como son CAI (*computer-assisted instruction*), CBI (*computer based instruction*) y CAL (*computer-assisted learning*).

La principal característica de este tipo de sistemas es su completo aislamiento, ya que estaban estructurados en un único paquete formado por el cliente y el LMS, y cada alumno disponía de una copia completa del sistema (cursos en CD-ROM u otros soportes anteriores). Estos sistemas adolecían de problemas de actualización, interactividad y otros muchos, pero para la época en que se comenzaron a utilizar también presentaban grandes avances y ventajas sobre todo en cuanto a costes y requerimientos hardware para su ejecución. Evidentemente estos sistemas no eran en absoluto distribuidos.

Siguiendo la misma evolución que cualquier otro tipo de sistema informático, se pasó a la distribución del sistema a través de las redes telemáticas. Esta distribución fue beneficiosa y se basó en las primeras redes, con clientes, protocolos específicos y extensión de los sistemas a las redes existentes, ya sean de área local o área extendida propietaria. Estos sistemas estaban formados básicamente por tres componentes activos: por un lado los equipos “clientes” de los usuarios, por otro el servidor con los contenidos y las herramientas de gestión del aprendizaje y, por último, la red que los comunica, como se muestra en la Figura 3.1. Estos primeros sistemas aparecieron cuando las capacidades de transmisión eran reducidas y no era factible enviar por la red cierto material como el video o el audio por limitaciones de ancho de banda. Además, la capacidad de almacenamiento de servidores y la de visualización de clientes, tampoco permitían grandes alardes multimedia dados los requerimientos hardware de estos contenidos.

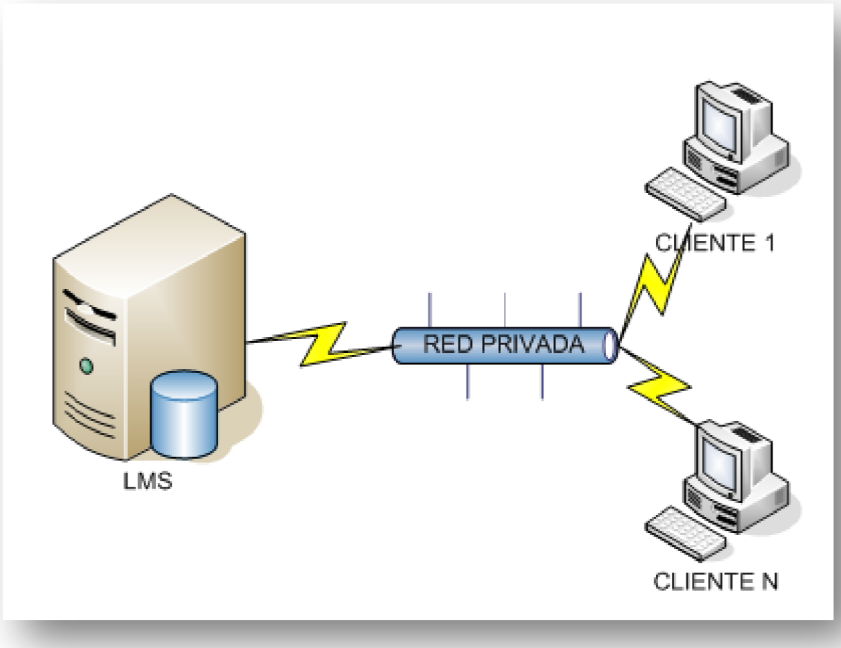


Figura 3.1 Sistema LMS basado en red local

Las redes y los equipos de hardware han seguido evolucionando a gran velocidad. En el momento actual, el ancho de banda que soportan las redes es muy elevado y los equipos tienen grandes capacidades multimedia. Además, con la aparición de Internet se ha producido una autentica revolución en la evolución de este tipo de sistemas, dando paso a protocolos estandarizados, sustituyendo protocolos y piezas de software propietarias, y su total integración con navegadores y servidores Web.

Sin embargo, a pesar de esta evolución, se siguen utilizando las mismas estructuras, el soporte de los contenidos es fijo y rígido (ya sea en el CD-ROM, el servidor específico o el servidor Web) y la interacción de los clientes se limita al uso de los contenidos del servidor. Como se aprecia en la Figura 3.2, se continúa teniendo un esquema similar a los anteriores, donde el único componente nuevo es Internet; esto significa un cambio de plataforma y no de concepto.

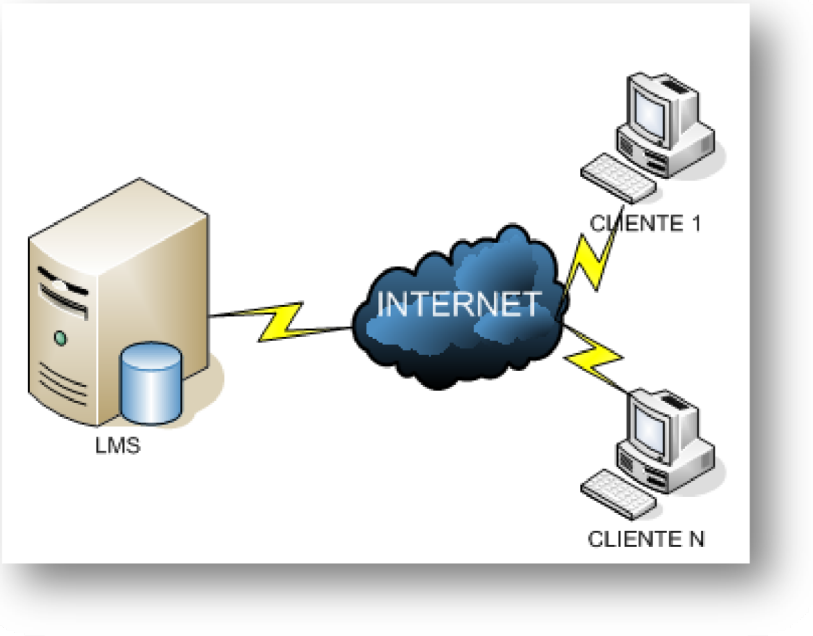


Figura 3.2 Sistema LMS basado en Internet

Es en este punto actual de la evolución donde parte el estudio de esta Tesis (recordemos que parte de un sistema ya diseñado de reutilización de objetos de aprendizaje). El objetivo que se plantea es abrir los sistemas para conseguir que se comuniquen entre ellos. En esta comunicación se producirá un intercambio de contenidos donde un usuario del sistema será capaz de localizar objetos de aprendizaje fuera de su LMS o LCMS, y donde un LMS o LCMS será capaz de publicar sus objetos de aprendizaje para que sean utilizados en otros, independientemente del formato de metadatos utilizado. La Figura 3.3 representa esta interconectividad entre sistemas.

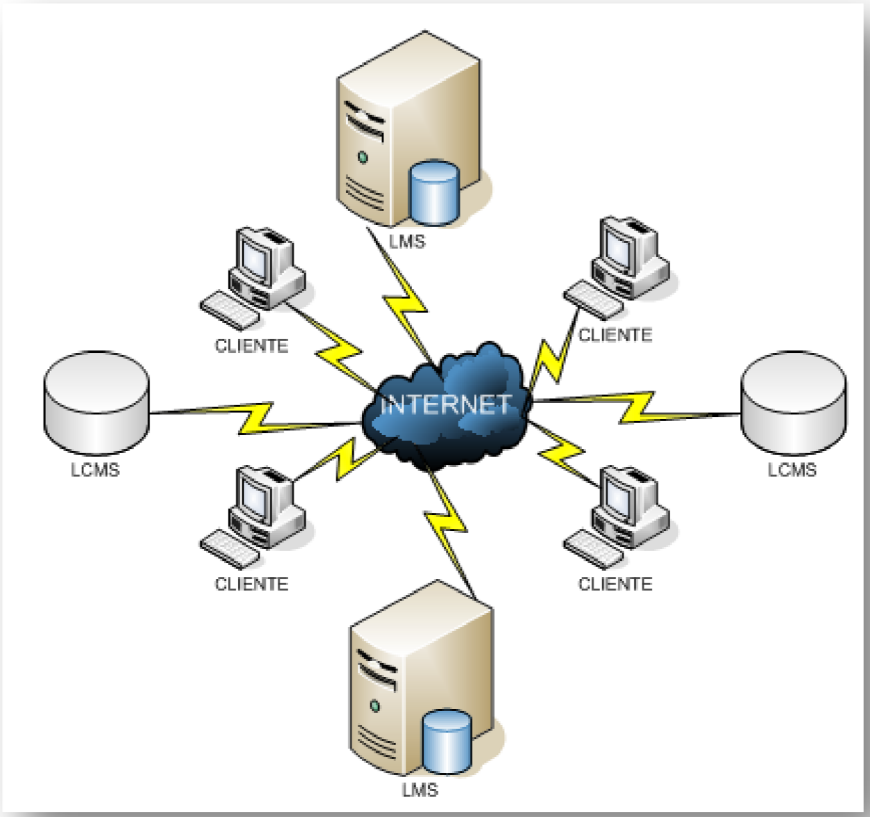


Figura 3.3 Sistemas que interactúan sobre la red

Pero, desde un punto de vista arquitectural, la cuestión es cómo conseguir ésta comunicabilidad. Obviamente se necesita un nexo de unión entre todos los participantes del sistema. Como ya se ha comentado anteriormente, se desarrolló un sistema que posibilitaba esto, al que se denominó LORS (Learning Objects Reusability System) [Otón et al., 2007], el cual se encarga de distribuir los contenidos docentes entre los distintos participantes. Esta forma de trabajo se representa en la Figura 3.4.

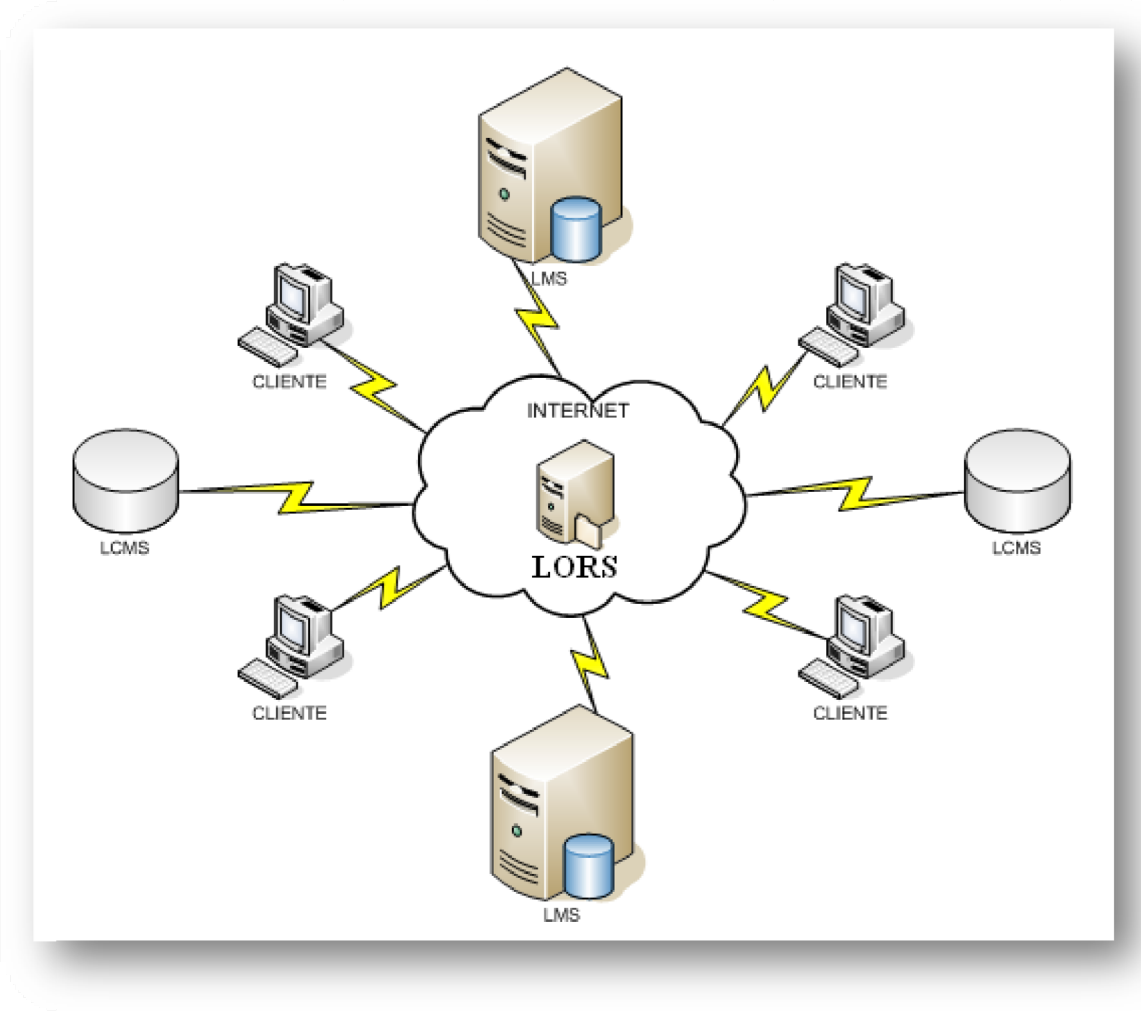


Figura 3.4 LMS, LCMS y clientes interactuando con LORS

Posteriormente el sistema se modificó para adaptarlo a la especificación SQI, de forma que garantizara la interoperabilidad entre SROA y todos los repositorios que cumplieran dicha especificación. Este es el sistema del que parte el estudio de esta Tesis, el que podríamos denominar **LORS v1.0**.

La idea que se plantea podría servir para diversas iniciativas en la distribución de contenidos educativos, tanto de carácter libre o gratuito, como comercial. Desde la perspectiva del material libre, podría servir sobre todo en el ámbito universitario, donde se crearían nodos de compartición de conocimiento fomentando el intercambio enriquecedor de la cultura que es una de sus misiones. Desde la perspectiva comercial se podría ver como un nuevo servicio de pago para la compra de material didáctico de alta calidad para la confección de nuevos cursos.

3.3. REPOSITORIOS DE OBJETOS DE APRENDIZAJE

Actualmente existen diferentes iniciativas en la dirección marcada en el apartado anterior; se trata de repositorios de objetos de aprendizaje que tratan de compartir sus recursos. Como se comentó en el capítulo anterior, la finalidad de éstos es permitir el almacenamiento de contenidos docentes, posibilitando su distribución a través de sistemas de localización, acceso y obtención remota por parte de diferentes usuarios o sistemas. Estos repositorios tienen la misión de servir de punto de encuentro entre los creadores de contenidos y los "consumidores" de los mismos, haciendo de éstos contenidos reutilizables.

Generalmente estos sistemas suelen ser gratuitos, tanto en su uso como en el precio de los contenidos que almacenan; por lo tanto, no se puede garantizar la calidad del material docente que almacenan ni en sus contenidos ni en los metadatos que los describen. Por otro lado, estos repositorios no llegan a ser útiles si no utilizan metadatos para la clasificación de sus objetos de aprendizaje [Gutiérrez y Otón, 2004], y si los utilizan suelen ceñirse a un tipo determinado de especificación para la confección de los metadatos, haciendo la búsqueda y reutilización muy compleja si se cambia esta especificación. Cierto es, también, que las especificaciones existentes no ayudan demasiado a esta tarea por su falta de completitud y especificidad para muchos de los metadatos que deben ser rellenados para anotar cada objeto docente.

Sin embargo, este tipo de sistemas representa una pieza clave a la hora de reutilizar el material didáctico, ya que se pueden convertir en fuentes de información donde los docentes busquen y publiquen sus trabajos, pero deben ir acompañados de una metodología de creación de ese material de forma que sigan los estándares. También sería muy recomendable acompañar a estos sistemas de algún mecanismo de pago, de forma que se puedan comprar contenidos de calidad contrastada.

A continuación se analizarán algunos de los repositorios que publican sus contenidos en Internet con objeto de detectar sus propiedades y carencias. Un buen punto de entrada es la página Web de ARIADNE (uno de los repositorios que se analizará) donde se publica un resumen de los principales repositorios que cumplen con la especificación SQI. La Web en cuestión es:

http://ariadne.cs.kuleuven.be/SqiInterop/free/SQIImplementationsRegistry.jsp

3.3.1. MERLOT

MERLOT (Multimedia Educational Resource for Learning and Online Teaching) [MERLOT, 2009], es un repositorio libre y gratuito diseñado principalmente para estudiantes de educación superior o universitarios, donde puedan dejar y encontrar recursos de aprendizaje "on-line" y evaluarlos. En él se puede inspeccionar su catálogo de contenidos relacionados con campos como negocios, arte, humanidades, ciencia y tecnología, ciencias sociales. También se pueden realizar búsquedas sobre él mismo; para ello, los contenidos suelen llevar una pequeña descripción mediante la utilización de algunos metadatos como la descripción, autor, audiencia, idioma, coste, derechos de autor, etc. Estos metadatos se basan en el estándar LOM [IEEE, 2002]. En la Figura 3.5 se puede ver la interfaz de cómo sería una búsqueda federada en MERLOT.

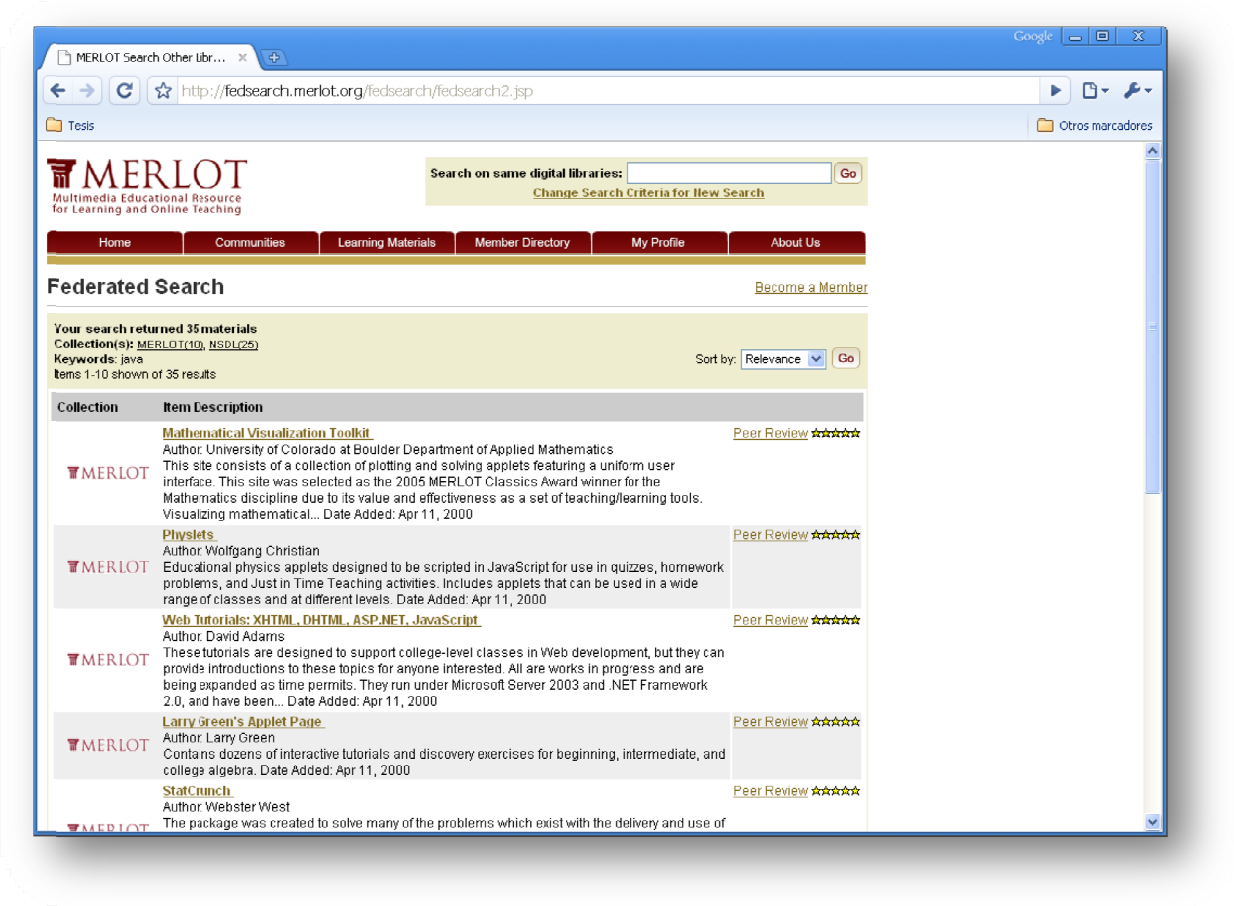


Figura 3.5 Repositorio MERLOT

El contenido físico generalmente es un enlace al contenido alojado en la Web del autor. El principal problema es que no suele ser fácil integrar ese contenido en un curso nuevo que un usuario quiera elaborar, ya que no da la posibilidad de descargarlo en un

formato empaquetado estándar como SCORM, en el que se tendrán los contenidos y los metadatos agrupados. Esto es un gran inconveniente, ya que para integrarlo en otro repositorio se debería empaquetar de forma manual con una herramienta como Reload [Reload, 2009] y rellenar los campos de metadatos con la información proporcionada por MERLOT campo a campo (aquellos que proporcione). Además de este inconveniente, como se señala en [Pagés et al., 2003], la completitud en los campos de los metadatos de los objetos de aprendizaje no es lo suficientemente buena para su correcta descripción.

3.3.2. CAREO

CAREO (Campus Alberta Repository of Educational Objects) [CAREO, 2009] es un proyecto cuyo objetivo es alcanzar la realización de una colección de materiales de aprendizaje multidisciplinares, basados en Web, que estén a disposición de los profesores en todo Canadá e incluso fuera de las fronteras del país. Es un proyecto en el que participan las universidades de Alberta, Calgary y Athabasca en cooperación con BELLE ("Broadband Enabled Lifelong Learning Environment") y CANAIRE ("Canadian Network for the Advancement of Research in Industry and Education"). El repositorio de objetos educativos CAREO es un prototipo en constante desarrollo. Como ocurría con el anterior, también es posible inspeccionar su catálogo o realizar búsquedas sobre el material que posee. En la Figura 3.6 se puede ver cómo sería la interfaz de este repositorio.

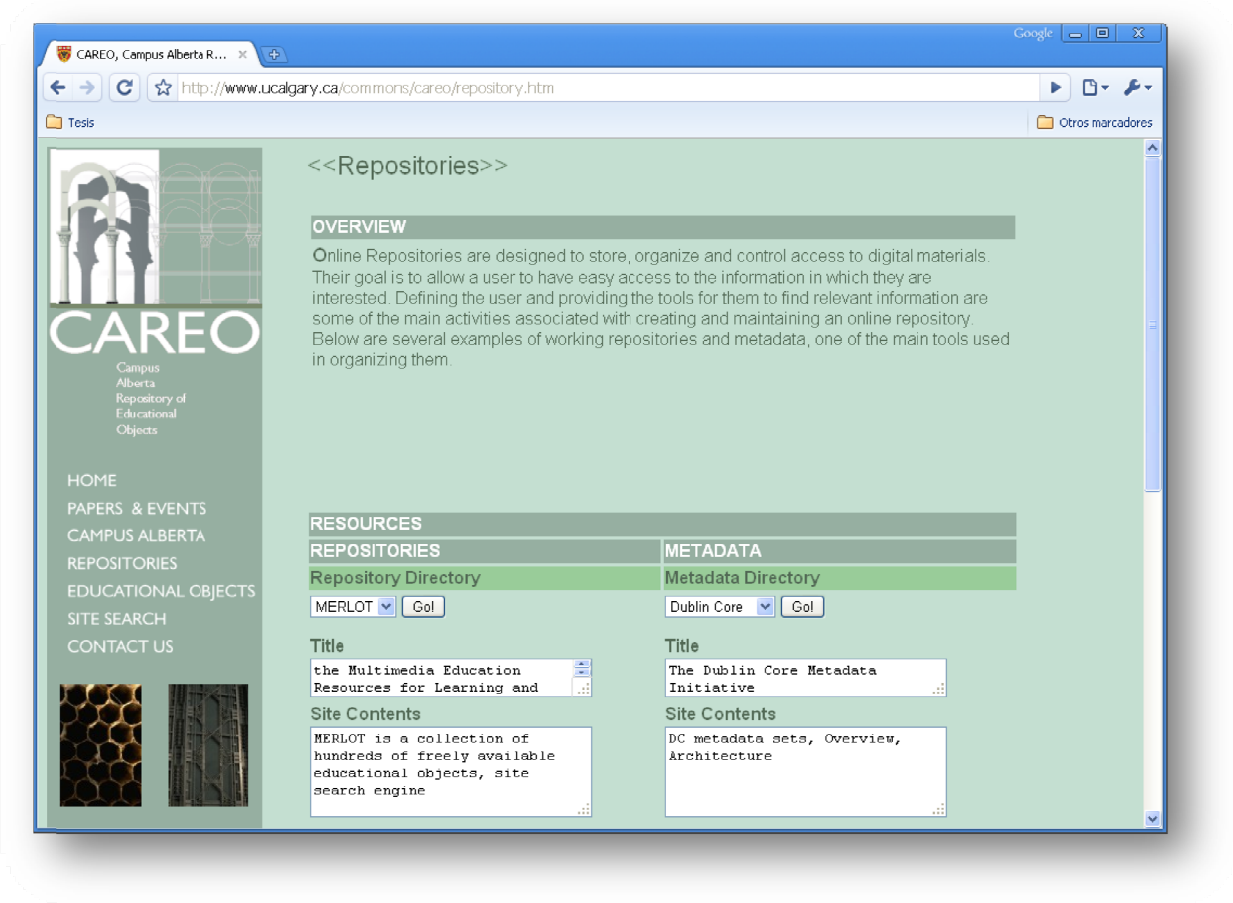


Figura 3.6 Repositorio CAREO

Se puede ver cómo sería a arquitectura conceptual del repositorio CAREO en la Figura 3.7. Está basada en una estructura de tres capas, un modelo cliente-servidor y está formada por tres componentes principales:

- **La aplicación del repositorio**: esta aplicación realiza para los usuarios y administradores la búsqueda y acceso en el almacén de meta-datos (en la capa de base de datos). Presenta a los usuarios una interfaz HTML para la búsqueda y acceso a los registros de meta-datos que describen y enlazan los objetos de aprendizaje. Para los administradores, la aplicación presenta una interfaz HTML para el mantenimiento de funciones (proporcionando control de calidad de los registros de meta-datos y gestión de funciones particulares del repositorio).
- **Los clientes o usuarios**: que pueden subdividirse en tres grupos principales: estudiantes, profesores y administradores. Los usuarios acceden a los metadatos a través del repositorio CAREO, pero accederán a los objetos educativos distribuidos a través de un servicio Web común. Exceptuando la conectividad en Internet, la única tecnología requerida por este grupo será un navegador Web que cumpla los estándares generales.
- **El almacén de metadatos**: que está localizado en el mismo servidor en el que se encuentra la aplicación del repositorio. Está estructurado de acuerdo con el protocolo de metadatos de objetos de aprendizaje *CanCore* que está basado en el LOM.

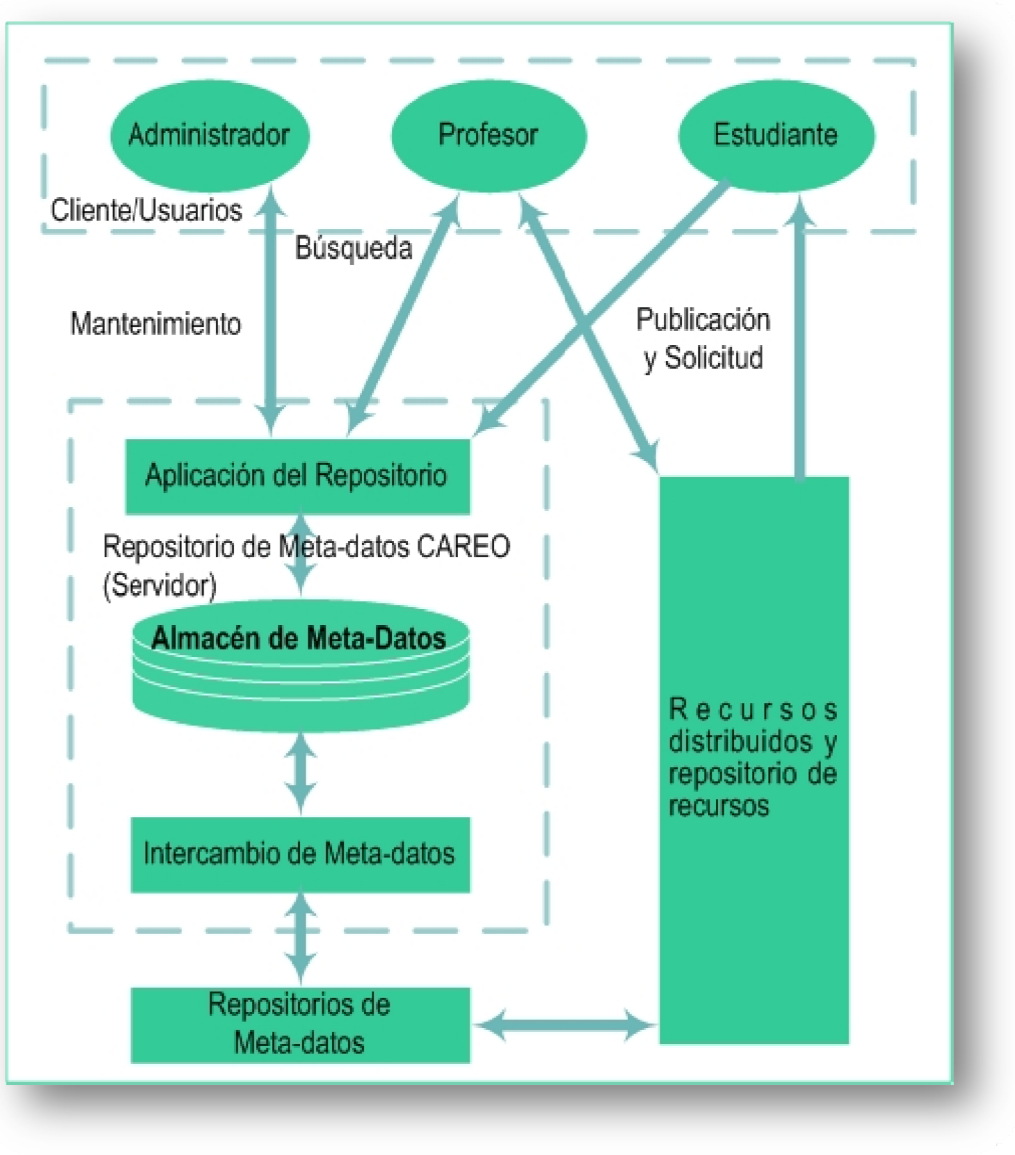


Figura 3.7 Arquitectura conceptual del repositorio CAREO

El objetivo de CAREO es que los recursos u objetos de aprendizaje se distribuyan por toda la red a través de servidores. Un recurso creado por un profesor, por ejemplo, se colocará en el servidor Web de la institución para la que esté trabajando dicho profesor. Los metadatos que describen dicho recurso se enviarán por profesor directamente al repositorio CAREO, o bien CAREO los recogerá de otro repositorio o almacén de repositorios. CAREO también será capaz de recoger los metadatos de los repositorios que contengan ambos elementos: metadatos y recursos, empleando para ello los protocolos aceptados.

Finalmente el componente para el intercambio de metadatos del repositorio CAREO recopilará la colección de registros disponibles en otros repositorios, o almacenes de metadatos. Este componente de intercambio de metadatos es capaz de emprender las funciones básicas de mapear e interpretar, para asegurar que los metadatos recopilados se ciñan al protocolo CanCore. Este componente además proporciona mecanismos para la creación de metadatos en el repositorio CAREO, estos metadatos serán accesibles por otros repositorios de metadatos que empleen los mismos protocolos que CAREO emplea para la recopilación de metadatos.

La diferencia más apreciable con MERLOT es que, en este caso, los metadatos que posee cada objeto de aprendizaje están mucho más detallados, y muchos de los objetos que posee los podemos descargar en un fichero comprimido, pero en el cual no se encuentran sus metadatos. Por lo tanto, adolece del mismo problema que MERLOT, ya que para poder integrar alguno de los objetos de aprendizaje en otro repositorio se necesitaría, por un lado descargar el contenido, y por otro copiar sus metadatos adaptándolos al estándar que se utilice y ensamblarlo todo.

3.3.3. ARIADNE - KNOWLEDGE POOL SYSTEM (KPS)

Es el repositorio empleado en un proyecto de la Unión Europea destinado a la creación de metodologías e instrumentos para la producción y gestión de material didáctico electrónico, así como sistemas e-learning encargados de su gestión, llamado ARIADNE (Alliance of Remote Instructional Authoring and Distribution Network for Europe) [ARIADNE, 2009]. Se trata de una red europea de recursos educativos distribuidos, alrededor de la cual se han creado una serie de herramientas que ayudan a la compartición y reutilización del material educativo. Se puede ver en la Figura 3.8 el aspecto de la interfaz de este repositorio.

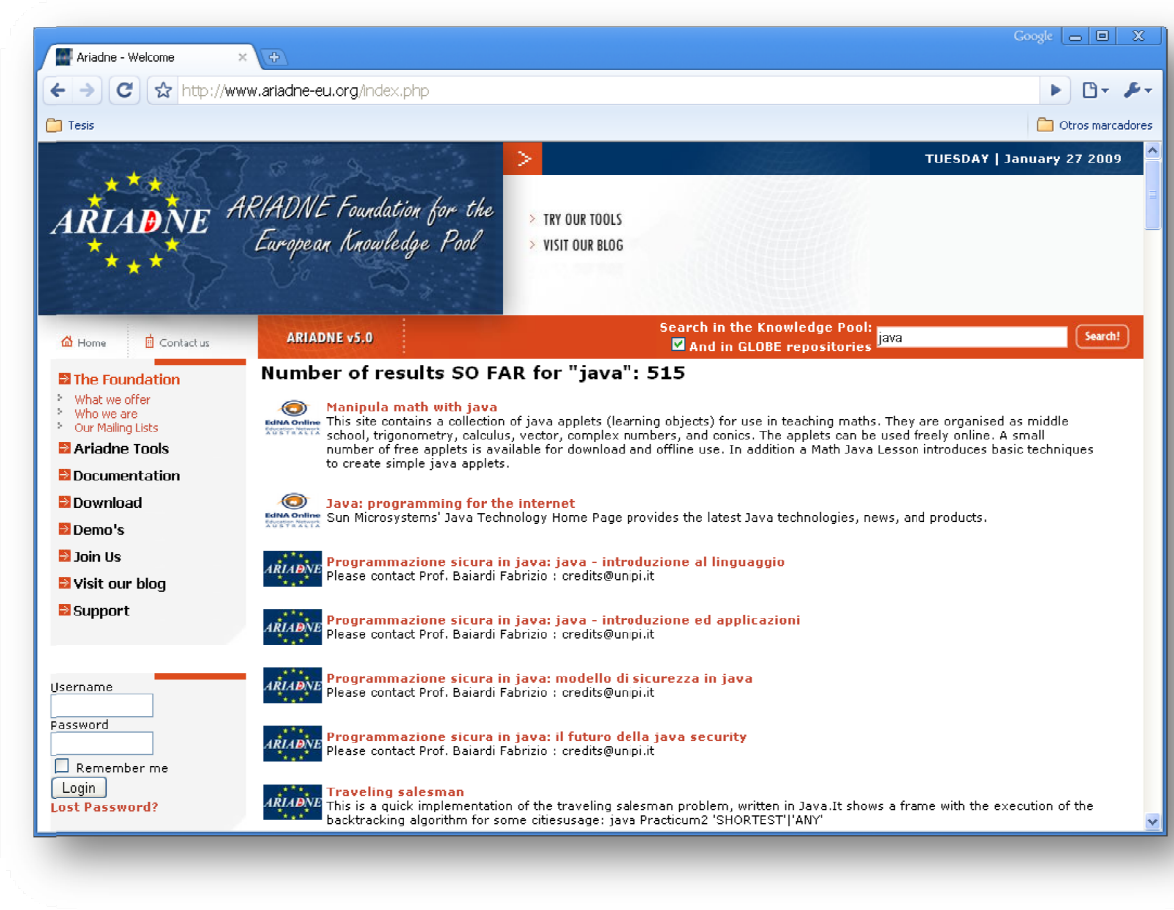


Figura 3.8 Repositorio ARIADNE

A continuación se detallan las principales características de la arquitectura de ARIADNE, que se detallan en la Figura 3.9:

- El núcleo central del proyecto es la distribución de los datos contenidos en un repositorio con componentes digitales reutilizables para la educación, llamado KPS (Knowledge Pool System).
- El gestor de KPS forma parte de la capa de datos, en la cual se almacenan los objetos de aprendizaje y los metadatos que los describen.
- El acceso al KPS para disponer de los datos, así como la gestión de cursos, se hace mediante las herramientas SILO (Search and Index Learning Objects) y WEBLE (Web-Based Learning Environment).
- KPS Client: Es la herramienta de acceso al KPS.

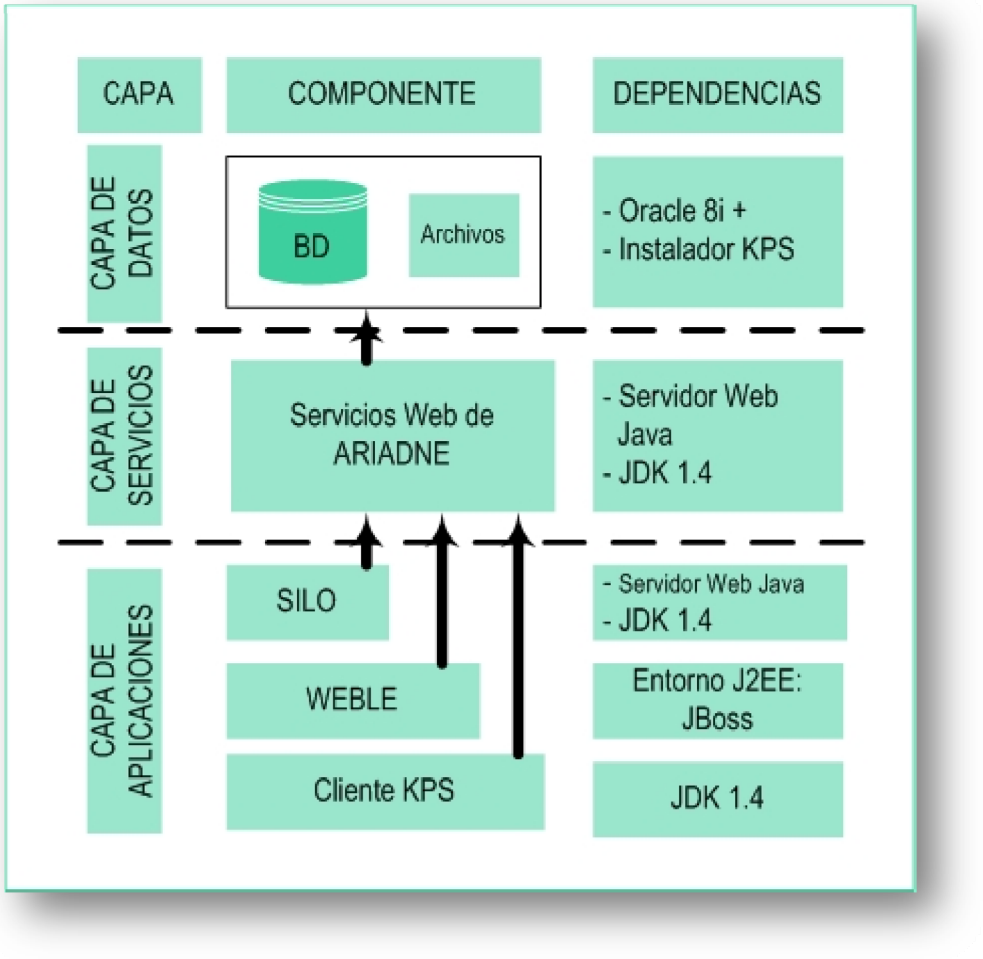


Figura 3.9 Arquitectura de ARIADNE

La principal ventaja que presenta ARIADNE es la posibilidad de hacer búsquedas en otros repositorios externos, una de las características que se integran también en la propuesta de esta Tesis. Inicialmente este era el único sistema que presentaba estas características, aunque actualmente tanto MERLOT como CAREO también presentan

búsquedas federadas. Estas búsquedas, llamadas "búsquedas federadas", permiten realizarlas de forma transparente al usuario en todos aquellos repositorios "compatibles" entre sí (más adelante se detallarán las premisas que deberán cumplir para estar dotados de dicha compatibilidad). Con esto se consigue una reutilización mayor de los objetos de aprendizaje ya que las búsquedas no solo se realizan en un repositorio. Se puede ver en la Figura 3.10 cómo se realizaría una búsqueda federada en los diferentes repositorios, mostrando en su parte izquierda los resultados ofrecidos por cada repositorio distribuido consultado.

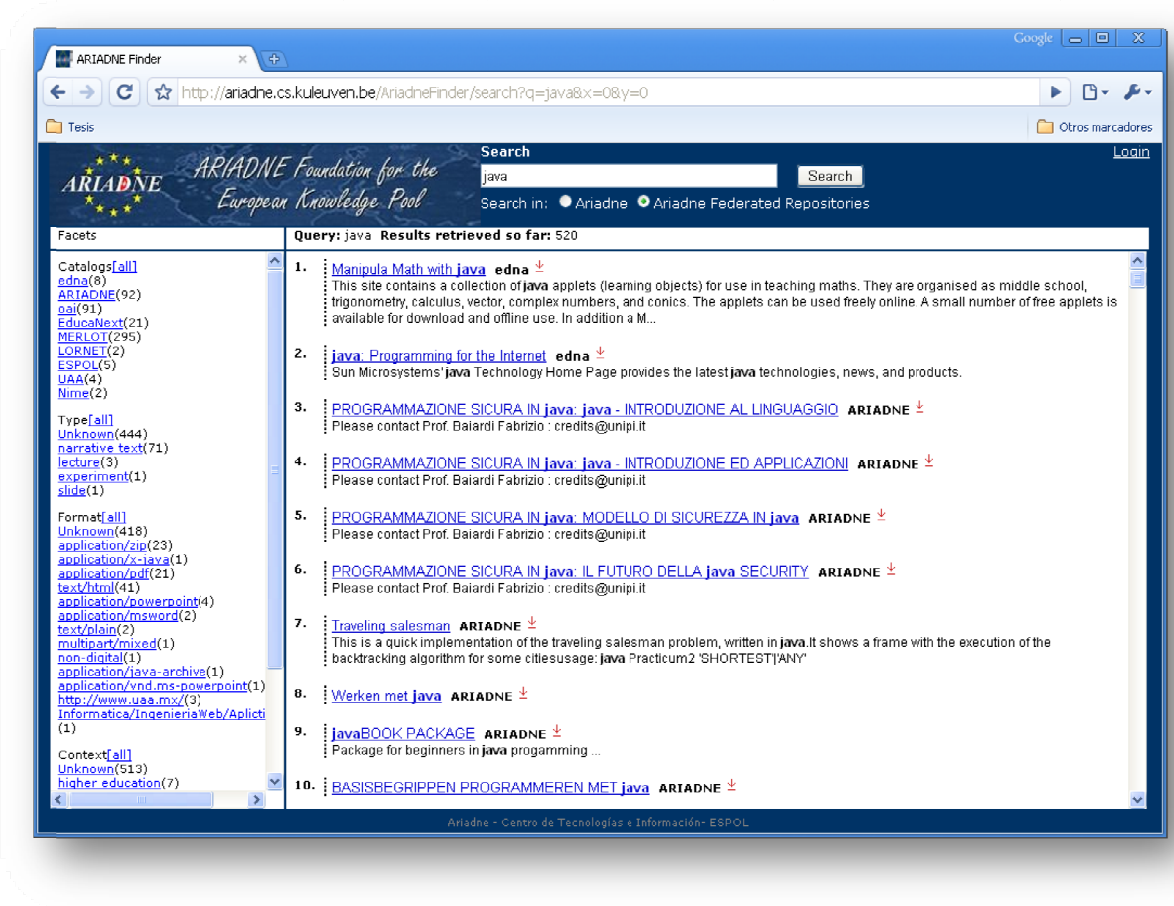


Figura 3.10 Búsquedas federadas en ARIADNE

Para realizar las búsquedas federadas se utiliza una interfaz de consulta llamada SQI (Simple Query Interface), ya explicado en el capítulo anterior. Gracias a esta interfaz de consulta es posible que cualquier repositorio se comunique con cualquier otro independientemente de protocolos, sistemas o estándares soportados, de forma que se garantice la interoperabilidad entre todos ellos. Como ya se comentó anteriormente ARIADNE presenta un listado de los principales repositorios y sistemas de búsqueda que

admiten este tipo de interfaz de consulta, indicando para todos ellos las rutas hacia sus servicios de consulta.

Con respecto a la descripción de los objetos de aprendizaje que almacena, al igual que ocurría en CAREO, la completitud de sus metadatos es bastante correcta (en este caso presenta una versión reducida de LOM). La principal ventaja que presenta con respecto a éste es que no solo permite la descarga del contenido sino que además permite exportar sus metadatos en LOM. Con esto se consigue que la reutilización de los objetos docentes sea muy grande, el único inconveniente es que el usuario final tenga que realizar una transformación de LOM al tipo de metadatos que utilice su repositorio. Se muestra en la Figura 3.11 la interfaz de descarga de material educativo que ofrece ARIADNE.

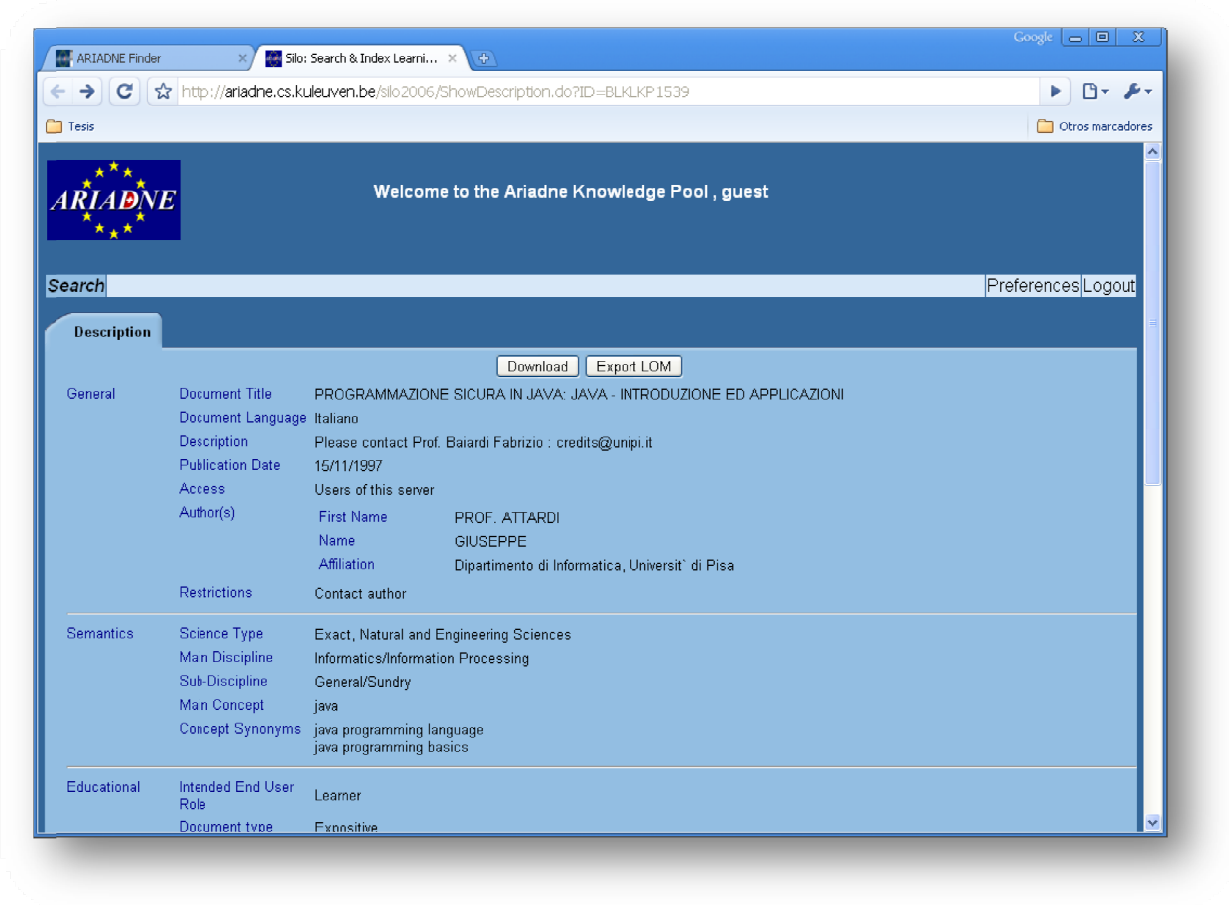


Figura 3.11 Metadatos de un LO en ARIADNE

3.3.4. GLOBE y CORDRA

El consorcio GLOBE [2009] está formado por ARIADNE, Education Network Australia (EdNA Online), eduSourceCanada, MERLOT y NIME (National Institute of Multimedia Education). Estas organizaciones colaboran para poder tener un acceso ubicuo a recursos educativos de calidad.

Los primeros pasos que el consorcio está llevando a cabo es el desarrollo de casos de uso, especificaciones, tecnologías y reglas de negocio que permitan la búsqueda de objetos de aprendizaje a través de los repositorios que los miembros de la alianza han desarrollado. Como se podía apreciar en ARIADNE se trata de expandir la búsqueda federada a más de un repositorio. A continuación se presenta en la Figura 3.12 cómo sería la interfaz de búsqueda de GLOBE, en la que se puede ver que se hacen consultas en los repositorios de ARIADNE, LORNET, MERLOT entre otros.

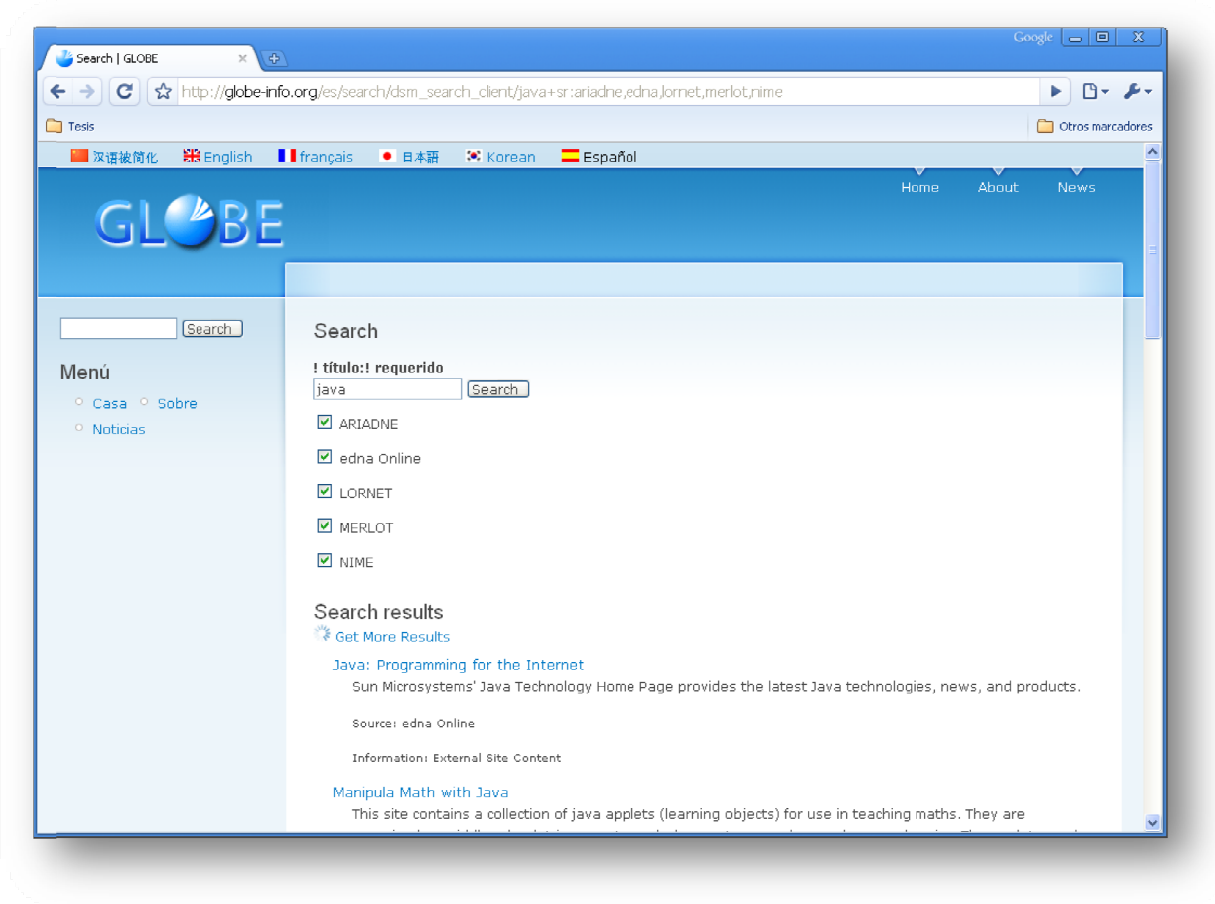


Figura 3.12 Búsqueda Federada en GLOBE

CORDRA (Content Object Repository Discovery and Registration/Resolution Architecture) [CORDRA, 2009] es un modelo abierto y basado en estándares para diseñar e implementar sistemas software destinados al descubrimiento, compartición y reutilización de material docente a través de repositorios interoperables. Este modelo hace de puente entre los materiales docentes y los repositorios. Trata de identificar y especificar (no desarrollar) las tecnologías apropiadas y los estándares de interoperabilidad para combinarlas en un modelo de referencia.

Por lo tanto, CORDRA es un modelo formal que puede ser usado para el diseño de repositorios federados y su interoperabilidad. Las actividades de CORDRA se coordinan entre ADL (*Advanced Distributed Learning Initiative*), CNRI (*Corporation for National Research Initiatives*) y LSAL (*Learning Systems Architecture Lab*).

Los componentes del modelo CORDRA, mostrados en la Figura 3.13, se pueden resumir en los siguientes:

- **Local Content Repositories**: Representan los repositorios locales donde se encuentra el material docente y su metainformación asociada.
- **System Repositories**: Integrado por los sistemas de CORDRA entre los que se encuentran el registro de repositorios y sistemas y el catálogo maestro.
- **Identifier System**: Infraestructura para la identificación de los distintos objetos de aprendizaje registrados a los que se puede acceder.
- **Common Services Infrastructure**: Servicios del núcleo del sistema y servicios administrativos usados para la implementación del modelo, por ejemplo, autentificación, derechos de autor, reglas de procesamiento, etc.
- **Applications**: Aplicaciones e interfaces usados para el manejo de los objetos de aprendizaje por los usuarios finales tales como búsqueda, registro, entrega, etc.

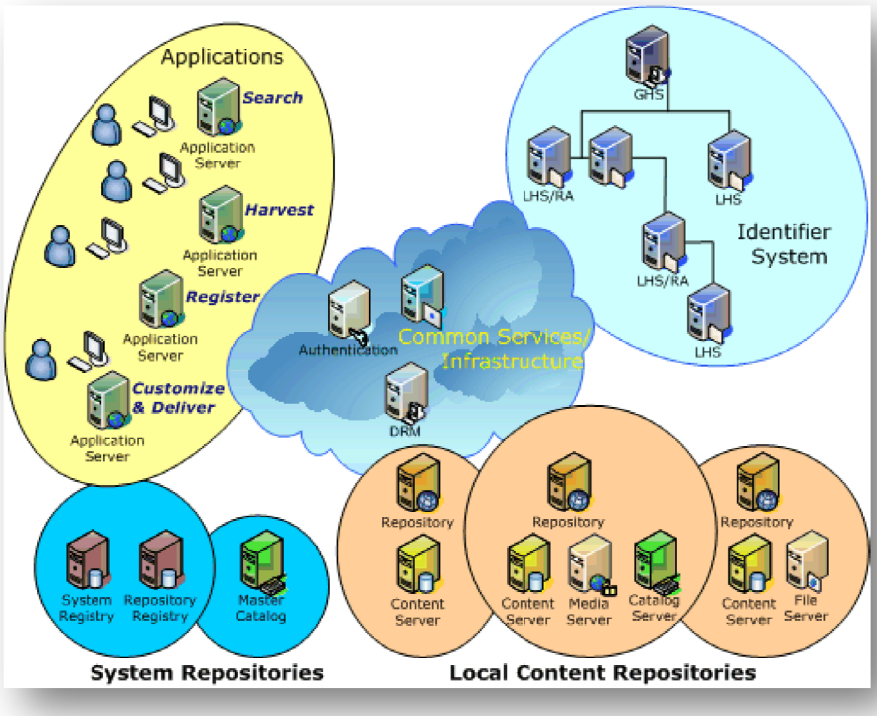


Figura 3.13 El modelo CORDRA [2009]

El proceso general de uso podría ser el siguiente: Un repositorio que contiene material docente con metainformación y unos métodos de gestión propios que quiere pertenecer al sistema federado CORDRA, debe inscribirse en el registro de repositorios para permitir su descubrimiento, acceso y administración. Los registros federados mantienen un catálogo maestro con los metadatos de todos los objetos de aprendizaje que contienen. Mediante este catálogo se pueden realizar las búsquedas de los objetos de aprendizaje y se mantiene un índice de los mismos para ayudar a realizar dichas búsquedas. El sistema de identificación permite la identificación, registro y definición de los objetos de aprendizaje. Mediante un conjunto de servicios comunes se da acceso a toda la funcionalidad del sistema. Por último, mediante una interfaz se proporcionan todos estos servicios a los usuarios que pueden acceder o publicar los objetos docentes.

Aunque las propuestas de GLOBE y CORDRA son las mejores soluciones a los problemas de la publicación y descubrimiento de los objetos de aprendizaje contenidos en un repositorio, en realidad se tratan sólo de modelos propuestos en los cuales no se entra en detalle en cuanto a cómo construir sistemas que se adapten al mismo. La única referencia clara es la de ARIADNE que sí detalla algo más sus búsquedas federadas e incluso publica las herramientas necesarias para montar un sistema similar.

CORDRA es el modelo más detallado y definido en el que se explica con claridad el diseño a seguir, pero no como llevarlo a cabo. ADL ha implementando parte del modelo y ha construido un registro de repositorios llamado ADL-Registry [ADL-R, 2009] que permite la publicación y búsqueda de objetos de aprendizaje. Se muestra en la Figura 3.14 el aspecto de su interfaz de consulta.

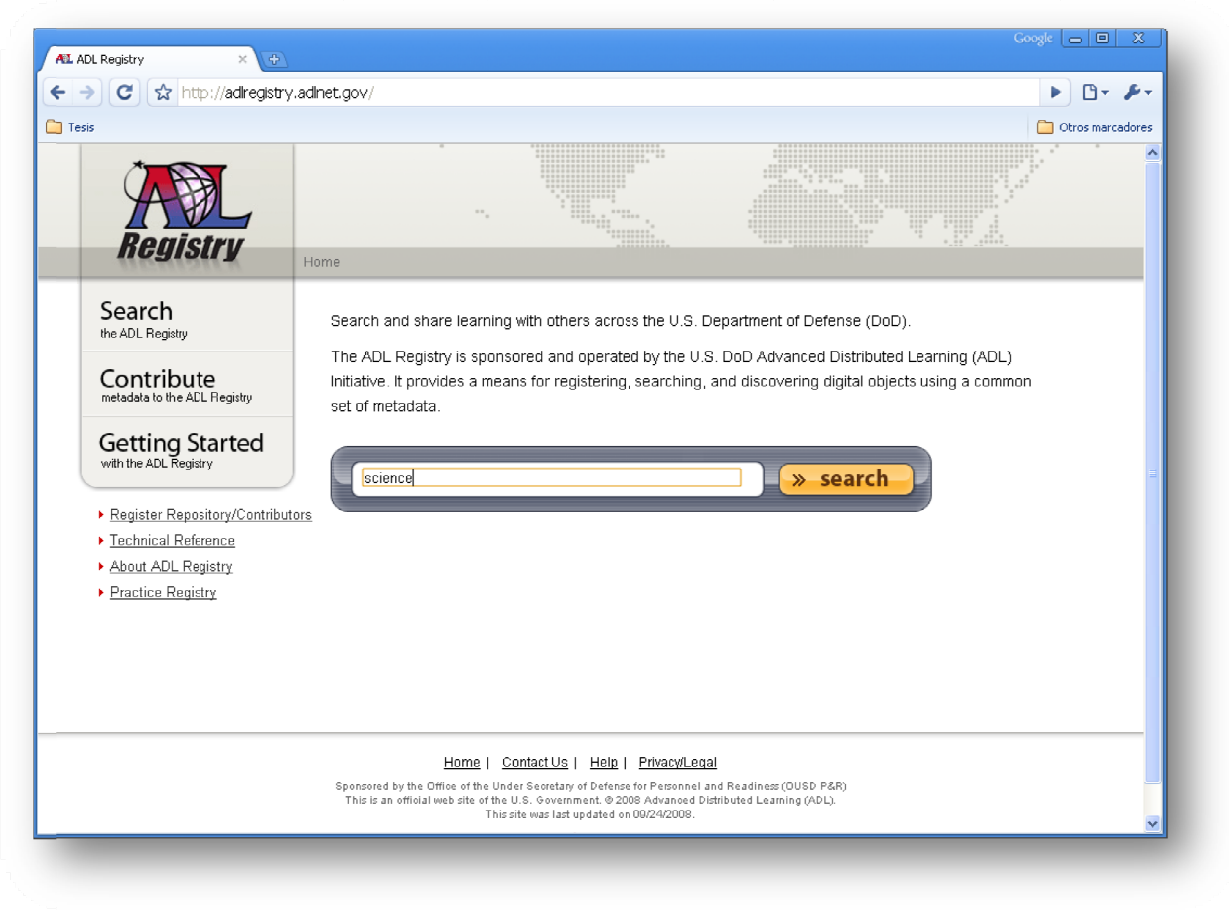


Figura 3.14 Sistema de búsqueda ADL-R

3.3.5. Conclusiones sobre los repositorios y sistemas de búsqueda analizados

Como conclusión a este apartado, se indicarán algunas de las principales diferencias que se han encontrado entre este modelo, y el que se va a presentar en esta Tesis.

En primer lugar resaltar que todos los repositorios analizados anteriormente permiten a fecha de hoy realizar búsquedas federadas en otros repositorios, gracias a lo cual se mejora en gran medida la reutilización del material educativo que albergan, así, en base a esta premisa se muestra en la Tabla 3.1 las relaciones existentes entre los repositorios.

		DESTINO DE LAS BÚSQUEDAS				
ORIGEN DE LAS BÚSQUEDAS		MERLOT	CAREO	ARIADNE	GLOBE	ADL-R
	MERLOT			✓		
	CAREO	✓				
	ARIADNE	✓				
	GLOBE	✓		✓		
	ADL-R					

Tabla 3.1 Comparativa entre los repositorios analizados (origen y búsquedas)

Con esta tabla no se indica que estos repositorios busquen solamente en los que se ha marcado en la tabla, sino que la tabla muestra solamente las interrelaciones entre ellos. Como característica adicional comentar que del repositorio ADL-R no se ha podido obtener información alguna acerca de en qué repositorios se realiza la búsqueda, sin embargo se asume que realiza búsquedas federadas ya que permite el alta de nuevos repositorios.

En lo que a los sistemas de búsqueda se refiere se presenta la Tabla 3.2 en la que se puede observar el tipo de búsqueda que admite cada uno de los repositorios, siendo la sencilla aquella que permite un solo término y avanzada aquella que permite buscar por los campos de una especificación de metadatos (LOM generalmente).

Repositorio	Búsqueda Sencilla	Búsqueda Avanzada
MERLOT	✓	
CAREO	✓	
ARIADNE	✓	✓
GLOBE	✓	
ADL-R	✓	✓

Tabla 3.2 Comparativa entre los repositorios analizados (tipos de búsquedas)

En lo que al cumplimiento de SQI [CEN, 2005] se refiere, se deben analizar los servicios Web ofrecidos por los diferentes repositorios considerados, confirmando qué métodos SQI han sido integrados en ellos, y se debe probar el funcionamiento de dichos métodos. Esto puede hacerse fácilmente desarrollando una aplicación que acceda a los servicios Web y realice las llamadas a los métodos correspondientes. Existen programas de libre distribución que pueden ayudar en esta tarea, como SQITest (http://ariadne.cs.kuleuven.ac.be/SQI/SQITest.jnlp), o también pueden utilizarse sistemas de búsqueda federada de objetos de aprendizaje en repositorios distribuidos con interfaz SQI, como Ariadne Search & Indexation tool (http://ariadne.cs.kuleuven.be/silo2006/NewFederatedQuery.do), que permiten al usuario seleccionar los repositorios remotos a los que extender una búsqueda.

A partir del análisis de los anteriores repositorios, se han obtenidos los resultados indicados en la Tabla 3.3. Como puede observarse, se presenta una relación de los repositorios que poseen los métodos de configuración de SQI, los de gestión de la sesión, los de consulta síncrona y los de asíncrona. Indicar que de los repositorios (o sistemas de búsqueda) GLOBE y ADL-R no se ha conseguido información.

Repositorio	Métodos de Configuración	Métodos de Gestión de la Sesión	Métodos de Consulta Síncrona	Métodos de Consulta Asíncrona
MERLOT	✓		✓	✓
CAREO	✓		✓	
ARIADNE	✓	✓	✓	✓

Tabla 3.3 Comparativa entre los repositorios analizados (cumplimiento de SQI)

Finalmente, en la Tabla 3.4 se ofrece un resumen de los resultados obtenidos, siendo importante resaltar que sólo uno de los repositorios analizados (ARIADNE) implementa el 100% de los métodos establecidos por la especificación SQI, es decir trece métodos. Los otros repositorios ofrecen solamente 10 métodos SQI, por tanto se puede decir que cumplen la especificación en un 77%.

Repositorio	Conf	G. SES	Sysn	Asyn	%
MERLOT	4	0	3	3	10 (77%)
CAREO	4	0	3	0	10 (77%)
ARIADNE	4	3	3	3	13 (100%)

Tabla 3.4 Comparativa entre los repositorios analizados (cumplimiento de SQI en porcentajes)

Por último indicar que en lo referente a la utilización de técnicas semánticas, solamente ARIADNE realiza tareas al respecto. Dentro de su herramienta de consulta SILO (Search & Index Learning Objects) tiene un visor de ontologías creadas para varias ramas. En la Figura 3.15 se muestra el aspecto de dicha interfaz.

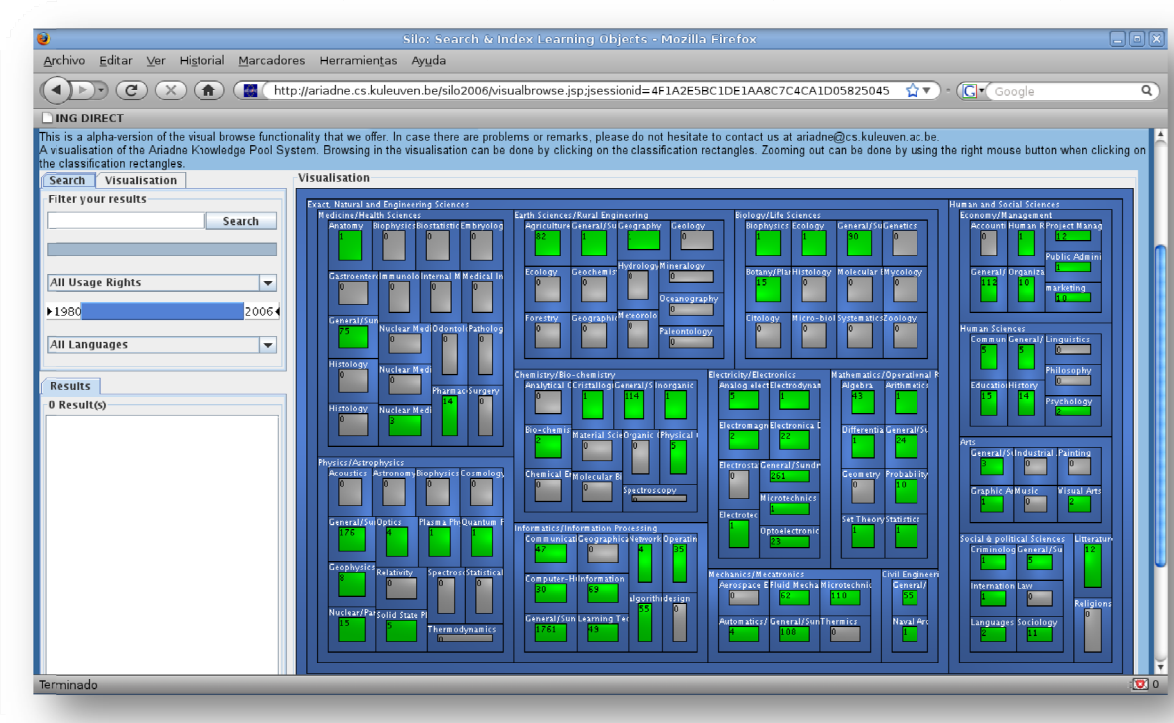


Figura 3.15 Interfaz de consulta semántica en ARIADNE

A través de dicho visor (realizado con un Applet Java) se puede ir navegando por los diferentes conceptos simplemente haciendo clic en ellos. Al final se mostrarán los objetos de aprendizaje relacionados con dichos conceptos.

Por otro lado, como característica principal de CORDRA recordar que se trata de una arquitectura, no de un desarrollo, por eso no ha sido incluida en las anteriores relaciones. Cuenta con un gran catálogo, en el que se indexaría el contenido de todos los repositorios. Es decir, se indexarían todos los objetos de aprendizaje, de todos y cada uno de los repositorios sobre los que se realizarían las futuras búsquedas. A priori, esta tarea parece bastante pesada para una arquitectura, ya que se plantean algunas preguntas: ¿cada cuánto tiempo habrá que reorganizar estos índices?, ¿qué tiempo puede suponer el realizar una búsqueda a través de un índice que crece exponencialmente?, ¿qué ocurre cuando se añaden nuevos objetos de aprendizaje o se modifican?, ¿es necesario reorganizar estos índices cada vez que se añadan nuevos objetos o nuevos repositorios? Esta tarea puede acabar siendo un gran cuello de botella para las búsquedas del sistema. Por último indicar que el repositorio ADL-R se realizó siguiendo esta arquitectura.

Sin embargo, la solución planteada en esta Tesis, así como el prototipo del que parte que será también explicado en el capítulo siguiente, es mucho más eficiente en este aspecto, ya que será un servicio Web asociado al repositorio el que se encargue de buscar objetos de aprendizaje en ese repositorio. El utilizar indexaciones para realizar las búsquedas puede resultar muy eficiente, pero en nuestro caso, se hace de forma interna en cada repositorio, y así se evita la necesidad de tener que indexar todos los objetos de aprendizaje de todos los repositorios. La arquitectura presentada, no cuenta con ninguna estructura en la que se almacenen todos los contenidos docentes de todos los repositorios que se van a analizar, ya que la cantidad de datos con la que se tendría que trabajar sería completamente excesiva.

Otra de las diferencias que se pueden observar es la utilización de estándares; es evidente que para que un objeto de aprendizaje sea reutilizable, debe crearse conforme a la definición de algún estándar, y por lo tanto el contenido del objeto debe estar descrito mediante metadatos. Pero una de las principales características con las que cuenta nuestra arquitectura, es que no está ligada a ningún estándar, por lo que es completamente válida, independientemente del estándar escogido. Sin embargo el modelo CORDRA está muy ligado a ADL, y con ello a SCORM, por lo tanto aquí se plantea otra duda, ¿qué ocurre con aquellos repositorios en los que se utilicen otros estándares? ¿Y si el usuario utiliza una plataforma (LMS) incompatible con SCORM? Desde el punto de vista del autor, una herramienta que busque una reutilización universal de objetos de aprendizaje, no debería estar ligada a un estándar determinado; más aún, cuando existen en el mercado varios de ellos, y no existe una unidad en cuanto a su utilización y total compatibilidad. Por ello se encuentra en este apartado otro punto a favor de la arquitectura presentada.

También es necesario hacer referencia a la posibilidad de descarga del material educativo. En todos los casos analizados los sistemas de búsqueda y los repositorios enlazan con la página del autor, y será a través de ésta, de la que se podrán descargar los objetos de aprendizaje (muchas veces sin su metainformación). Estos enlaces en muchas ocasiones cambian y se vuelven ilocalizables, por lo que no parece a priori una buena solución. En el caso de la propuesta realizada en esta Tesis, es el propio repositorio el que proporciona directamente el objeto de aprendizaje junto con su metainformación, garantizando en todo momento la accesibilidad al material educativo que es lo que persigue este tipo de arquitecturas y sistemas.

Por último comentar una característica que no posee ninguno de los repositorios y sistemas analizados y que si presenta la solución planteada en esta Tesis. Se trata de la posibilidad de confeccionar cursos completos a partir de la composición de objetos de aprendizaje de diferentes repositorios distribuidos. Así por ejemplo si un usuario quisiera un curso de Java básico, el sistema presentado buscaría en la ontología adecuada para determinar qué es lo que debería incluir un curso básico de Java, y a partir de esta información, determinará con qué objetos de aprendizaje contará, realizando la oportuna búsqueda de todos ellos en los diferentes repositorios distribuidos a los que tenga acceso. Una vez que haya descargado todos ellos, podrá componer un curso completo y proporcionárselo al usuario. Esta utilidad reviste de gran complejidad para el sistema, pero que garantizará aún más la reutilización de material educativo y con ello favorecerá el uso de este tipo de técnicas como apoyo a la formación.

3.4. DEFINICIÓN DE ARQUITECTURA Y SU COMETIDO

Antes de realizar una propuesta de arquitectura, es importante tener claro lo que se entiende precisamente por “arquitectura de un sistema software”. Para ello se utilizará una metáfora, comparando la arquitectura del software con la arquitectura de edificios. Un edificio es habitualmente una sola unidad desde la perspectiva del cliente. El arquitecto del edificio puede encontrar útil hacer una maqueta a escala del edificio, junto con los dibujos del edificio visto desde distintas perspectivas. Como un edificio, un sistema software es una única entidad, pero al arquitecto de software y a los desarrolladores les resulta útil presentar el sistema desde diferentes perspectivas para comprender mejor el diseño. Estas perspectivas son vistas del modelo del sistema. Todas las vistas juntas representan la arquitectura.

La arquitectura software abarca decisiones importantes sobre:

- La organización del sistema software.
- Los elementos estructurales que compondrán el sistema y sus interfaces, junto con sus comportamientos, tal y como se especifican en las colaboraciones entre estos elementos.
- La composición de los elementos estructurales y del comportamiento en subsistemas progresivamente más grandes.
- El estilo de la arquitectura que guía esta organización: los elementos y sus interfaces, sus colaboraciones y su composición.

Sin embargo, la arquitectura software está afectada no sólo por la estructura y el comportamiento, sino también por el uso, la funcionalidad, el rendimiento, la flexibilidad, la reutilización, la facilidad de comprensión, las restricciones y compromisos económicos y tecnológicos, y la estética.

Un sistema grande y complejo requiere una arquitectura para que los desarrolladores puedan progresar hasta tener una visión común. Un sistema software es difícil de abarcar visualmente porque no existe en un mundo de tres dimensiones. Suele utilizar tecnología poco probada o una mezcla de tecnologías nuevas. Tampoco es raro que el sistema lleve a sus últimos límites la tecnología existente. Además, debe ser construido para acomodar gran cantidad de componentes que sufrirán cambios futuros.

A medida que los sistemas se hacen más complejos, los problemas de diseño van más allá de los algoritmos y las estructuras de datos para su computación.

Se necesita una arquitectura de sistema para:

- Comprender el sistema
- Organizar el desarrollo
- Fomentar la reutilización
- Hacer evolucionar el sistema
- La utilización del sistema con un rendimiento óptimo

Si hay algo de lo que se puede estar seguro, es que cualquier sistema de un tamaño considerable evolucionará. Evolucionará incluso aunque aún esté en desarrollo. Más tarde, cuando esté en uso, el entorno cambiante provocará futuras evoluciones. Hasta que esto ocurra, el sistema debe ser fácil de modificar, esto quiere decir que los desarrolladores deberán ser capaces de modificar partes del diseño o implementación sin tener que preocuparse por los efectos inesperados que puedan tener repercusión en el sistema. En la mayoría de los casos, deberían ser capaces de implementar nuevas funcionalidades en el sistema sin tener que pensar en un impacto dramático en el diseño e implementación existentes. En otras palabras, el sistema debe ser en sí mismo flexible o tolerante a los cambios. Las arquitecturas del sistema pobres, suelen degradarse con el paso del tiempo y necesitan ser “parcheadas” hasta que al final no es posible actualizarlas con un coste razonable.

3.4.1. Definición de arquitectura

No existe una definición universalmente aceptada, del concepto de “Arquitectura”. Así, por ejemplo para Bass et al. [1997], la arquitectura de un sistema es la estructura del sistema que comprende sus componentes, las propiedades de estos componentes y las relaciones entre ellos.

Es decir, según estos autores:

- La arquitectura define componentes, pero define lo necesario para su interrelación no sus características propias.
- Un mismo sistema puede definirse con más de una arquitectura y no se especifica qué tipo de componentes son los que forman el sistema, por ejemplo, en los sistemas software no se especifica si los componentes son objetos, procesos, librerías, etc.
- Todos los sistemas tienen una arquitectura ya que todos están formados por componentes y tienen relaciones entre ellos.
- El funcionamiento de cada componente es parte de la arquitectura desde el punto de vista del resto de los componentes.

Para otros autores, como Jazayeri et al. [2000], la arquitectura de un sistema es un conjunto de conceptos y decisiones de diseño sobre la estructura del sistema, que deben ser tomadas en cuenta antes de su realización para satisfacer de una manera efectiva los requisitos de funcionamiento y calidad implícitos en el sistema.

Otros autores como [Shaw y Garlan, 1996], clasifican las vistas de estos sistemas en:

- **Modelo estructural**: Todos los sistemas software están compuestos de componentes y relaciones entre los componentes, incluyendo además aspectos como: configuración, estilo, requisitos, semántica, análisis y propiedades.
- **Modelo de entorno de trabajo**: Es similar al modelo estructural pero enfatiza la coherencia del sistema completo en oposición a la definición de los componentes singulares.
- **Modelo dinámico**: Se refiere a la calidad del sistema, entendiendo por dinámico la posibilidad de cambios en la configuración del sistema.

- **Modelo de procesos**: Centra la atención en la construcción de la arquitectura y los pasos o procesos involucrados en esa construcción.

Para Booch et al. [1999], la arquitectura es el conjunto de decisiones significativas acerca de la organización de un sistema, la selección de sus elementos estructurales y las interfaces que componen el sistema, junto con la colaboración entre los elementos estructurales y la descomposición en subsistemas según el funcionamiento de estos elementos. Por tanto, la arquitectura se compone de elementos que forman el sistema, sus interfaces, la colaboración entre ellos y su composición.

Para Boehm y Port [1998], una arquitectura de sistemas comprende:

- Un conjunto de componentes del sistema, conexiones y restricciones.
- Un conjunto de requisitos del sistema.
- Un razonamiento que demuestre que los componentes, sus conexiones y restricciones que definen el sistema, una vez implementados, satisfacen el conjunto de requisitos del sistema.

La Architecture Tradeoff Analysis [Kazman et al, 1998] proporcionó las siguientes definiciones desde distintos puntos de vista:

- Desde un punto de vista técnico, una arquitectura es el conjunto mínimo de reglas que gobiernan la interdependencia de las partes o elementos que juntos se usan para formar un sistema.
- Desde un punto de vista operativo, una arquitectura es una descripción, a veces gramatical, que define los elementos y las necesidades de intercambio de información entre ellos en cuanto a tipo de información y frecuencia de intercambio.
- Desde un punto de vista de sistemas, una arquitectura es una descripción, a menudo gramatical, de la solución utilizada para satisfacer los requisitos del sistema y los parámetros de medida de cumplimiento de requisitos.

Soni et al. [1995], también han definido una arquitectura desde distintos puntos de vista, según estos autores:

- Conceptualmente, una arquitectura describe un sistema en función de los principales elementos del diseño y las relaciones entre ellos.
- Modularmente, una arquitectura establece la descomposición funcional del sistema y sus distintos niveles.
- Desde un punto de vista ejecutivo, una arquitectura debe describir la capacidad y estructura dinámica del sistema.
- La codificación de una arquitectura detalla cómo están organizados los elementos que forma el entorno de desarrollo del sistema.

IEEE en su glosario estándar sobre Ingeniería del Software define el término arquitectura como "la estructura organizacional de un sistema o componente" y el término "diseño arquitectural" como "el resultado del proceso de definición de una colección de componentes hardware y software y sus interfaces para establecer el framework de desarrollo de un sistema" [IEEE, 1990].

También IEEE, en su estándar 1471, ofrece otra definición más detallada de arquitectura como "la organización fundamental de un sistema expresado en sus componentes, relaciones con los demás componentes y con el entorno, y los principios que guían su diseño y evolución" [IEEE, 2000]. En este estándar se indica que la descripción de la arquitectura de un sistema debe:

- Especificar los participantes en el sistema.
- Identificar los objetivos del sistema en términos de:
  - Funcionalidad, planteando la cuestión de qué necesita hacer el sistema.
  - Rendimiento, considerando cómo se comportará el sistema ante situaciones extremas de carga de trabajo.
  - Seguridad, teniendo en cuenta si el sistema puede proteger adecuadamente la información del usuario.
  - Viabilidad, planteándose si realmente se puede implementar el sistema.
- Organizarse en una o más vistas de la misma arquitectura.
- Ofrecer alternativas para que los arquitectos puedan tomar decisiones.

3.4.2. Objetivos de una arquitectura

Según Joe Batman, del Software Engineering Institute [Batman, 1999], los objetivos que debe cumplir una arquitectura efectiva son los siguientes:

- La arquitectura se debe usar como un plan prescriptivo de la construcción del sistema, estructurando su resultado.
- Mediante la arquitectura debe ser fácil identificar los modelos estructurales del dominio de aplicación, intentando proporcionar la solución más simple.
- La arquitectura debe identificar las partes del sistema, ordenándolo mediante su descomposición.
- La arquitectura debe aislar los componentes del sistema, presentando las instrucciones y líneas de acción necesarias para llegar al detalle de la descomposición.
- La arquitectura debe identificar las relaciones con sistemas externos.
- El entorno de desarrollo del sistema debe estar soportado por la arquitectura y ésta debe facilitar su conocimiento, en cuanto a servicios y utilidades.
- La arquitectura debe soportar los incrementos adicionales de capacidad del sistema.
- La arquitectura debe utilizar estándares para uniformizar problemas y soluciones en cuanto a nomenclatura, estilo, planificación, procedimientos, etc.
- Ya que las excepciones son inevitables, la arquitectura debe anticipar lo necesario para cubrirlas.
- La arquitectura debe proporcionar documentación adecuada.
- Para el uso efectivo de la arquitectura debe realizarse entrenamientos y dar soporte a los cambios culturales que conlleva.
- El control de calidad debe estar integrado en la arquitectura.
- La arquitectura debe proporcionar propiedades estructurales y de calidad que permitan un proceso de prueba eficiente.

Allamaraju [1998], cuando realiza un estudio de la arquitectura Paradox, afirma que un arquitecto para cumplir su cometido debe, en una primera fase, basándose en los requisitos del sistema diseñar un esbozo de la arquitectura y establecer los objetivos que debe cumplir para satisfacer estos requisitos, a continuación, establecer los principios que debe seguir para cumplir los objetivos definidos e integrar esos principios en el diseño de la arquitectura y, por último, evaluar y redefinir de acuerdo a los resultados la arquitectura para todos los casos de uso, midiendo parámetros de facilidad de adaptación al tamaño de los sistemas, prestaciones, integración, etc.

3.4.3. Características de una arquitectura

Vistos los objetivos que debe tender una arquitectura, a continuación se detallan qué características debe cumplir para poder lograr estos objetivos.

Según el Grupo de Ingeniería del Software de la Universidad de Málaga [UMA, 2009] es muy importante la componibilidad, es decir, si sus componentes presentan comportamientos compatibles y pueden ser combinados de forma segura para formar un sistema. Además se asegura la reutilización si se comprueba que un cierto componente puede ser utilizado en un nuevo sistema donde se necesita un comportamiento similar, incidiendo en la compatibilidad de los componentes. Estas características conducirán a una arquitectura a poseer propiedades relativas a consistencia y operatividad.

Allamaraju [1998] aprecia como esenciales las siguientes características:

- Una arquitectura debe representar un punto de vista elevado de los sistemas que revele su estructura, pero esconder todos los detalles de implementación.
- Una arquitectura debe representar todos los posibles casos de uso. Esto garantiza que la estructura representada por la arquitectura responda a todos los requerimientos funcionales.

Según Batman [1999], las características que debe cumplir una arquitectura efectiva son las siguientes:

- Debe ser reutilizable y exportable, para lo cual debe cubrir los requerimientos del dominio entero.
- Debe ser compacta, es decir debe ser lo suficientemente flexible para adaptarse sin comprometer la integridad conceptual.
- Debe tener un alto nivel de abstracción.
- Debe estar bien documentada, de forma clara y no ambigua.

Para el SEI (The Software Engineering Institute), que ha trabajado en la ATA (Architecture Tradeoff Analysis) [Kazman et al., 1998] con el objetivo de establecer técnicas válidas para representar arquitecturas y promover su conocimiento, los atributos de calidad que debe tener una arquitectura son [Barbacci et al., 1997]:

- Rendimiento
- Posibilidad de Modificación
- Disponibilidad
- Seguridad
- Interoperabilidad

Garlan y Perry [1995], creen que las principales características de la configuración de una arquitectura se dividen en tres categorías generales:

- **Cualidades de la descripción de la configuración**: especificaciones comprensibles, modularidad, refinamiento y trazabilidad y heterogeneidad.
- **Cualidades del sistema descrito**: heterogeneidad, escalas diversas, evolución y dinamismo.
- **Propiedades del sistema descrito**: dinamismo, restricciones y propiedades no funcionales.

A continuación se describe la definición de cada una de estas características:

- **Especificaciones comprensibles**: Uno de los papeles principales de una arquitectura es servir de vía de comunicación entre los distintos componentes de un proyecto y facilitar el entendimiento del sistema a un nivel alto de abstracción.
- **Modularidad**: La modularidad o descomposición jerárquica permite describir los sistemas según diferentes niveles de detalle; por ejemplo, se pueden representar estructuras complejas como un único componente del sistema.
- **Refinamiento y trazabilidad**: La arquitectura debe poder ser revisada y corregida en distintas fases de refinamiento, y debe ser posible seguir los cambios realizados en cada una de esas fases.
- **Heterogeneidad**: Un objetivo de cualquier arquitectura es facilitar el desarrollo de sistemas a gran escala, posiblemente integrados por componentes y conectores de tipos distintos y soportados por entornos diferentes.
- **Escalas diversas**: La arquitectura intenta proporcionar a sus usuarios herramientas válidas para tratar sistemas de complejidad y tamaño diversos.
- **Evolución**: La arquitectura debe evolucionar y así poder reflejar los cambios de los sistemas, ya que sus componentes y conectores se incrementan, se modifican y cambian sus relaciones.

- **Dinamismo**: El término evolución refleja los cambios externos de una arquitectura (y sus sistemas resultantes); en cambio, dinamismo se refiere a la modificación de una arquitectura y sus sistemas resultantes mientras éstos se están ejecutando; son sistemas que cambian dinámicamente y la arquitectura que los soporta debe estar basada en una evolución run-time. De este tipo son los sistemas de control de tráfico y de redes telefónicas.
- **Restricciones**: Son dependencias de la configuración completa debido a restricciones en los componentes o conectores individuales. La seguridad y prestaciones de un sistema completo siempre depende de la seguridad y prestaciones de sus componentes.
- **Propiedades no funcionales**: Son propiedades referentes al sistema completo y no a sus componentes ni conectores. Se refieren a ayudas a la dirección del proyecto, análisis, restricciones de la dirección, etc.

Por su parte, OMG [2001], ha establecido que las características necesarias para que una arquitectura soporte la interoperabilidad e integración durante todo el ciclo de vida de un sistema, separando especificaciones de realización; estas son:

- Posibilidad de utilización con distintas tecnologías.
- Portabilidad a través de múltiples plataformas.
- Integración basada en un modelo de relaciones entre diferentes dominios.

3.4.4. Factores que influyen en el diseño de una arquitectura

Existen diferentes tipos de factores que influyen en la arquitectura, como puede apreciarse en la Figura 3.16. En primer lugar se encuentran los casos de uso; un caso de uso es una secuencia típica de acciones en un sistema, desde el punto de vista del usuario, que muestra cómo el sistema interacciona con el exterior y que se obtiene como resultado del uso del mismo [Jacobson et al., 2000]. También son de ayuda en el diseño de una arquitectura la experiencia de trabajos anteriores y las estructuras que se puedan identificar como los patrones de la arquitectura.

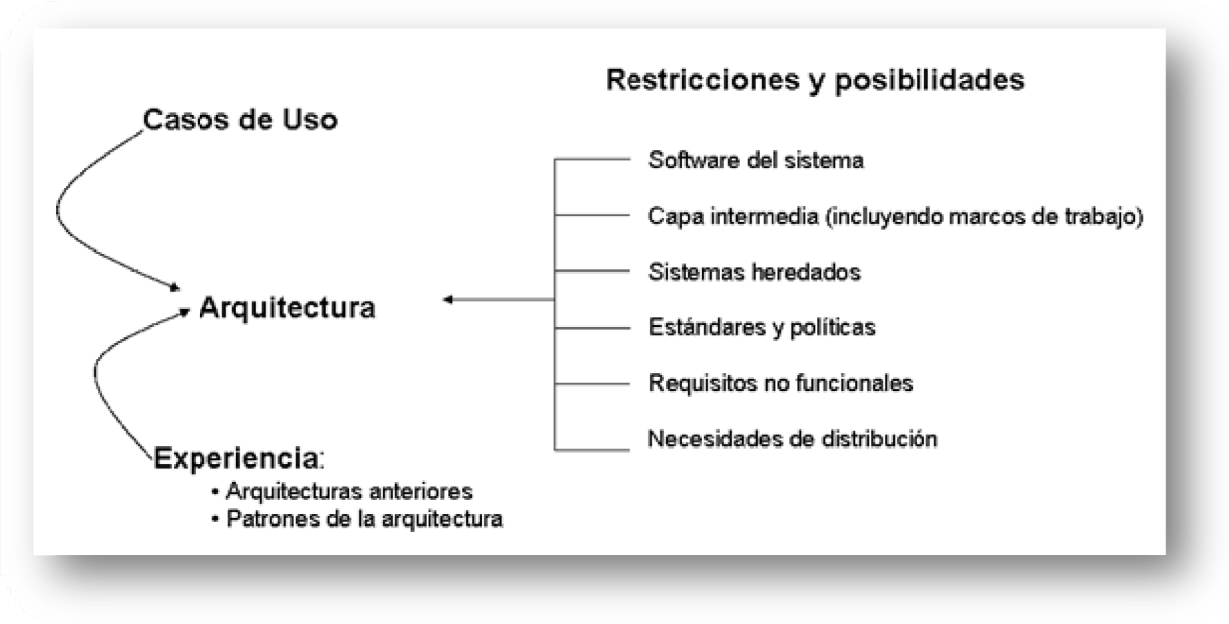


Figura 3.16 Factores que influyen en la arquitectura [Jacobson et al., 2000]

Si el sistema proporciona los casos de uso correctos, casos de uso de alto rendimiento, calidad y facilidad de utilización; los usuarios pueden emplearlo para llevar a cabo sus objetivos. Pero, la cuestión es: ¿cómo se puede conseguir esto? La respuesta es construir una arquitectura que permita implementar los casos de uso propuestos de una forma económica, ahora y en el futuro.

Sin embargo, la arquitectura no sólo se ve condicionada por los casos de uso arquitectónicamente significativos, sino también por los siguientes factores (Figura 3.16):

- Sobre qué productos software del sistema se quiere desarrollar, como sistemas operativos o sistemas de gestión de bases de datos concretos.

- Qué productos middleware (capa intermedia) se quieren utilizar. Por ejemplo, si se tiene que seleccionar un mecanismo para la conversión y envío de mensajes a objetos en entornos heterogéneos, o un marco de trabajo independiente de la plataforma, es decir, un subsistema “prefabricado”.
- Qué sistemas heredados se quiere utilizar en el sistema.
- A qué estándares y políticas corporativas hay que adaptarse.
- Qué requisitos no funcionales generales se deben satisfacer, como requisitos de disponibilidad, tiempo de recuperación o uso de memoria.
- Cuáles son las necesidades de distribución, que especifican cómo distribuir el sistema, quizá a través de una arquitectura cliente/servidor.

3.4.5. Utilización de patrones en la arquitectura

Las ideas del arquitecto Christopher Alexander [Alexander et al., 1977] sobre cómo los "lenguajes de patrones" se utilizan para sistematizar principios y prácticas importantes en el diseño de edificios y comunidades, han inspirado a muchos desarrolladores, como se comentó en el Capítulo 2 (Estado del Arte). Los patrones de las arquitecturas se utilizan de una forma parecida, pero se centran en estructuras e interacciones de grano más grueso, entre subsistemas e incluso entre sistemas. Existen muchos patrones de arquitectura [Buschmann et al., 1996], pero se van a tratar los más interesantes en el ámbito de las arquitecturas orientadas a objetos, por ser en las que se basa la propuesta de esta Tesis.

- El patrón Broker es un mecanismo genérico para la gestión de objetos distribuidos [Gamma et al., 2000]. Permite que los objetos hagan llamadas a otros objetos remotos a través de un gestor que redirige la llamada al nodo y al proceso que guardan al objeto deseado. Esta redirección se hace de manera transparente, lo cual quiere decir que el llamante no necesita saber si el objeto llamado es remoto.
- El patrón Layers es aplicable a muchos tipos de sistemas [Gamma et al., 2000]. Este patrón define cómo organizar el modelo de diseño en capas, lo cual quiere decir que los componentes de una capa sólo pueden hacer referencia a componentes en capas inmediatamente inferiores. Este patrón es importante porque simplifica la comprensión y la organización del desarrollo de sistemas complejos, reduciendo las dependencias de forma que las capas más bajas no "son conscientes" de ningún detalle o interfaz de las superiores. Además, ayuda a identificar qué puede reutilizarse, y proporciona una estructura que ayuda a tomar decisiones sobre qué partes conseguir de terceros y qué partes construir.

Hay otros patrones que ayudan a comprender el hardware de los sistemas a construir y que ayudan a diseñar un sistema software sobre él, como por ejemplo *Client/Server*, *Three-Tier* y *Peer-to-Peer*. Estos patrones definen una estructura para el modelo de despliegue y sugieren cómo se deben asignar los componentes a los nodos [Gamma et al., 2000].

Como patrón arquitectónico también se podría considerar la estructuración de una arquitectura en capas. Una capa es un conjunto de subsistemas que comparten el mismo grado de generalidad y de volatilidad en las interfaces: las capas inferiores son de

aplicación general a varias aplicaciones y deben poseer interfaces más estables, mientras que las capas más altas son más dependientes de la aplicación y pueden tener interfaces menos estables.

Un sistema con una arquitectura en capas sitúa a los subsistemas de aplicación individuales en la capa más alta. Estos se construyen a partir de subsistemas en las capas más bajas, como son los marcos de trabajo (*frameworks*) y las bibliotecas de clases. Todas estas características se pueden observar en la Figura 3.17.

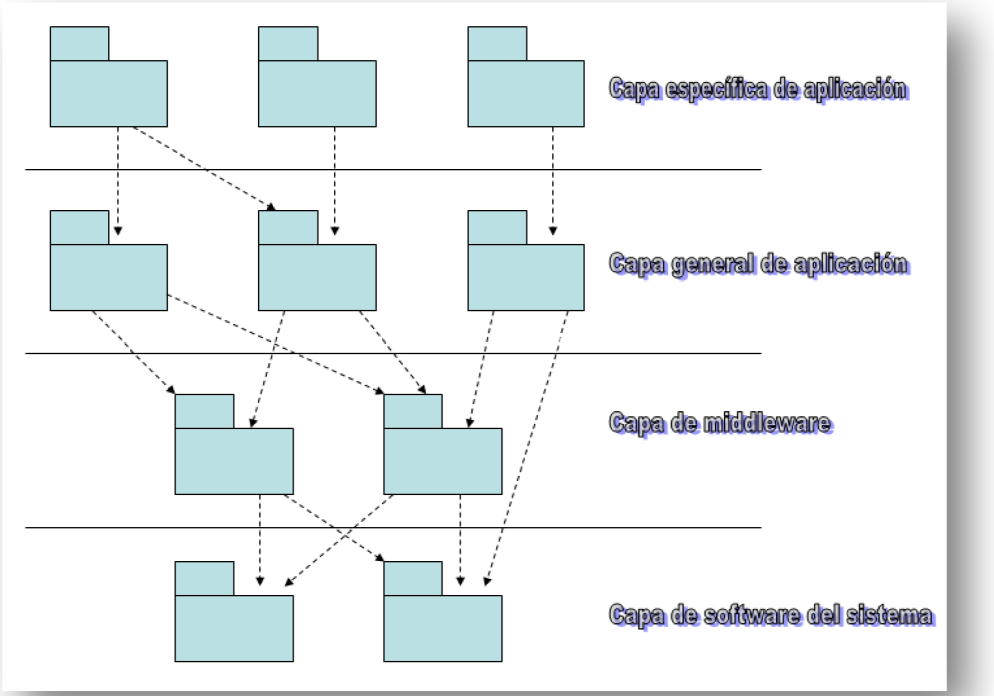


Figura 3.17 Arquitectura en capas

La capa general de aplicación contiene los subsistemas que no son específicos de una sola aplicación, sino que pueden ser reutilizados por muchas aplicaciones diferentes dentro del mismo dominio o negocio. La arquitectura de las dos capas inferiores puede establecerse sin considerar los requisitos o casos de uso debido a que no son dependientes del negocio. La arquitectura de las dos capas superiores se crea a partir de los casos de uso significativos para la arquitectura (estas capas son dependientes del negocio).

3.4.6. Conclusiones

Tras esta breve introducción a las arquitecturas, y antes de plantear el sistema de partida o la propuesta arquitectural que se presenta en esta Tesis es conveniente tratar de sintetizar las múltiples visiones hasta aquí descritas con la idea de presentar la arquitectura del sistema en forma multicapa o multinivel con los componentes que forman cada nivel, incluyendo:

- Los niveles o capas en los que se estructura la arquitectura del sistema.
- Los componentes o elementos estructurales.
- Las propiedades visibles de estos componentes y su funcionalidad.
- Las relaciones y colaboración entre ellos.

Con respecto a los objetivos de una arquitectura se puede resaltar lo siguiente:

- La arquitectura debe establecer los objetivos que ha de cumplir para satisfacer los requisitos del sistema e integrarlos en el diseño de la misma.
- La arquitectura debe identificar las partes del sistema, ordenándolo mediante su descomposición.
- La arquitectura debe identificar las relaciones con sistemas externos.
- La arquitectura debe utilizar estándares para uniformizar problemas y soluciones en cuanto a nomenclatura, estilo, planificación y procedimientos.
- La arquitectura debe soportar los incrementos adicionales de capacidad del sistema.

Con respecto a sus características se puede resaltar:

- La arquitectura debe representar todos los posibles casos de uso.
- Debe ser reutilizable y exportable.
- Debe tener un alto nivel de abstracción.
- Debe permitir la interoperabilidad (posibilidad de utilización con distintas tecnologías, portabilidad a través de múltiples plataformas).
- Debe ser heterogénea, facilitando la integración de componentes de distinto tipo.
- Debe ser evolutiva: La arquitectura debe evolucionar y así poder reflejar los cambios de los sistemas.

3.5. PUNTO DE PARTIDA DE LA TESIS: SISTEMA LORS

Como se ha comentado en diversos apartados de esta Tesis, las aplicaciones e-learning actuales deberían permitir un acceso universal a la información docente, garantizando una completa accesibilidad, independientemente de los estándares, protocolos o lenguajes de programación utilizados. De esta forma, se potenciará la reutilización universal de objetos de aprendizaje. En el mercado existen actualmente algunas herramientas de búsqueda que han sido desarrolladas para dar acceso a estos almacenes didácticos de objetos de aprendizaje (como se ha visto en los capítulos anteriores), pero sin embargo, ninguna de ellas permite un acceso global y flexible. Los creadores de material docente deberían permitir el acceso a sus contenidos de una forma sencilla, sin que se vieran en la obligación de cumplir una serie de restricciones. Para que esta característica se vea plasmada, es necesario que los desarrolladores de las herramientas de búsqueda diseñen productos lo más flexibles y adaptables posibles, ya que de lo contrario, esto supondría un gran problema a superar en el objetivo de la reutilización universal del material docente.

En este apartado se presenta una arquitectura orientada a servicios, e implementada mediante servicios Web, para el descubrimiento universal de objetos de aprendizaje almacenados en diferentes repositorios, que permite localizarlos independientemente de su ubicación física y de su tecnología de almacenamiento, y que sirve como punto de partida al posterior desarrollo del cometido de esta Tesis.

3.5.1. Introducción

Cada vez es más frecuente encontrar en Internet una gran cantidad de recursos educativos en formato digital, lo que permite su reutilización para mejorar la diversidad en los contenidos docentes empleados para la confección de acciones formativas. Generalmente estos recursos se encuentran dispersos en sistemas de e-learning, sistemas de gestión de contenidos y repositorios. Por lo tanto, surge la necesidad de realizar búsquedas distribuidas de recursos docentes almacenados en diversos "contenedores" educativos. Este tipo de búsquedas se denominan "búsquedas federadas".

Para poder dar acceso a esa gran cantidad de recursos se deben proporcionar una serie de herramientas que faciliten, por un lado, una búsqueda eficiente, y por otro, la facilidad para incorporar nuevos almacenes de recursos al sistema de búsqueda.

Sin embargo, se presentan una serie de problemas a resolver, ya que la mayoría de estos sistemas de almacenamiento emplean sus propios metadatos para la descripción de sus recursos educativos, lo que trae implícitamente la heterogeneidad en la información. Otro aspecto problemático es que cada sistema utiliza su propia tecnología de acceso y recuperación de recursos educativos, con lo que se debe diseñar un mecanismo lo más general posible, para que de esta forma, no interfiera en el sistema que tenga desarrollado el proveedor de recursos.

Para resolver estos problemas se plantean una serie de técnicas y sistemas que se pasan a describir en los siguientes apartados.

3.5.2. Arquitectura orientada a servicios para la localización y publicación de objetos de aprendizaje: Análisis y Diseño

La reutilización de contenidos docentes es una de las prioridades actuales en los entornos e-learning. De poco sirve un objeto de aprendizaje con un alto nivel de calidad, si solo es accesible por unos cuantos usuarios de una determinada plataforma o repositorio. Las instituciones educativas requieren mecanismos de interoperabilidad, ya que sería muy costoso quedar con contenido aislado en un mundo cada vez más interconectado y que clama por la colaboración institucional como mecanismo para garantizar una educación de calidad.

Para que un objeto de aprendizaje sea reutilizable, como se ha indicado en el anterior capítulo, debe ser accedido desde la mayor cantidad de plataformas e-learning posibles, así como por todos sus clientes potenciales. Para resolver los problemas anteriormente planteados, se realizó un sistema denominado SROA (Sistema de Reutilización de Objetos de Aprendizaje) o en sus siglas en inglés LORS (Learning Object Reusability System). Este sistema forma parte de un proyecto de investigación PROFIT "FIT-350101-2005-4", denominado "Sistema para la publicación y localización universal de objetos de aprendizaje", financiado por el Ministerio de Industria, Turismo y Comercio de España, en el que ha participado el autor de la Tesis.

Con este sistema se pretenden hacer interoperables un conjunto de repositorios distribuidos, consiguiendo así la reutilización de los objetos de aprendizaje que contienen. Por extensión, se puede incorporar sobre cualquier sistema de e-learning que quiera hacer accesible su contenido.

El sistema se ha construido utilizando una arquitectura basada en SOA e implementada mediante servicios Web Java. Ofrece a través de una única interfaz, un acceso transparente a los objetos almacenados en repositorios distribuidos, independientemente de sus tecnologías de almacenamiento, estándares de metainformación o su ubicación física, permitiendo así su completa reutilización y accesibilidad. Para su diseño y construcción, se han seguido las especificaciones marcadas por IMS a través de Digital Repositories Interoperability [IMS, 2003a] y Abstract Framework [IMS, 2003b] entre otras.

Un servicio Web es una colección de protocolos y estándares abiertos cuya misión principal es la de intercambiar datos entre aplicaciones, independientemente del lenguaje de programación, entorno o protocolos utilizados. La interoperabilidad se consigue mediante la adopción de estándares abiertos. Por su parte SOA (Service Oriented Architecture), es un tipo de arquitectura de software que define la utilización de servicios para dar soporte a los requerimientos de software del usuario. SOA proporciona una metodología y un marco de trabajo para documentar las capacidades de negocio, y puede dar soporte a las actividades de integración y consolidación.

Los servicios Web y las arquitecturas SOA, constituyen una tecnología muy adecuada para la implementación de repositorios que gestionen objetos de aprendizaje ubicados en diferentes almacenes de recursos didácticos, ya que permiten diseñar sistemas que garanticen una correcta e independiente forma de acceso a los recursos distribuidos.

En [Otón et al., 2005] se propuso la primera versión de la arquitectura que sustenta el sistema SROA, cuya versión más avanzada se puede ver en la Figura 3.18. A continuación se describen, de forma resumida, las cuatro capas de la arquitectura.

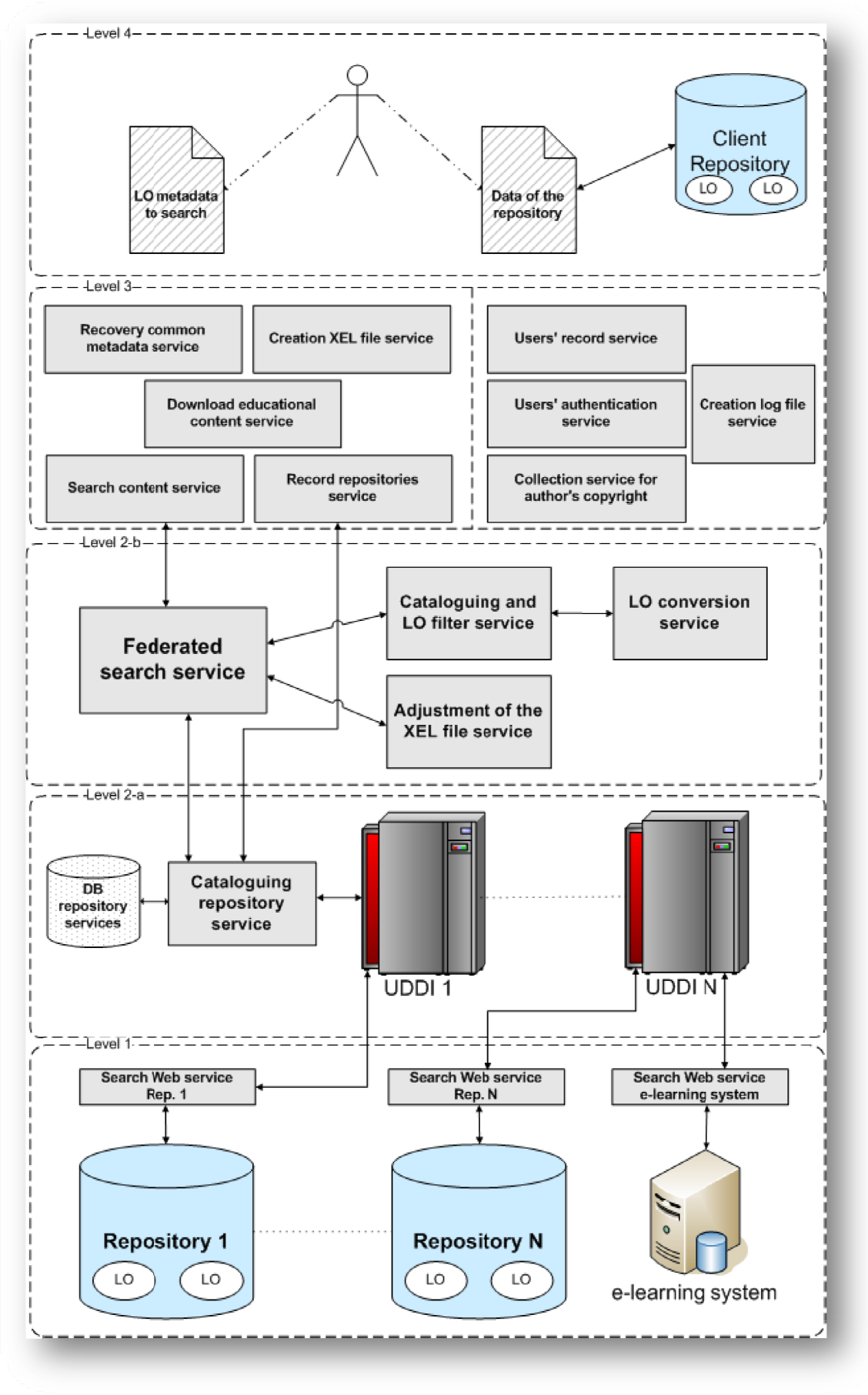


Figura 3.18 Arquitectura de LORS propuesta en 2005 [Otón et al., 2005]

Nivel 1. Repositorios Distribuidos

El objetivo de la arquitectura propuesta es hacer transparente el acceso a múltiples repositorios de objetos de aprendizaje (LO), por lo que el nivel más bajo estaría compuesto precisamente por todos estos repositorios que contienen los objetos.

La orientación a servicios se pone ya de manifiesto en este nivel para ocultar a las capas superiores los detalles de implementación de cada repositorio (como determina IMS), existiendo, al menos, un servicio Web asociado a cada repositorio, encargado de gestionar el acceso al mismo. Este servicio debería recibir un fichero XEL (eXtensible E-learning Language, fichero XML interno del sistema), el cual contiene la información de búsqueda proporcionada por el usuario, definida conforme al estándar de metadatos utilizado en los LO de su repositorio.

Para que dicho fichero XEL esté adaptado a la especificación utilizada por cada repositorio, SROA contará con un servicio específico de adaptación, que será utilizado por el servicio de búsqueda federada, como se detallará posteriormente. Este servicio asociado al repositorio, por lo tanto, realizará llamadas a métodos de búsqueda internos del repositorio y cuyo resultado serán los LO que contengan metadatos similares a los recibidos a través del fichero XEL.

Se debe puntualizar que los LO recuperados, en principio, tendrán el formato de metadatos estándar con el que trabaje el repositorio (IMS, SCORM, etc.), correspondiendo a las capas superiores de la arquitectura la función de conversión necesaria para su adaptación al formato deseado por el usuario. No obstante, en algunos casos, cuando los repositorios se hayan desarrollado siguiendo las recomendaciones de la especificación IMS DRI, existirá la posibilidad de que la conversión indicada se realice en esta capa, por parte del propio repositorio, como sugiere la propia especificación DRI [IMS, 2003a].

Una de las ventajas que tiene esta arquitectura, es que es totalmente adaptable a cualquier estándar actual o futuro, y por lo tanto, totalmente compatible con cualquier repositorio de objetos de aprendizaje. En SROA, será el usuario el que indique qué tipo de campos desea que se busquen en los ficheros de metainformación de los objetos de

aprendizaje. Esta solicitud se realizará a través de un fichero XSD, en el que se especificarán los campos educativos válidos para la búsqueda (pudiendo agruparlos por categorías o darle restricciones a cada campo). Será a partir de estos campos, con los que se construyan los ficheros XEL anteriormente comentados.

A continuación se muestra en la Figura 3.19, un gráfico explicativo sobre cómo sería el proceso de conversión del fichero XSD al fichero XEL.

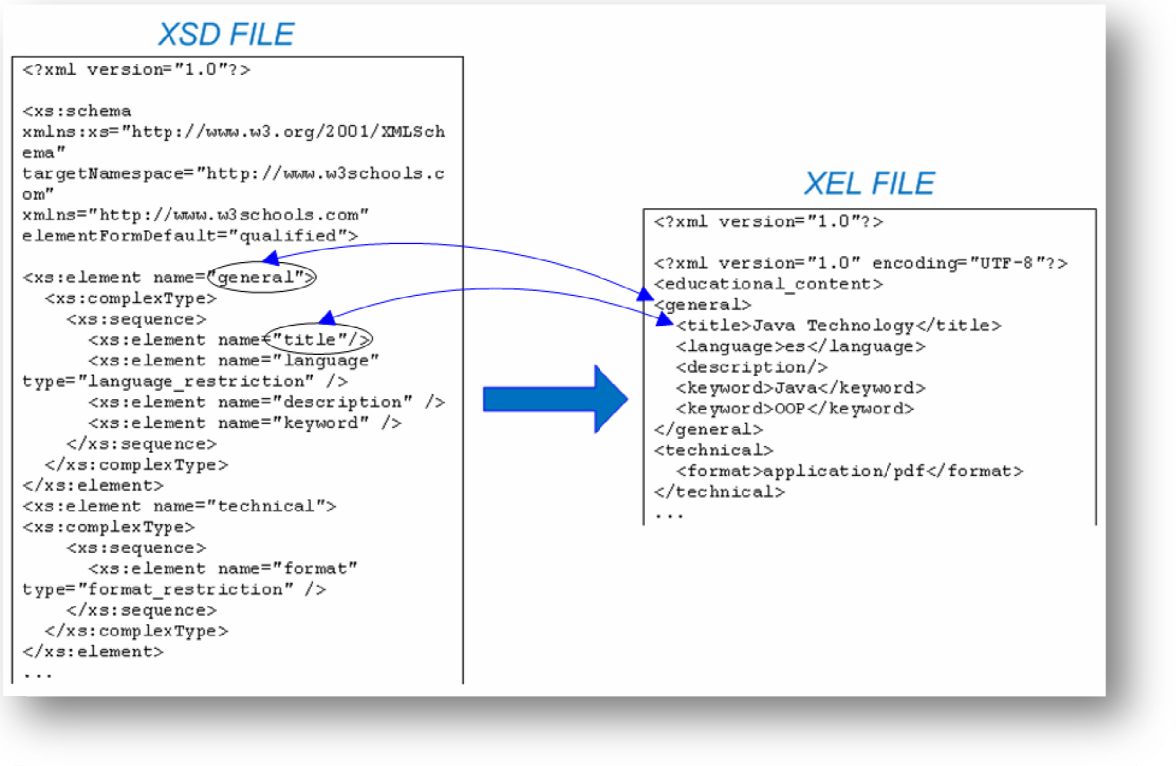


Figura 3.19 Sistema de obtención de los metadatos (XSD a XEL)

Además de utilizar para la generación de los ficheros XEL, el fichero XSD lo utiliza SROA, como se verá en la capa superior de la arquitectura, para generar un formulario de entrada totalmente dinámico, en el que el usuario introducirá los datos de búsqueda. Por lo tanto, el único cambio que tendría que hacer un usuario para adaptar el sistema SROA a otro conjunto de metadatos, sería modificar el fichero XSD, y se generarían de nuevo todos los ficheros, así como el formulario de entrada, sin que este cambio afecte a ni una sola línea de código del sistema.

El servicio Web asociado a cada repositorio, tendrá que, por un lado, devolver a SROA un listado con todos aquellos objetos de aprendizaje que se correspondan con los parámetros de búsqueda especificados por el usuario. Para realizar esta comparación, el servicio Web deberá llamar a operaciones propias del repositorio, que proporcionen los mecanismos de búsqueda y comparación de sus objetos de aprendizaje. Por otro lado, tendrá que devolver un fichero físico cuando el usuario lo solicite (una vez que recibe el listado, el usuario podrá seleccionar cada fichero para su descarga). Para la devolución de los diferentes objetos de aprendizaje, se utilizó SAAJ (SOAP with Attachments API for Java [SAAJ, 2009]). Por lo tanto, se enviará un mensaje con archivos adjuntos, que incorpore aquel objeto de aprendizaje que este le solicitó de entre todo el listado de objetos que se correspondían con sus parámetros de búsqueda.

Nivel 2. Capa de Interoperabilidad

Esta capa es la que relaciona los repositorios distribuidos con el sistema. Además de contener los servicios más complejos de procesado de información, se provee de un mecanismo de orquestación de servicios para que su ejecución se haga de una forma controlada y ordenada. Este nivel se puede subdividir en otros dos, según se muestra a continuación:

Nivel 2-a. Directorio de servicios

Una vez desarrollados los servicios de la capa anterior, éstos deben ser publicados en un directorio de servicios para su posterior localización.

Para que un repositorio sea localizable, debe publicar su servicio Web de acceso en un registro UDDI, de forma que si desde otro sistema se quiere acceder a sus objetos de aprendizaje, tan solo tendrá que localizar su registro en el UDDI y descargarse el WSDL (WSDL: Web Services Description Language), que describe el servicio. Por lo tanto, cuando un usuario quiera dar acceso exterior a un repositorio, lo primero que hará será publicarlo en un registro UDDI. Para añadir un nuevo repositorio será necesaria la información del UDDI donde está publicado y la propia del servicio. De esta necesidad, surge un servicio Web encargado de mantener una base de datos con un catálogo actualizado de todos los servicios de acceso a los repositorios registrados en el sistema;

de esta forma el usuario se podrá despreocupar de si los enlaces a los servicios pueden variar en el futuro. Esta funcionalidad será explicada en mayor profundidad en el nivel de presentación, en el que se muestran las funcionalidades que puede realizar un usuario con el sistema.

Nivel 2-b. Servicios de integración

En esta capa es donde se encuentran los servicios más complejos, ya que serán los encargados de realizar las búsquedas federadas en diversos repositorios. Además se encargarán de localizar los objetos de aprendizaje, a través de la localización del servicio asociado a cada repositorio. Una vez recuperados los listados de objetos de aprendizaje, deberán ser catalogados.

Antes de poder realizar el conjunto de esta búsqueda, será necesario conocer el tipo de especificación con la que trabaja cada repositorio configurado, para que de esta manera, se pueda adaptar el fichero XEL de metainformación a la especificación utilizada, y se pueda realizar así la búsqueda solicitada. Para realizar esta tarea, la arquitectura propuesta cuenta explícitamente con un servicio que realizará esta conversión en aquellos casos en los que sea necesario, este será el servicio de adaptación del fichero XEL.

Cuando finaliza la búsqueda federada sobre los distintos repositorios, se habrá obtenido un listado de objetos de aprendizaje que se deberán filtrar, transformar al estándar requerido por el cliente y ordenar por índice de coincidencia. Por lo tanto, existirá un servicio de catalogación encargado del filtrado y la ordenación, y otro servicio de conversión. La principal labor del servicio de catalogación es el filtrado, es decir, hacer una clasificación de todos los LO recibidos de forma que se eliminen los duplicados. La siguiente tarea que se realizará es, en su caso, la conversión de los objetos de aprendizaje al formato del estándar esperado por el cliente. Para ello se convertirán los metadatos de un estándar a otro (esta tarea solamente se realizará cuando el usuario decida descargar el objeto de aprendizaje).

Cuando se ejecuta un servicio de búsqueda de contenidos (explicado en la capa 3), en el sistema se desencadena la llamada a una serie de servicios. El servicio principal es el

servicio de búsqueda federada, que se encargará de realizar las llamadas a los distintos servicios de búsqueda de los repositorios a los que se tiene acceso.

Nivel 3. Servicios de Aplicación y Servicios Comunes

En esta capa residen los servicios de aplicación y comunes correspondientes a la funcionalidad de la arquitectura, por lo tanto se encuentran los servicios que el usuario invoca a través de la capa de acceso y presentación. Algunos de estos servicios realizan llamadas a los servicios de las capas inferiores. A continuación se comentan los principales servicios que incluye esta capa:

- **Servicios de Aplicación**:
    - Servicio de búsqueda de contenidos: Sin duda es el más importante y es el que desencadena todas las llamadas a los servicios de las capas inferiores. Es el encargado de recoger la información introducida por el usuario en el formulario, y hacerla llegar hasta el servicio de búsqueda federada en el formato deseado, recibiendo posteriormente una lista ya catalogada con todos los objetos que se corresponden con los parámetros indicados por el usuario.
    - Servicio de descarga de contenidos: Una vez se presenten los contenidos catalogados al usuario, este podrá descargarlos uno a uno.
    - Servicio de registro de repositorios: Este servicio se encarga de registrar un nuevo repositorio en el sistema.
    - Servicio de creación del fichero de metadatos: Este servicio se encarga de generar un fichero XEL con la información de metadatos que el usuario ha completado en el formulario de entrada para la realización de una búsqueda.
- **Servicios Comunes**:
    - Servicio de gestión de cobros por derechos de autor de los contenidos: Cuando algún contenido esté sujeto a derechos de autor en los cuales se deba pagar por su utilización, este servicio se encargará de establecer los mecanismos necesarios para realizar el cobro pertinente.
    - Servicio de registro de usuarios: Este servicio es el encargado de registrar a cada uno de los usuarios del sistema y establecer sus privilegios. Como datos básicos se tendrá el nombre de usuario y contraseña. Se podrán añadir cualquier dato extra que se pueda

utilizar con posterioridad, como una dirección de correo donde enviarle algún tipo de información.

- Servicio de autenticación de usuarios: Será el encargado de autentificar a los usuarios cuando quieren utilizar el sistema y establecer sus privilegios.

Nivel 4. Acceso y Presentación

En este nivel se describe cómo sería la interacción de un cliente con SROA. En primer lugar, el cliente se conectará a una interfaz Web, donde podrá realizar dos acciones fundamentales, que son la de realizar una búsqueda federada en el sistema o dar de alta un nuevo repositorio.

Para la acción de búsqueda federada, deberá cumplimentar los campos de un formulario con los metadatos de búsqueda. Partiendo de la base que el sistema cuenta con un fichero XSD externo, a la hora de generar el formulario de entrada de datos, también será necesario que se procese este fichero, para de esta manera saber cuáles son los campos que se le tienen que pedir al usuario en cada momento. A través de este sistema, se obtiene una arquitectura totalmente adaptable a cualquier conjunto de campos educativos, tan sólo modificando el fichero XSD externo.

Una vez completado el formulario, este pasará a ser procesado por un servicio Web encargado de confeccionar el fichero XEL (eXtensible E-learning Language) de metadatos, que como ya se ha comentado es la base para la realización de las búsquedas posteriores.

Este servicio, será el encargado de llamar al servicio Web de búsqueda federada de la capa 2 y desencadenará todo el proceso explicado anteriormente. Como resultado recibirá, ya clasificados, los objetos de aprendizaje que coincidan con el patrón de búsqueda, los cuales podrán formar parte del repositorio del cliente (por ejemplo, el de la plataforma LMS que esté utilizando).

La segunda de las funcionalidades que aporta la arquitectura a los usuarios, es la relacionada con la catalogación de repositorios. Cuando un usuario desee registrar un

nuevo repositorio al sistema, se conectará de igual forma a una interfaz sobre la cual podrá registrar su servicio de tres formas distintas: sobre un UDDI que tendrá predefinido "por defecto" el sistema, sobre el que solamente necesitará proporcionar la URL de su servicio. Sobre otro UDDI cualquiera, siendo en ese caso necesario que tenga una cuenta con permisos de registro sobre dicho UDDI. O por último, podrá indicar al sistema que ya tiene registrado el servicio sobre un UDDI, y solamente necesita que sea accesible, por lo que automáticamente se registrará ese enlace sobre la base de datos del sistema.

De esta manera un usuario podrá añadir fácilmente un nuevo repositorio al sistema, dotándole a la arquitectura de una escalabilidad importante, no sólo por poder incorporar nuevos repositorios, sino por la transparencia a la hora de hacerlo, ya que será posible incorporar un sistema completo de reutilización como el mostrado en este trabajo, como si fuera un servicio asociado a un repositorio, pudiendo crear una estructura tan compleja como se quiera.

Una descripción más detallada de todos los niveles y servicios del sistema se puede obtener de los documentos [Otón et al., 2006a], [Otón et al., 2006b] y [Otón et al., 2007].

3.5.3. Puesta en práctica de la arquitectura: Implementación

En este apartado, se presenta una implementación de la arquitectura explicada en el apartado anterior.

Para el desarrollo del sistema, se eligió la plataforma Java por ser considerada actualmente la más utilizada [TIOBE, 2009], además de estar perfectamente ligada al desarrollo Web. Al utilizar una arquitectura SOA, el sistema presentado puede ser completamente accesible desde cualquier otro lenguaje de programación, plataforma o sistema que utilice un protocolo de comunicación; además, posibilita la incorporación de nuevos servicios sin que por ello afecte a lo desarrollado hasta ahora.

Para el desarrollo de los diferentes servicios Web, se ha elegido una plataforma de libre distribución, que permite el despliegue en una máquina local de un registro UDDI, para poder realizar así todas las pruebas necesarias durante el desarrollo. Estas plataformas son Systinet Server y UDDI Registry, ambas herramientas de Systinet (Hewlett-Packard). Desde el año 2007 Systinet pasa a formar parte del grupo HP y dejan de actualizar su producto. Hoy en día (Marzo 2009), prácticamente no queda constancia del sistema, por lo que para futuras modificaciones de SROA fue necesaria la modificación de algunas partes del sistema, como se verá en el siguiente apartado.

La interfaz Web de la aplicación ha sido desplegada en el servidor de aplicaciones de mayor utilización a nivel empresarial, como es Apache Tomcat [Apache, 2009], que servirá como contenedor de Servlets/JSP de la aplicación. Además se integra con el servidor de servicios Web seleccionado. En cuanto al gestor de base de datos utilizado será MySQL [2009], ya que además de ser también de libre distribución, se trata de un gestor multiplataforma.

En la Figura 3.20 (izda) se pueden ver las dos acciones que se podrán realizar, como son la búsqueda federada y la catalogación de repositorios. A continuación se muestra cómo será el aspecto de esta interfaz:

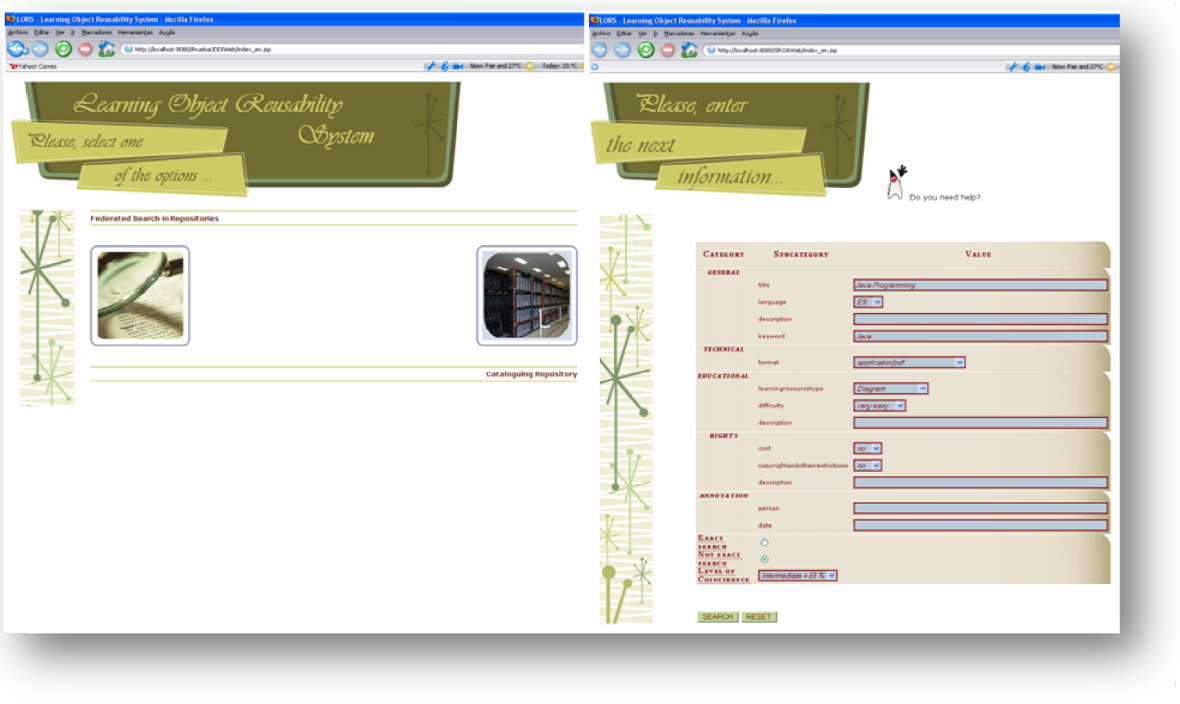


Figura 3.20 (izda.) Interfaz de entrada al sistema. (dcha.) Interfaz de entrada al sistema de búsqueda federada

Cuando el usuario inicie la búsqueda federada en repositorios, el sistema procesará el fichero XSD de campos educativos y generará el formulario de entrada de datos dinámicamente, mostrándole al usuario los datos que tendrá que rellenar, teniendo en cuenta las posibles restricciones marcadas para determinados campos. Además, le pedirá que especifique el porcentaje mínimo de coincidencia que tendrán que cumplir los distintos objetos de aprendizaje para ser considerados como válidos, así como el tipo de búsqueda solicitada de las dos posibles: Búsqueda Exacta, en la que se darán solamente por válidos aquellos campos que coincidan exactamente, o Búsqueda Inexacta, en la que además se darán por válidos aquellos campos que coincidan al menos en un 25% (parámetro configurable). En la Figura 3.20 (dcha), se mostró el resultado es esta interfaz.

Cuando el usuario rellene toda la información que él considere oportuna (cuantos más campos rellene, mayores serán sus posibilidades de obtener mejores resultados), el sistema buscará en todos los repositorios configurados, todos aquellos objetos de aprendizaje que coincidan con los parámetros marcados por el usuario, devolviéndole a éste, una lista ordenada de la que podrá ir descargando todos y cada uno de ellos, como se puede ver en la Figura 3.21 (izda.).

A partir de esta ventana, el sistema mostrará el nombre del objeto de aprendizaje en cuestión, junto con su porcentaje de coincidencia y un enlace para procesar su descarga. Esta es una de las principales diferencias con respecto a los demás repositorios analizados; la arquitectura presentada posibilita la descarga del objeto de aprendizaje, junto con su metainformación, de manera totalmente independiente del formato o manera de organizar los objetos docentes. Se muestra en la Figura 3.21 (dcha.) cómo será este proceso:

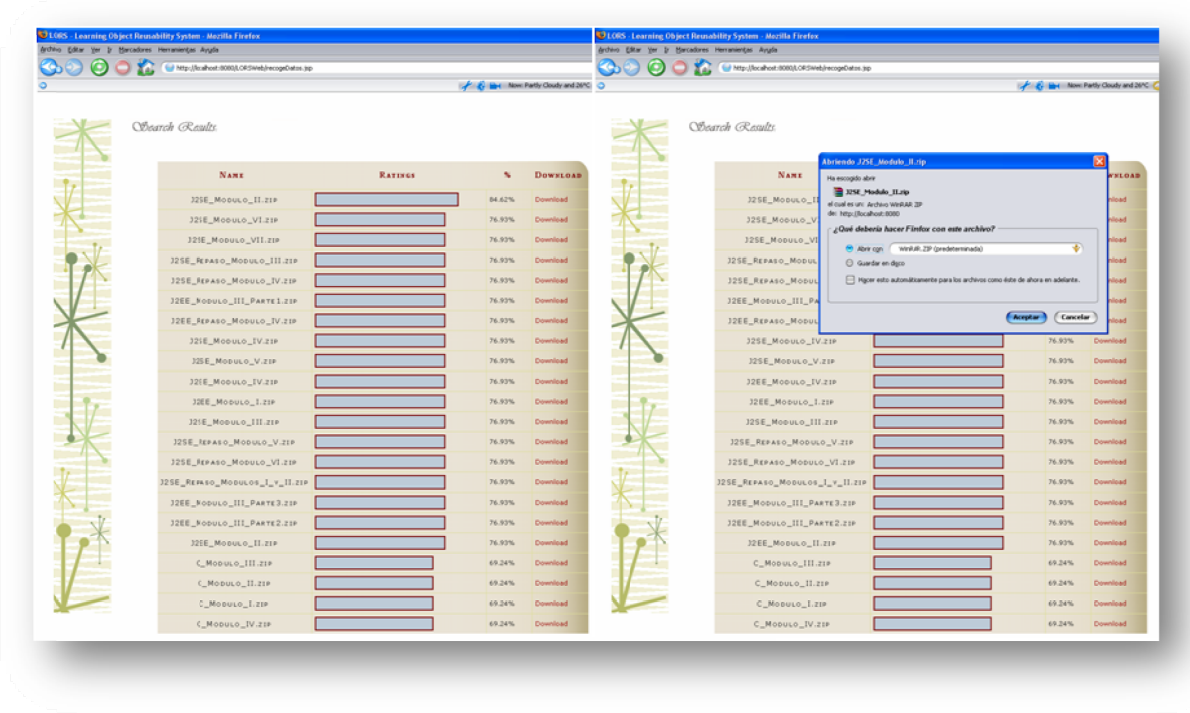


Figura 3.21 (izda.) Coincidencias encontradas. (dcha.) Descarga del objeto de aprendizaje

En cuanto a la catalogación de repositorios, se puede indicar que se presentará una interfaz Web sobre la que el usuario tendrá que seleccionar una de las tres formas distintas de añadir un nuevo repositorio al sistema (en realidad lo que se añadirá será el servicio Web de búsqueda que referenciará a dicho repositorio). El diseño de la interfaz se muestra en la Figura 3.22 (izda.).

Si el usuario deseara registrar un servicio Web de búsqueda sobre el UDDI del sistema, solamente tendría que proporcionar la URL de dicho servicio, ya que el resto de datos, como por ejemplo nombre de usuario o contraseña de la cuenta necesaria para registrar servicios sobre cualquier UDDI, o negocio sobre el que publicarlo, serán proporcionados automáticamente. Si por el contrario el usuario deseara registrarlo sobre

un UDDI externo al mismo, todos estos datos deberían de ser proporcionados. Un ejemplo de esta interfaz lo podemos ver a través de la Figura 3.22 (dcha.).

Una vez que el usuario haya rellenado la información necesaria, en caso de necesitarse se le pedirá la confirmación de sus datos. Un ejemplo de esto podría ser la imagen anterior, en la que el usuario desea registrar el servicio sobre un UDDI externo al sistema, por lo que entre otros, es necesario que proporcione el nombre del negocio sobre el que publicar dicho servicio. Sin embargo, para registrar un servicio no es necesario el nombre del negocio, sino su clave asociada. Para facilitar esta tarea el sistema le pedirá el nombre del negocio y buscará la(s) claves de todos los negocios que tienen ese nombre, para así mostrárselos al usuario en un menú desplegable y que él pueda seleccionarlo, facilitándole así la labor al usuario.

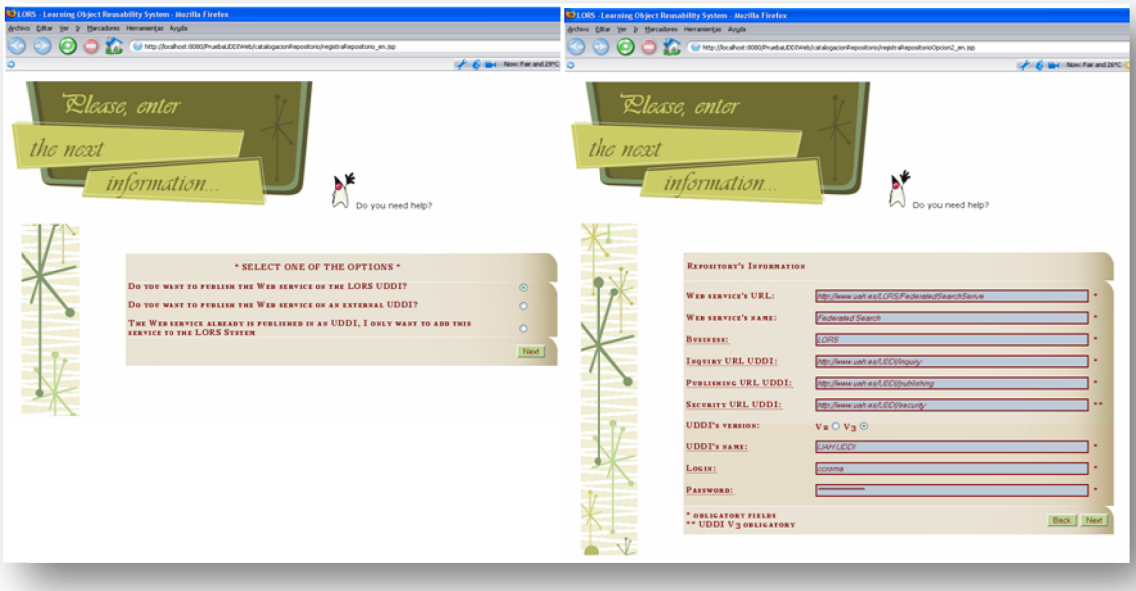


Figura 3.22 (izda.) Interfaz de entrada al sistema de catalogación de repositorios. (dcha.) Interfaz de entrada de información del repositorio

3.5.4. Implantación de SROA

Los anteriores apartados comenzaron poniendo de manifiesto la problemática que existe actualmente con los sistemas de formación online, debido principalmente a la poca reutilización que se hace de toda la información docente que se encuentra en formato electrónico. Esto, evidentemente, repercute en el coste total de los procesos educativos, incrementando sus presupuestos notablemente. Se estudiaron varias propuestas, pero ninguna de ellas proponía ninguna solución global a estos problemas, por lo que se decidió desarrollar una arquitectura que permitiera solventar este problema, y un sistema software que la pone de manifiesto. Sin embargo, todavía queda un problema sin resolver, ¿qué es lo que tiene que hacer un usuario cualquiera para poder implantar SROA? ¿O, qué es lo que tiene que hacer para hacer accesibles sus contenidos docentes al sistema SROA? ¿Tiene que ceñirse a unas especificaciones poco flexibles? ¿Necesita adaptar sus sistemas/lenguajes/protocolos? Y así podrían formularse algunas preguntas más al respecto. La respuesta a todas ellas es NO. SROA es completamente flexible en este aspecto, y a lo largo de este apartado se explicará cómo lo consigue.

Para incorporar cualquier repositorio distribuido (independientemente del estándar educativo que maneje) sobre la plataforma SROA, será necesario realizar los siguientes pasos:

- En primer lugar, el usuario u organización que desee dar acceso a su repositorio, y con él a todo el material docente albergado, deberá desarrollar un sistema de búsqueda. El sistema de búsqueda federada con el que cuenta SROA, podrá de esta manera ponerse en contacto con el sistema de búsqueda del repositorio, y así obtener todos aquellos objetos de aprendizaje que cumplan con las características especificadas por el usuario en el formulario de entrada.
- En segundo lugar, se deberá dar de alta el repositorio en la plataforma SROA, de forma que este último sea consciente que las búsquedas realizadas a partir de ese momento, tendrán que efectuarse sobre ese repositorio también.

Del primero de los pasos no se conoce nada todavía, solamente se sabe que todo repositorio que quiera hacer accesibles sus elementos docentes a SROA, deberá contar con un servicio Web con el que comunicarse. Sin embargo, el segundo de los apartados sí que ha sido comentado anteriormente, así que se centrará el estudio en el primero de ellos.

Para facilitar la creación de un servicio de búsqueda adaptado a cada repositorio de un usuario u organización, se decidió llevarlo a cabo de dos maneras diferentes:

Por un lado, el proveedor del repositorio podrá decidir desarrollar directamente el servicio de búsqueda asociado a su repositorio. De esta manera, tendrá un control absoluto sobre la forma de realizar búsquedas de objetos en su repositorio, el rendimiento que estas ofrecen, el lenguaje de programación que desee utilizar para su desarrollo, etc. La plataforma SROA no impone ninguna de estas condiciones para que un nuevo repositorio forme parte de su sistema de búsqueda federada; por lo tanto, se puede indicar que en este aspecto es totalmente flexible y adaptable.

Sin embargo, también presenta otra alternativa de incorporación más eficiente: utilizar un sistema de búsqueda ya creado, y adaptarlo a las necesidades particulares del repositorio. Para esta tarea, se distribuye una librería llamada JSBR-API (API Java - Sistema de Búsqueda para Repositorios), desarrollada bajo el lenguaje de programación Java, y distribuida a través de un fichero de recopilatorio JAR (Java ARchives). De esta manera, cualquier repositorio podrá hacer accesible el contenido docente, sin necesidad de desarrollar ningún sistema de búsqueda complejo, ya que SROA proporcionará dicho sistema, haciendo que el usuario u organización solamente tengan que invocar a la funcionalidad necesaria de la API.

Para la adaptación del sistema de búsqueda a las peculiaridades propias de cada repositorio, se optó por utilizar dos ficheros de configuración externos al propio fichero JAR, y que también serán distribuidos.

A continuación se muestra en la Figura 3.23, un esquema general de SROA, en el que se identifican los servicios Web de búsqueda asociados a cada repositorio, utilizándose para su creación la librería JSBR-API (archivo JAR), así como los archivos de configuración que permitirán adaptarla al mismo.

La librería JSBR-API ofrece dos funcionalidades:

- Generar la búsqueda de objetos a partir de unos parámetros marcados por el usuario, devolviendo al sistema SROA un listado con todos aquellos objetos coincidentes.
- Devolver al usuario un objeto de aprendizaje en concreto.

Una vez comentadas las funcionalidades ofrecidas por la librería JSBR-API, sólo quedaría establecer un servicio Web sobre el repositorio, que utilizando los procesos anteriormente comentados, posibilitara el acceso a la información docente albergada en su interior. Esta será una tarea sencilla, ya que el servicio Web solamente tendrá que invocar a estos dos procesos, y retornar al sistema SROA la información que le devuelvan. Esto también supone un factor muy importante dentro de la sencillez de incorporación de un nuevo repositorio al sistema, ya que el proveedor podrá utilizar cualquier plataforma de servicios Web, sin que de nuevo esto suponga una restricción para su uso.

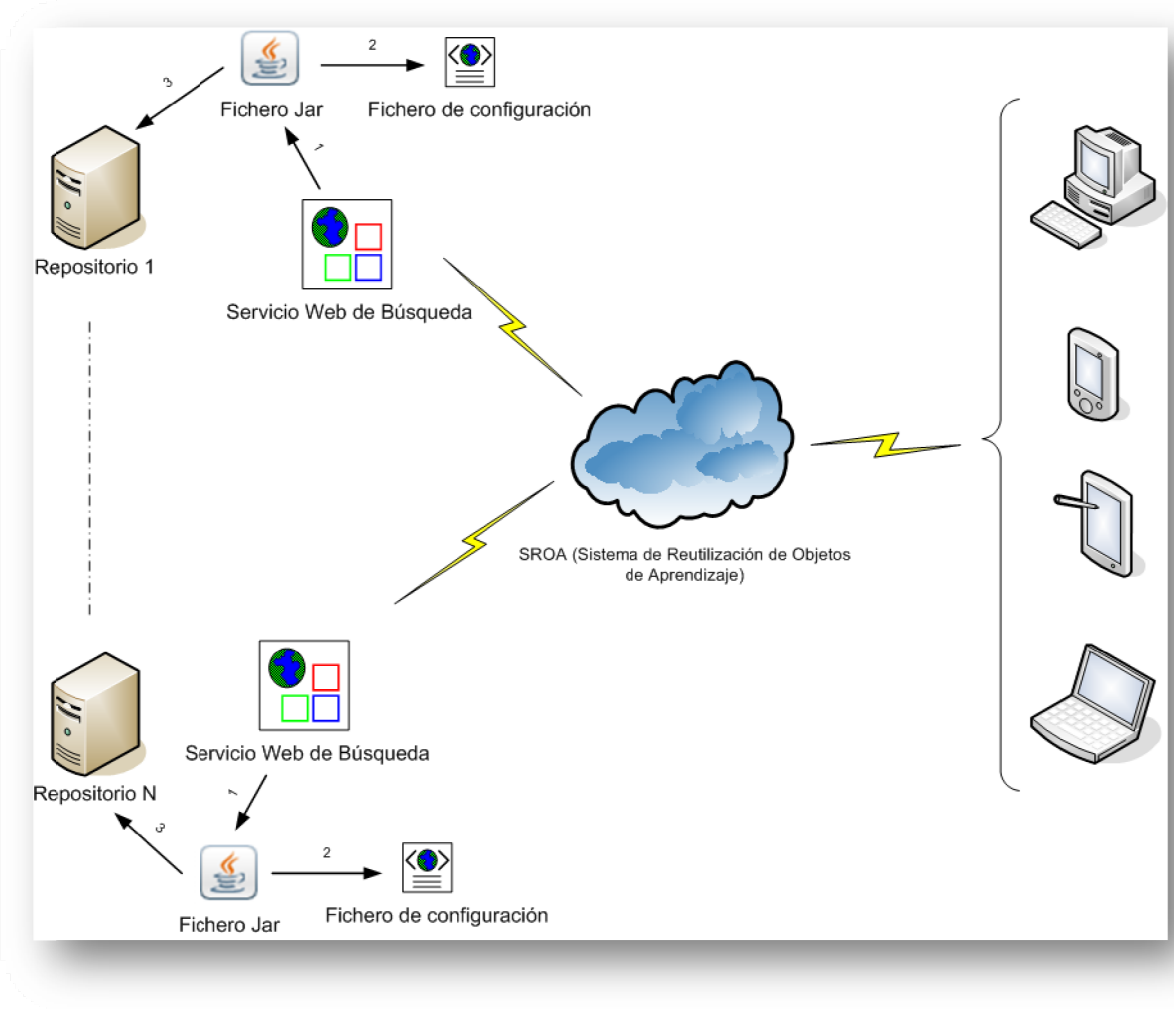


Figura 3.23 Esquema general de funcionamiento del sistema SROA

Este es otro de los motivos por los que SROA sí que es un sistema de búsqueda que realmente mejora la reutilización y distribución de material docente, realizando todo el trabajo de adaptación automáticamente, lo que supone un gran beneficio para la industria del e-learning.

3.6. JUSTIFICACIÓN DEL OBJETIVO DE LA TESIS

Sin duda, el mayor problema que aborda la industria del e-learning en la actualidad, aún sin resolver en aspectos muy fundamentales, es la ausencia de unas metodologías técnicas, documentales y psicopedagógicas comunes y aceptadas que garanticen los objetivos de accesibilidad, interoperabilidad, durabilidad y reutilización de los materiales docentes. Por otro lado, el rápido avance de las infraestructuras tecnológicas y la falta de compatibilidad entre tecnologías e-learning existentes es una de las principales barreras para su desarrollo, y hace de la adopción de estándares el instrumento crucial para su correcta y rápida evolución. Por estos motivos, actualmente se está produciendo una convergencia hacia estándares comunes e intercambiables, así como la definición de modelos que permiten la interoperabilidad de gestores de contenidos docentes.

Ya que el requisito indispensable de los estándares es garantizar la interoperabilidad entre los componentes de diferentes vendedores [Henderson, 2003], es decir, que distintos sistemas o plataformas puedan intercambiarse información y trabajar conjuntamente, es necesario que las plataformas de construcción de componentes e-learning y las herramientas de autoría se basen en ellos. Al estar los estándares en evolución, no es extraño encontrar productos que no siguen estrictamente un estándar de una organización en concreto, sino que han sido desarrollados aplicando diferentes recomendaciones de los estándares más comunes en cada una de sus partes, adaptando de esta manera los estándares a las necesidades del producto.

Queda claro que para que un objeto de aprendizaje sea reutilizable debe crearse conforme a la definición de algún estándar y, por lo tanto, el contenido del objeto debe estar descrito mediante metadatos. Sin embargo, los estándares no ofrecen ninguna pauta sobre cómo puede ser descubierto un objeto de aprendizaje. Por lo tanto, los metadatos son el primer paso para hacer que un objeto de aprendizaje sea reutilizado, pero faltan otros muy importantes como el de su descubrimiento e intercambio entre diversas plataformas de formación.

Resumiendo lo anterior, se puede afirmar que el material educativo debe ser:

- Desarrollado conforme a estándares.
- Descubierto de forma universal.
- Fácil de encontrar.

- Independiente del formato de metadatos.
- Reusable.
- Independiente de la plataforma que lo almacena.
- Integrable en otros sistemas de formación.

Como se indicó al principio de este capítulo, en el apartado de evolución de los sistemas de aprendizaje, éstos han evolucionado hacia la comunicación a través de Internet, ya sea para comunicarse con sus usuarios o para interoperar con otros sistemas. La principal característica de los nuevos entornos de aprendizaje virtuales es que utilizan la Web como única plataforma de distribución; esto implica que disponen de una serie de capacidades hasta ahora inexistentes para las aplicaciones de enseñanza asistida por ordenador, que permiten superar ciertas deficiencias, tales como los elevados costes de producción y tiempos de desarrollo. Sin embargo, se aprecia una reutilización prácticamente nula tanto de los materiales docentes como de los componentes software que integran una herramienta de teleformación.

Estos problemas los resume Koper [2000] en los siguientes puntos:

- El aumento en la heterogeneidad de los productos y en la interacción entre personas y sistemas, únicamente entre personas y únicamente entre sistemas.
- El espectacular aumento de información disponible y su dispersión en distintos sistemas y aplicaciones, lo que implica la necesidad de poner en comunicación distintos productos software y plataformas.
- La organización de procesos de aprendizaje distribuidos, motivada por la dispersión geográfica de los usuarios de los cursos.

Y es para solucionar estos inconvenientes por lo que surgen diferentes iniciativas que pretenden unificar, tanto la forma de crear contenidos, como la forma de implantar plataformas y repositorios educativos, y de comunicar esos contenidos. Ofrecer una solución a estos inconvenientes es exactamente el objetivo de esta Tesis.

Para solucionar los problemas planteados se necesita establecer la arquitectura de un sistema software capaz de asumir las funcionalidades derivadas de esta problemática,

que se puede resumir en la interoperabilidad de sistemas de e-learning (su integración) y en la reutilización de los objetos de aprendizaje que contienen.

Teniendo en cuenta la arquitectura de un sistema e-learning desarrollado mediante la integración de un conjunto de componentes independientes, debería tener las siguientes características [Cisco, 2001]:

- **Abierta**. El objetivo es crear aplicaciones e-learning interoperables y conectables entre sí de forma sencilla ("plug-and-play"), es decir, que herramientas comerciales de fabricantes distintos puedan ensamblarse en un único sistema global. Para ello es necesario que el marco de definición de la arquitectura del sistema sea conforme a un modelo estándar.
- **Escalable**. Independientemente del tamaño inicial con que se conciba el sistema, la arquitectura debe estar definida de tal forma que permita su crecimiento. Por ejemplo, al ir aumentando los repositorios de objetos educativos las aplicaciones encargadas de gestionarlos deben tener capacidad suficiente para no sobrecargarse.
- **Global**. Permitir la diversidad lingüística y cultural. Este es uno de los objetivos más difíciles de conseguir, puesto que la gran mayoría de las aplicaciones están destinadas para una audiencia anglosajona: actualmente existen varios esfuerzos de estandarización en marcha cuyo objetivo es presentar un mismo contenido (incluso un mismo entorno de aprendizaje) en diferentes lenguas en función del usuario a quien esté destinado.
- **Integrada**. No sólo entre los componentes del propio sistema, sino entre otras aplicaciones no directamente relacionadas con el aprendizaje (por ejemplo, de gestión de recursos humanos, financieras, de gestión del conocimiento, etc.). El objetivo es conseguir la interoperabilidad entre todas ellas.
- **Flexible**. Es importante la capacidad de implementar nuevas soluciones sin tener que efectuar grandes cambios en la arquitectura del sistema.

Aunque como se ha visto en el apartado anterior ya existen determinados repositorios y sistemas de búsqueda que intentan potenciar esta reutilización de objetos educativos, se ha comprobado que aún no es suficiente. Existen todavía muchos aspectos en los que estas arquitecturas no cumplen con su cometido y precisamente el objetivo de esta Tesis es paliar estas necesidades detectadas en aspectos como:

- Permitir la búsqueda y descarga de cursos a través de la composición de éstos como conjuntos de objetos de aprendizaje provenientes de diferentes repositorios distribuidos.
- Afinar los sistemas de búsqueda a través de la introducción de técnicas semánticas en diferentes niveles.
- Mejorar la interoperabilidad entre plataformas y repositorios.
- Garantizar siempre la accesibilidad al material educativo.

Por lo tanto, el objetivo propuesto en esta Tesis es el de definir la arquitectura orientada a servicios de un sistema capaz de resolver los problemas planteados anteriormente e implementar un prototipo real, para demostrar con su funcionamiento que la arquitectura planteada se puede llevar a la práctica.

4. PROPUESTA ARQUITECTURAL

Hasta el momento se ha realizado un estudio del estado del arte (capítulo 2) en el que se han expuesto las principales tecnologías que existen hasta la fecha tanto para la estandarización de los sistemas e-learning como para la construcción de los mismos.

En la primera categoría se pueden destacar las especificaciones creadas por organizaciones como IEEE, IMS, ADL o ISO entre otras, las cuales velan por garantizar la interoperabilidad de los entornos e-learning, garantizar la mayor reutilización posible del contenido educativo que albergan, garantizar la completa accesibilidad al material docente, ofrecer sistemas de búsqueda eficientes, tecnologías orientadas a ofrecer sistemas de control de documentación protegida por derechos de autor, etc. En la segunda categoría se pueden destacar especificaciones orientadas a indicar cuáles deberían ser las principales características con las que contara un sistema orientado a la distribución y búsqueda de este tipo de material.

Posteriormente se realizó un estudio en el que se detallaban las principales características con las que cuentan los sistemas de búsqueda y almacenamiento de material e-learning (capítulo 3). De este estudio se obtuvo la conclusión de que, hoy en día no existe un sistema global de búsqueda de contenidos educativos en sistemas de almacenamiento distribuido que cuente, entre otras, con las características de interoperabilidad, eficiencia en las búsquedas, composición de cursos como conjuntos de varios objetos educativos procedentes de diferentes sistemas, etc. Si que existen sistemas que ofrecen algunas de estas características, como se vio en el anterior capítulo, pero no un sistema global de aprendizaje. Por lo tanto como conclusión a este estudio, surge la posibilidad de plantear un sistema que sea capaz de paliar estas necesidades y ofrezca un sistema global de aprendizaje que garantice la reutilización del material docente y con ello, el éxito de los sistemas de formación a distancia como apoyo a los sistemas de formación tradicional.

También en el capítulo 3 se realizó una pequeña introducción a lo que desarrollar una arquitectura software se refiere, para posteriormente describir tanto la arquitectura como el sistema que precede al estudio de esta Tesis, ya que como se ha comentado anteriormente, este estudio no parte de cero, sino que es una evolución de una arquitectura anterior.

Por lo tanto, en este capítulo se mostrarán las arquitecturas generadas para solventar los problemas descritos en el capítulo 3, siendo estas validadas con las implementaciones de prototipos reales en el capítulo 5. En este capítulo se presentan dos arquitecturas:

- LORA-SQI: Adaptación de la arquitectura inicial a las nuevas especificaciones existentes para garantizar su completa interoperabilidad con el resto de sistemas.
- LORA-SC: Modificación de la arquitectura anterior en la que se añaden técnicas semánticas para posibilitar tanto una mayor eficiencia en las búsquedas como nuevas funcionalidades derivadas de su uso, entre las que se destaca la de permitir la búsqueda de cursos completos de aprendizaje.

4.1. PROPUESTA DE ARQUITECTURA INICIAL: LORA-SQI

La arquitectura que se propone en este apartado es una evolución de la arquitectura SROA, definida en una tesis doctoral anterior [Otón, 2006], para adaptarla a la especificación SQI (Simple Query Interface) [CEN, 2005]. Para ello se han realizado diversas modificaciones en su arquitectura, las cuales se presentarán a continuación, desarrollando lo que se conoce como LORA-SQI (Learning Object Reusability Architecture – SQI).

El porqué de esta adaptación radica fundamentalmente en los siguientes aspectos:

- Aunque la arquitectura SROA se desarrolló conforme a los estándares y especificaciones existentes hasta la fecha (se recuerda que la primera versión de SROA es de mediados de 2005 y SQI data de Noviembre de 2005), no es 100% compatible con todos los repositorios del mercado. Para que esta compatibilidad sea posible era necesario la incorporación de una librería denominada JSBR-API (explicada en el capítulo 2). La tarea de adaptabilidad era relativamente sencilla, pero aún así necesitaba de determinados parámetros que podrían hacer de ésta un escollo inalcanzable para algunos sistemas de búsqueda.
- La especificación SQI cada vez ha ido tomando más fuerza, hasta terminar convirtiéndose en un referente en lo que a interoperabilidad entre sistemas de búsqueda se refiere. Es tal su uso, que hoy en día prácticamente todos los repositorios del mercado (junto con sus sistemas de búsqueda) la utilizan como interfaz de consulta totalmente estandarizada, por lo que este sistema también necesitaba de su implementación para salvaguardar su cometido.
- Junto con esta interfaz de consulta, surgen determinados lenguajes que permiten llevar a cabo las consultas de objetos de aprendizaje. En función de la complejidad de estas consultas, se podrán utilizar unos u otros lenguajes. Por lo tanto otra modificación necesaria iba a ser la incorporación al sistema de motores de interpretación de este tipo de consultas.

Gracias a la utilización de servicios, y a la flexibilidad del código, todas estas modificaciones se pueden llevar a cabo sin necesidad de grandes cambios estructurales en la arquitectura. A continuación se pasan a detallar todos estos cambios, presentando la nueva arquitectura del sistema.

4.1.1. Niveles de la arquitectura

La nueva versión de la arquitectura también se compone de varios niveles, al igual que en el caso anterior, pero mucho más simplificado, haciendo que determinados procesos sean más rápidos y simples, pues como se puede apreciar en la Figura 4.1 los componentes de este sistema son más concretos y siempre ligados a la especificación SQI.

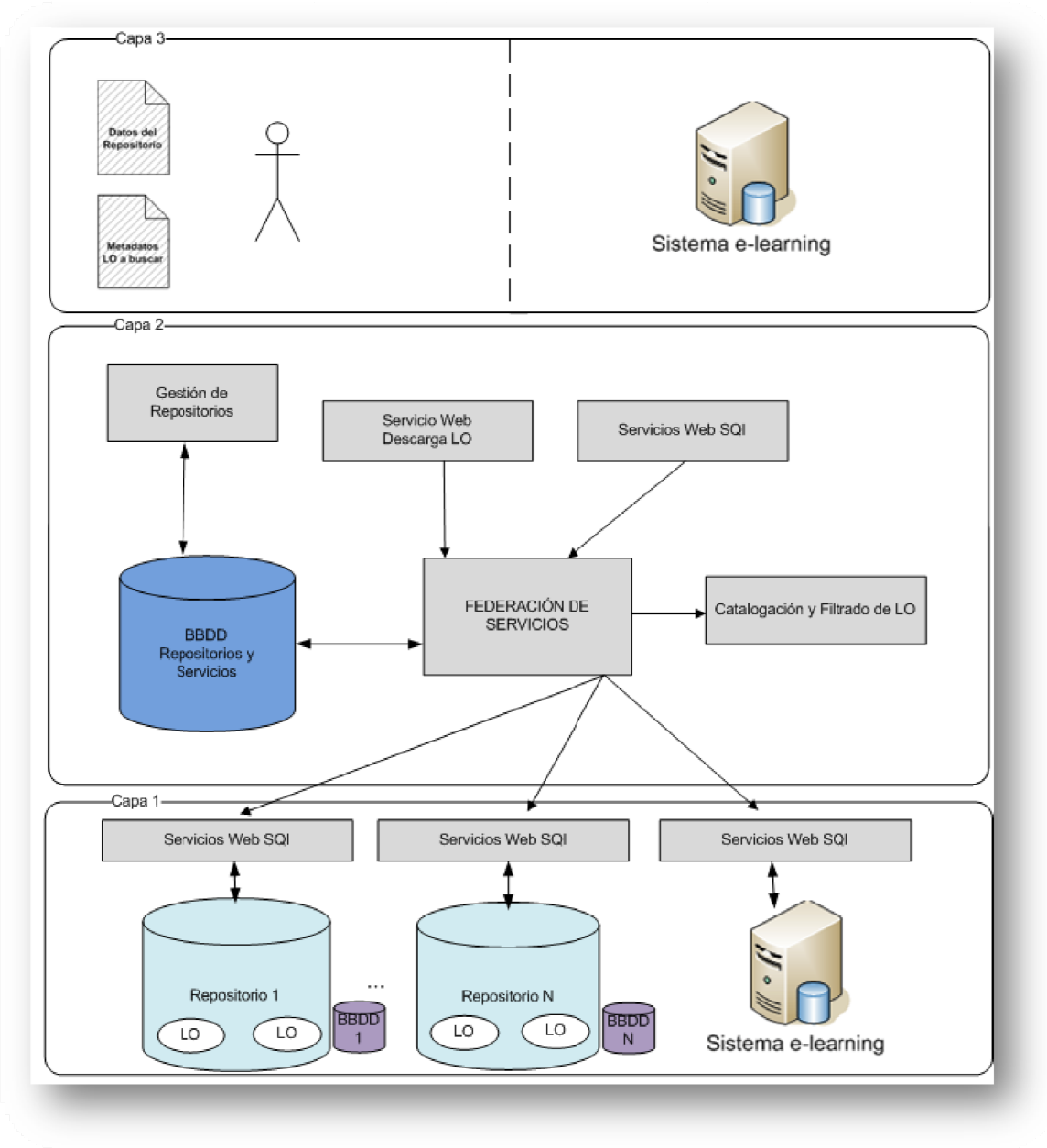


Figura 4.1 Niveles de la arquitectura de LORA-SQI

Nivel 1. Repositorios SQI

En el primer nivel se encuentran todos los repositorios sobre los que se realizarán búsquedas utilizando el interfaz SQI. En este nivel se encontrarán repositorios implementados utilizando la librería SQITL-API (la equivalente a la anteriormente mencionada JSBR-API pero con los cambios y actualizaciones de SQI), y otros repositorios; los cuales disponen de implementaciones SQI y a su vez publican los servicios Web que dan acceso a dichas implementaciones.

Los repositorios desarrollados mediante la librería anteriormente mencionada en vez de hacer uso de un fichero XEL como proponía la anterior versión de la arquitectura, cuentan con una base de datos que mantiene la propia librería con la metainformación de los recursos pertenecientes al repositorio. De esta manera se consiguen consultas y búsquedas mucho más rápidas, ya que ahora se cuenta con la indexación de una base de datos para la realización de consultas; además la librería convierte la consulta SQI en consulta SQL y la ejecuta obteniendo un rendimiento notablemente más alto. Este sistema llevaría una penalización, y es la posible desincronización entre la metainformación con la que cuenta la base de datos y con la que cuentan los objetos de aprendizaje. Para solventar este problema se propone un mecanismo automático de actualización de estos datos (que también podrá ejecutarse de manera manual), de forma que en todo momento se cuente con datos lo más actualizados posibles. Es cierto que se debe invertir un tiempo en estas actualizaciones, pero estas operaciones no se realizan tan a menudo como las búsquedas o consultas por lo que parece sensato ganar rapidez en este último tipo de operaciones puesto que son el objetivo principal del sistema.

Estos repositorios dispondrán de servicios Web para la gestión de sesión y consultas SQI, por lo que además de poder ser incluidos en las búsquedas de LORA-SQI, también pueden ser invocados por cualquier repositorio en Internet utilizando los métodos SQI adecuados. Como se puede comprobar se ha dado un paso más en cuanto a la interoperabilidad se refiere, ya que los repositorios son accesibles únicamente publicando sus servicios Web, siendo posibles búsquedas sobre ellos por cualquier sistema que implemente la especificación SQI. Además estos repositorios disponen de un servicio Web más, independiente de los correspondientes a SQI, que permite la descarga directa de recursos de aprendizaje como se explicará más detalladamente a lo largo de este apartado.

Nivel 2. Federación de Servicios

Este nivel lo compone la parte de federación de servicios de LORA-SQI, así como la interfaz SQI con la que cuente esta arquitectura tanto en el sentido de "fuente" de consultas (sobre los repositorios del nivel 1), como en el sentido de "destino" de consultas SQI (consultas provenientes de otros sistemas de búsqueda). Esto quiere decir que de la misma forma que antes, cualquier sistema que implemente la especificación SQI puede realizar búsquedas en LORA-SQI, lo que significa realizar búsquedas en todos los repositorios registrados en él.

Como se puede observar, la especificación SQI está presente en todas las formas de comunicación existentes entre los sistemas de esta arquitectura, garantizando una completa interoperabilidad entre orígenes y destinos de información.

La federación de servicios de LORA-SQI consiste en realizar las operaciones SQI que recibe a todos los repositorios registrados en su base de datos, multiplicando de esta manera las consultas y también los resultados, los cuales hay que filtrar y ordenar, pero de una forma sencilla (puesto que los resultados devueltos por cada repositorio están ya en orden – normalmente ordenados por índices de importancia), para producir los resultados deseados correspondientes a una determinada consulta.

Se ha decidido en este caso prescindir también de la parte de gestión de usuarios, puesto que la especificación SQI dispone de una parte que abarca esta funcionalidad: la gestión de sesión SQI.

Nivel 3. Acceso y Presentación

Este nivel corresponde al nivel de presentación, y es dónde un usuario podrá realizar todas las operaciones SQI necesarias para realizar búsquedas y obtener recursos de aprendizaje.

También se dispone de una parte de gestión de repositorios en la cual se podrán registrar los repositorios o sistemas que implementen SQI para realizar búsquedas sobre

sus repositorios y de esta manera aumentar la cantidad, calidad y diversidad de los resultados obtenidos en las consultas. En cuanto a las operaciones SQI disponibles en la arquitectura, el usuario podrá personalizar las consultas siempre respetando la especificación, pero con total libertad para parametrizar dichas consultas.

La utilización de un fichero XSD con los campos educativos sobre los que realizar las búsquedas, en el caso de las búsquedas avanzadas o exactas para generar el formulario de entrada dinámico, es un aspecto que se ha mantenido de la versión anterior, por ser de gran utilidad a la hora de implementar dicha funcionalidad, puesto que de esta manera resulta sencillo establecer los valores a los campos sobre los que después realizar la búsqueda.

Esta arquitectura ya ha sido presentada en varios congresos internacionales, entre los que se destacan la décima conferencia internacional de información, integración y aplicaciones y servicios basados en el Web (IIWAS 2008), con el título de "The Integration Of SQI in a Reusable Learning Objects System: Advantages and Disadvantages" [Otón et al., 2008], o la novena conferencia internacional IEEE de tecnologías avanzadas de educación (ICALT 2009), con el título de "Assessment of the standard query interface of public learning objects repositories" [Hilera et al., 2009].

Además de los congresos anteriormente citados, esta arquitectura se ha presentado en la revista "International Journal of Innovative Computing, Information and Control (IJICIC)" con un índice de impacto JCR en 2008 de 2,791 (primer tercio) con el título de "Service Oriented Architecture for the Implementation of Distributed Repositories of Learning Objects" [Otón et al., 2010]

4.2. PROPUESTA DE ARQUITECTURA AMPLIADA: LORA-SC

Durante este apartado se detallarán las características de la ampliación de la arquitectura anterior, cometido principal del estudio de esta Tesis. En el anterior apartado se definió una arquitectura, modificación del punto de partida de esta Tesis descrita en el capítulo 3, pero que sin embargo no supone un cambio importante en cuanto a la funcionalidad ofrecida por la misma, ya que, a grandes rasgos la funcionalidad es la misma. Los cambios más importantes se han producido a nivel interno, para su adaptación a las nuevas especificaciones aparecidas, en particular a la especificación SQI, de forma que esta arquitectura, y su posterior desarrollo, hagan del mismo un entorno completamente interoperable con cualquier otra arquitectura o sistema de búsqueda de contenidos docentes.

Sin embargo la arquitectura presentada en este apartado sí que supone un cambio funcional importante. Se presentan tres nuevas propuestas, descritas en los siguientes seis sub-apartados:

- El primer sub-apartado sirve de introducción, y en él se detallan los motivos que llevaron al planteamiento y desarrollo de una nueva arquitectura para el desarrollo de un sistema completo de reutilización de objetos de aprendizaje. Este sub-apartado sirve de introducción al resto, de forma que se muestra resumidamente los diferentes cambios y los motivos que llevaron a su estudio y posterior desarrollo.
- El segundo sub-apartado se centra en el estudio de una nueva especificación para la definición de una interfaz de consulta que solucione algunos problemas con los que cuenta SQI. En este sub-apartado se realiza una exposición de la principal problemática con la que cuenta la interfaz de consulta estándar que se utilizaba en la versión anterior, junto con una solución a la misma.
- El tercer sub-apartado hace referencia a cómo la introducción de técnicas semánticas podrá ayudar a que el acceso a la información sea en términos de qué es lo que representan los conceptos usados en la descripción de los objetos de aprendizaje y no como hasta ahora, que solo se aceptan coincidencias porque los términos coinciden sin importar el significado de los mismos.
- El cuarto sub-apartado muestra una nueva funcionalidad que puede ser añadida a los sistemas de búsqueda federada actuales. Se trata de permitir a los usuarios realizar búsquedas de cursos completos, de forma que el sistema acceda a los distintos repositorios distribuidos y obtenga de ellos todos los

objetos de aprendizaje individuales que sean necesarios para la generación del curso, para posteriormente, componerlo y proporcionárselo al usuario.

- El quinto sub-apartado muestra un resumen de la nueva arquitectura presentada, incorporando todas las propuestas anteriores.
- El último sub-apartado muestra un resumen de las principales características de este nuevo sistema de búsqueda federada.

4.2.1. Introducción

Partiendo de la arquitectura mostrada en el apartado 4.1, se podría indicar que se ha desarrollado una arquitectura que garantiza la completa reutilización de los objetos de aprendizaje. Es posible realizar esta afirmación ya que, entre sus principales características se destacan:

- Está basada en los principales estándares y especificaciones emitidos por los principales consorcios, agencias o comités encargados de velar por el desarrollo de sistemas eficientes, interoperables e independientes de las posibles especificaciones que pudieran elaborar cada empresa para adaptarlos a sus necesidades.
- Gracias a lo anterior, se trata de una arquitectura completamente interoperable con el resto de sistemas y repositorios. Este requisito se cumple no solamente indicando que un sistema que se base en la arquitectura LORA podrá realizar búsquedas en el resto de sistemas o repositorios que cumplan SQI (la gran mayoría, como se indicó anteriormente), sino que estos repositorios o sistemas de búsqueda también podrán realizar búsquedas en los repositorios asociados a LORA, haciendo que la reutilización del material docente pueda ser todo lo amplia y compleja como se quiera.
- Se trata de una arquitectura que permite una búsqueda eficiente del material educativo, presentando dos tipos distintos de búsquedas. Tan solo ARIADNE y ADL-R presentan dos tipos de búsquedas (se recuerda que estas tienen que ver con los niveles de PLQL que se implementen). De esta forma se agudizan en la medida de lo posible las búsquedas realizadas, pudiendo incorporar además de búsquedas por un campo clave (PLQL de nivel 0), búsquedas por varios campos de metainformación (PLQL de nivel 1).
- Se trata de una arquitectura que permite la adaptación del material educativo que se descargue al estándar de metainformación utilizado por el destinatario de la información; de esta forma tendrá un objeto de aprendizaje completamente adaptado a su sistema y listo para ser usado. Este sistema de adaptación ya se incorporó en la primera versión de la arquitectura y a día de hoy sigue siendo la única arquitectura de búsqueda que cuenta con esta característica.
- También presenta la característica de poder ser adaptable completamente a cualquier conjunto de metadatos que se utilice a día de hoy o que pueda utilizarse en el futuro. Cuando antes se hablaba de que LORA presenta dos tipos distintos de búsqueda, se indicó que la búsqueda que utiliza PLQL de nivel 1 es aquella que se realiza sobre un conjunto de metadatos, pero, ¿qué ocurre si estos metadatos cambian en un futuro o un usuario desea utilizar

otros? ¿habría que cambiar todo el sistema? En LORA no ocurre esto, con el cambio de un solo fichero de configuración el sistema se adapta a cualquier conjunto o sub-conjunto de metadatos que se quiera utilizar, bien que exista actualmente (como LOM) o que pudiera surgir en un futuro. Esta es otra característica que no está presente en ninguna otra arquitectura de búsqueda, y que radica vital importancia de cara a la interoperabilidad del sistema presentado.

A pesar de estas ventajas, muchas veces exclusivas de esta arquitectura, existen algunas mejoras que harían aún más si cabe de LORA una arquitectura completa de reutilización de material educativo. A continuación se indican aquellos cambios o técnicas que se podrían modificar o aplicar para dotarlo de mayor capacidad de reutilización:

- En primer lugar sería necesario modificar la interfaz de consulta SQI de forma que las búsquedas de material docente puedan por un lado manejar resultados completos y por otro lado, posibilitar la descarga del material docente junto con su metainformación.
- En segundo lugar sería factible añadir técnicas semánticas en todo el proceso de búsqueda, para que en base a determinadas ontologías, el acceso a la información fuera mucho más efectivo y eficiente, de forma que se acerque más a búsquedas basadas en el significado de los términos y no solamente en los términos propiamente dichos.
- Por último presentar la posibilidad de buscar cursos completos, de forma que el sistema, basándose en una ontología que le muestre qué contenido tiene cada curso, pueda realizar una búsqueda de los diferentes objetos de aprendizaje que lo integrarán, y realizar así una composición del mismo.

En los siguientes apartados se realizará una exposición mucho más detallada de cada uno de los puntos anteriores.

4.2.2. Propuesta de una nueva interfaz de consulta: SQI 2.0

Como se ha indicado anteriormente uno de los problemas detectados en la anterior versión de LORA, es que la interfaz de consulta utilizada para estandarizar el acceso a la información entre todos los sistemas de búsqueda y los repositorios, aunque presenta grandes ventajas, también presenta dos graves problemas:

- El método de consulta no es eficiente devolviendo resultados complejos.
- La interfaz de consulta no presenta un método que permita la descarga del material educativo.

A continuación se comentarán estas dos problemáticas, indicando algunas de las posibles soluciones que el autor de esta Tesis propone incorporar en una nueva versión de SQI, denominada SQI v 2.0.

Propuesta 1: Modificación de los métodos de consulta de SQI

El método de consulta síncrono[2] con el que cuenta SQI se muestra en la Figura 4.2 y tiene el siguiente formato:

Method name	**synchronousQuery**	
Return type	String	
Parameters	*Name*	*Type*
	targetSessionID	String
	queryStatement	String
	startResult	Integer
Fault	NO_SUCH_SESSION INVALID_ QUERY_STATEMENT QUERY_MODE_NOT_SUPPORTED METHOD_FAILURE INVALID_START_RESULT NO_MORE_RESULTS METHOD_FAILURE	

Figura 4.2 Formato del método de consulta síncrona de SQI

[2] Todas las modificaciones realizadas para el método de consulta síncrono con el que cuenta SQI, son igualmente aplicables al método de consulta asíncrono; en este caso a través del atributo *queryResults* del método *queryResultsListener* que es el encargado de darle tratamiento a los datos obtenidos tras una consulta asíncrona.

Como información del método se destaca que:

- La declaración de la consulta es proporcionada en el parámetro *queryStatement*. Esta podrá venir detallada en formato PLQL de nivel 0 o PLQL de nivel 1 en el caso de LORS.
- El parámetro *targetSessionID* indica el identificador de la sesión que se obtendría en el paso de conexión con el repositorio o repositorios destino.
- El parámetro *startResult* indica el registro de comienzo del conjunto de resultados. El número de resultados devueltos en cada consulta se controla mediante *setResultsSetSize* y su valor por defecto y el número total de resultados producidos se limita mediante *setMaxQueryResults*.
- El método devuelve un String (cadena de caracteres) como conjunto de registros de metainformación coincidentes resultado de la consulta.

En cuanto a los posibles errores manejados por la interfaz, se destacan:

- "NO_SUCH_SESSION" en caso que el parámetro *targetSessionID* sea inválido;
- "QUERY_MODE_NOT_SUPPORTED" en caso que el destino no soporte consultas síncronas.
- "INVALID_QUERY_STATEMENT" si la declaración de la consulta no cumple con la sintaxis del lenguaje de consulta.
- "INVALID_START_RESULT" si se proporciona un número inválido en *startResult*.
- "NO_MORE_RESULTS" si *startResult* es cero y no hay más resultados disponibles.
- "METHOD_FAILURE" si la operación falla por otro motivo.

Evidentemente el formato String no es el más adecuado para devolver un conjunto de objetos de aprendizaje coincidentes con unos parámetros de búsqueda elegidos por el usuario. Para llegar a esta conclusión se ha utilizado una herramienta de testeo de repositorios presente en la Web de ARIADNE denominada SQITester (http://ariadne.cs.kuleuven.ac.be/SQI/SQITest.jnlp). Se trata de una herramienta Java que permite, utilizando la interfaz SQI, realizar consultas sobre cualquier repositorio SQI disponible en Internet. Así pues, se puede ver en la Figura 4.3 cómo sería la interfaz de esta herramienta en la que se indica la URL del repositorio en cuestión.

Figura 4.3 Interfaz de acceso a los repositorios en SQITester

Una vez establecida la conexión, y siguiendo los pasos SQI, se indicarán los parámetros de búsqueda. En este caso se realizará una búsqueda del término "Java" sobre ARIADNE utilizando para ello PLQL de nivel 0, como se puede ver en la Figura 4.4.

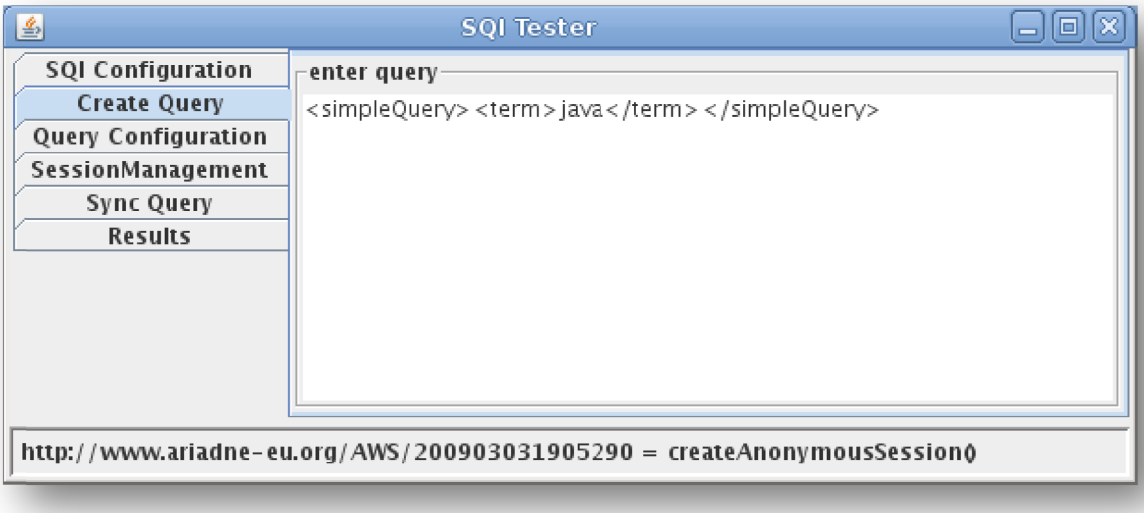


Figura 4.4 Interfaz de consulta en SQITester

Por último, se muestran en la Figura 4.5 los resultados ofrecidos por este repositorio para el término "Java", estando estos representados en formato String, como se indicó anteriormente.

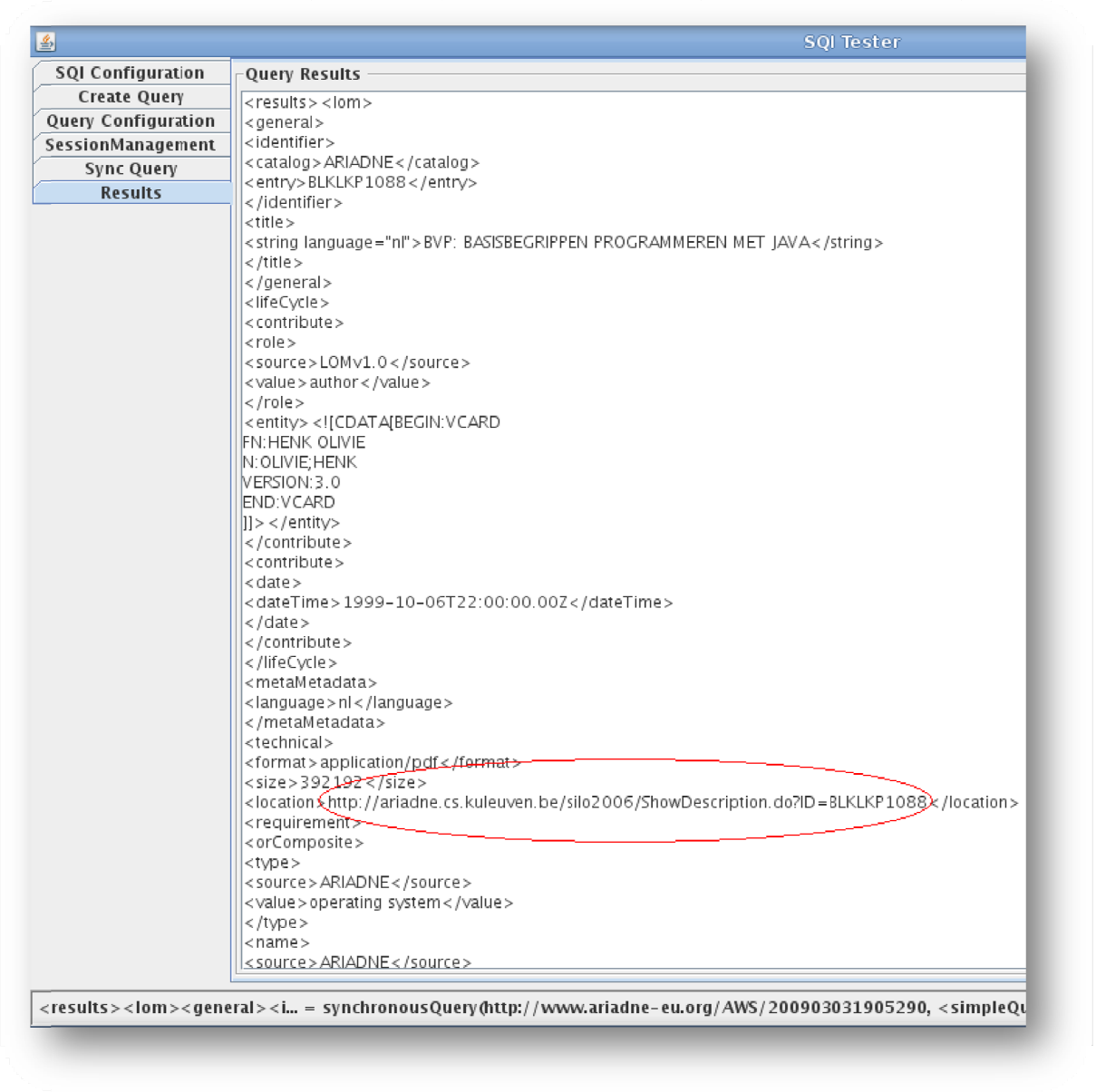


Figura 4.5 Interfaz de visualización de resultados en SQITester

Como se puede ver en la Figura 4.5, los resultados que ofrece el SQITester no son todo lo óptimos que cabría esperar. En la Figura 4.5 se muestra solamente un fragmento de estos datos, pero que sirve para que se entienda la problemática de SQI en este aspecto.

SQI para la búsqueda síncrona de objetos de aprendizaje en repositorios (la más utilizada) utiliza una cadena de caracteres para su retorno. Cada objeto de aprendizaje estará especificado por las etiquetas que correspondan al estándar de metainformación utilizado en cada caso (en este caso LOM: <lom> … </lom>), y a su vez todos ellos

encerrados entre las etiquetas <results> … </results>. Así pues, el formato de resultados sería una cadena de caracteres con el siguiente formato:

```
<results>
    <lom>
    …
    </lom>
    <lom>
    …
    </lom>
    …
</results>
```

Evidentemente no es la forma más óptima de manejar grandes cantidades de información, ya que la cantidad de objetos de aprendizaje que podrá devolver un repositorio puede ser lo suficientemente amplia como para que sea bastante laboriosa la labor de su manejo, filtrado y análisis. Recordemos que según las características de LORS indicadas en apartados anteriores, cuando un usuario realice una búsqueda, esta se hará sobre una serie de repositorios distribuidos, no solamente sobre uno en concreto. Así pues se obtendrá una cadena de caracteres con todos los objetos de aprendizaje que cumplan las características marcadas por el usuario de cada uno de los repositorios. Habrá que juntar todas esas cadenas de caracteres, separar cada objeto de aprendizaje por separado, para de esta forma poder filtrarlos (eliminar los objetos de aprendizaje que se encuentren repetidos en diferentes repositorios) y ordenarlos para mostrarle primero los que tengan un mayor índice de coincidencia.

El proceso a realizar, además de laborioso, como se puede ver, es poco eficiente, ya que obliga a realizar múltiples operaciones de fragmentación de cadenas de caracteres, por lo que esto puede acarrear un coste extra de rendimiento al proceso de búsqueda, siendo este además completamente innecesario de no contar con este sistema de almacenamiento de resultados.

Hoy en día, la mayor parte de las aplicaciones desarrolladas se realizan utilizando algún tipo de lenguaje orientado a objetos (55.9%) o estructurado (39.9%) [TIOBE, 2009]. En ambos tipos de lenguajes es factible la utilización de estructuras de datos: objetos en el caso de los lenguajes orientados a objetos o estructuras en el caso de los estructurados. Gracias a la utilización de estas estructuras de datos, la tarea de filtrar, tratar y ordenar los objetos de aprendizaje sería mucho más efectiva.

Como primera medida de mejora de la especificación SQI se propone modificar el método de consulta para que en lugar de devolver una cadena de caracteres y así dificultar el tratamiento de la información, devolver una estructura de datos que sería un listado dinámico de datos como muestra la Figura 4.6:

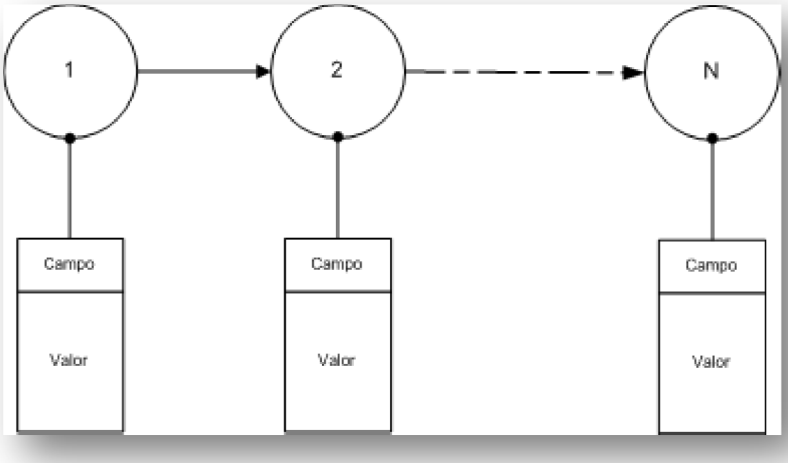


Figura 4.6 Ejemplo de tipo de datos utilizado en SQI v2.0

Se puede ver en la Figura 4.6 cómo sería la estructura de datos seleccionada. Se trata de un listado enlazado de pares de tuplas de campo y valor. El campo especificaría el metadato en cuestión y valor el contenido asociado al mismo, así por ejemplo para el caso anterior que mostraba la Figura 4.5, el resultado sería:

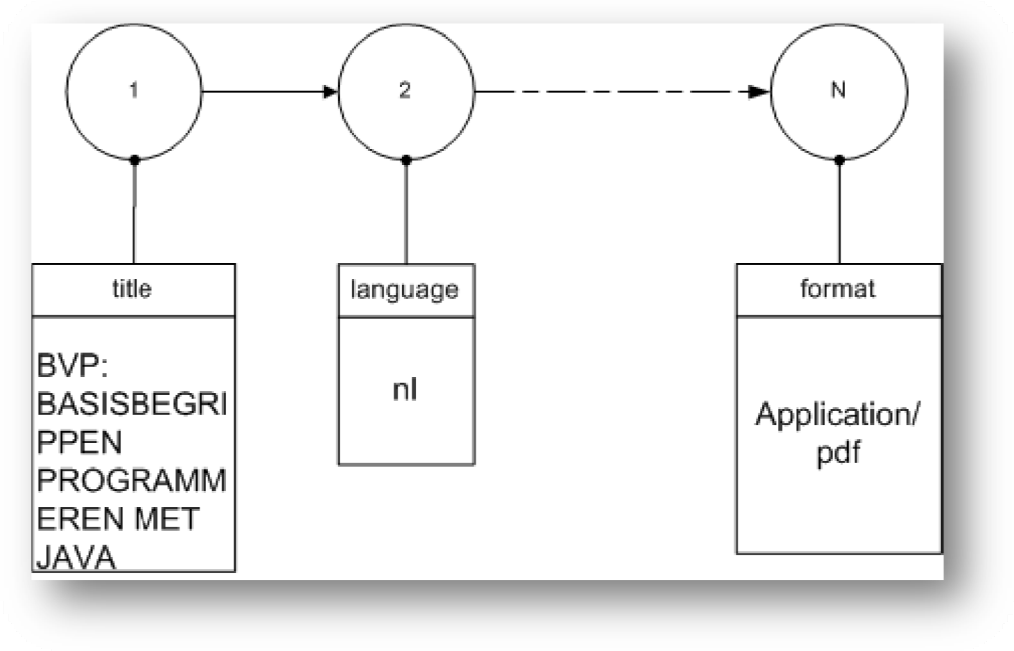


Figura 4.7 Ejemplo de tipo de datos utilizado en SQI v2.0 con información

Como se puede ver en la Figura 4.7 se trataría de un conjunto de tuplas enlazadas. De esta forma el análisis de la información por parte de los sistemas encargados de ello, será una tarea mucho más efectiva. Entre las principales ventajas de utilizar este nuevo sistema se destacan:

- Al ser un listado enlazado de datos, la ordenación de sus valores por cualquiera que sea el campo o campos claves será una tarea trivial, puesto que no será necesario mover la información de lugar como ocurría antes, solamente será necesario cambiar los enlaces que hay entre los nodos. Es más, de esta forma se podrán tener varios tipos distintos de ordenaciones sin necesidad de mover ni un solo dato, de esta forma se le podrá presentar al usuario los datos ordenados por varios factores, sin que con ello penalice el tiempo empleado en su ordenación.
- Gracias también a este tipo de datos, la eliminación de información también será una tarea sencilla y rápida, bastará con modificar un enlace y "puentear" el nodo a eliminar.
- Al utilizar enlaces se podría pensar en mejorar los sistemas de búsqueda dentro de grandes cantidades de objetos de aprendizaje. Se podría optar en este caso por utilizar una variación de este sistema, modificando la estructura anteriormente presentada para que funcione como un árbol binario de búsqueda, de esta forma la búsqueda de objetos de aprendizaje será una tarea sumamente rápida y efectiva.

A continuación se muestra en la Figura 4.8 cómo podría ser una estructura de datos como la planteada en el punto anterior. Cada uno de los nodos representaría un objeto de aprendizaje (LO) descrito a través del uso de una estructura de datos como la descrita en las anteriores imágenes.

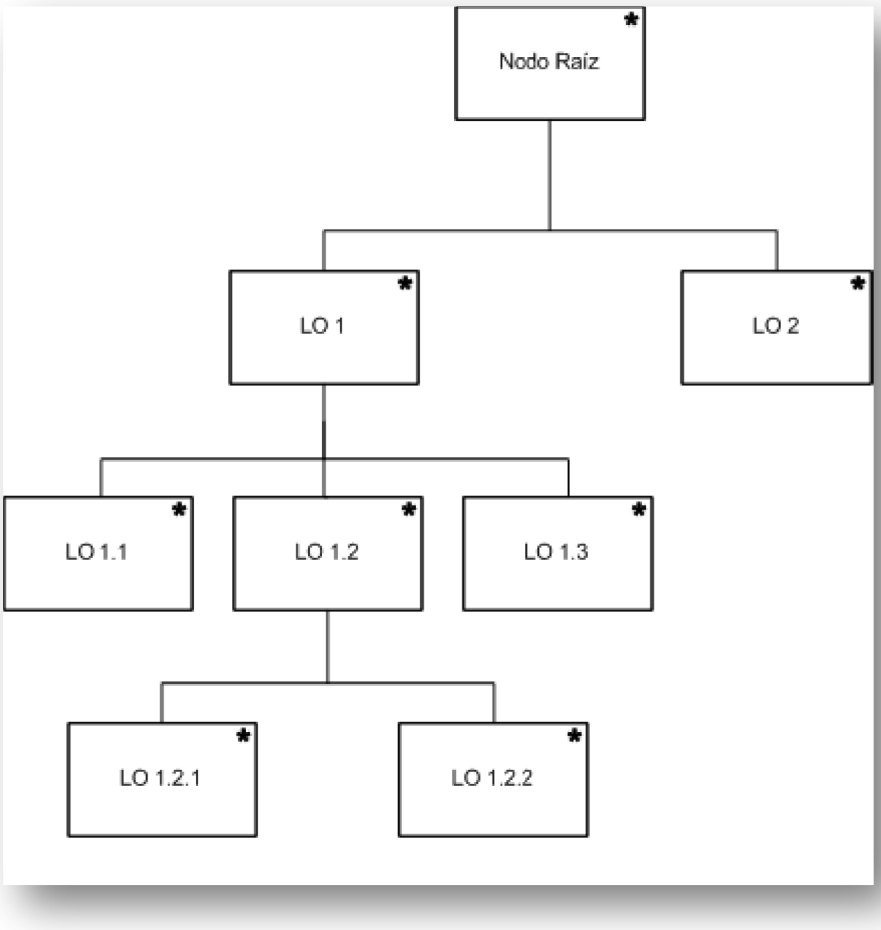


Figura 4.8 Ejemplo de estructura de árbol binario en SQI v2.0

Para la implementación de la modificación de este método se podría utilizar cualquier estructura de datos con las que cuentan los lenguajes de hoy en día, así por ejemplo:

- En Java se podría optar por la clase *java.util.List* que ya implementa este listado de información.
- En C++, C# o VB.NET se podría recurrir a conjuntos de punteros a estructuras o clases como Systems.Collections.Generic.List o System.Collections.ArrayList

entre otras, que también podrán implementar perfectamente esta estructura de datos.

- En C se podría recurrir al igual que antes a punteros a estructuras (en este caso no se podrían utilizar clases ya que C es estructurado y no orientado a objetos como en el anterior caso).

Independientemente del lenguaje seleccionado no habría problema en implementar esta solución ya que la gran mayoría de los sistemas de esta tipología utilizan servicios Web en su implementación, y como se vio anteriormente, estos independizan el uso de protocolos, lenguajes o sistemas utilizados, por lo que la interoperabilidad entre sistemas no se verá paliada en este caso, y si se conseguirá una estructuración mucho más eficiente de los datos manejados y con ello una mayor efectividad en las búsquedas, análisis y eliminaciones de datos.

Propuesta 2: Añadir un método de descarga del objeto de aprendizaje en SQI

La siguiente modificación que se plantea está relacionada con añadir un método que posibilite la descarga del material docente junto con su metainformación. La interfaz SQI que se conoce en la actualidad presenta una serie de métodos relacionados con la conexión con un repositorio o sistema de búsqueda distribuido, para posteriormente realizar una búsqueda de información en el. Sin embargo, quedaría el último paso: una vez que se ha encontrado el o los objetos de aprendizaje que coinciden con los parámetros marcados durante la búsqueda, lo lógico es proceder ahora a su descarga.

La especificación con la que cuenta actualmente SQI no define ningún método para procesar la descarga del material. Si recurrimos nuevamente a la herramienta SQITester, podremos ver que uno de los campos que describen el objeto de aprendizaje contiene la URL del objeto. También son interesantes los metadatos que contienen el tamaño del archivo o el tipo de archivo del que se trata. Sin embargo, estos campos serán rellenados por el usuario que describa el objeto, por lo que no existe la total certeza que estén presentes en todos los objetos de aprendizaje. Con la especificación que existe hoy en día, la única posibilidad que existe para procesar la descarga del objeto de aprendizaje será crear un hipervínculo sobre la URL especificada (en caso de que esta exista en la metainformación del objeto de aprendizaje).

En la Figura 4.5 se mostraban los resultados de la búsqueda realizada sobre el repositorio ARIADNE, resaltando el metadato que contiene la URL de enlace al objeto de aprendizaje. También se puede ver justo encima el tamaño del objeto y el tipo de objeto (application/pdf en este caso).

No parece ser la mejor forma de procesar la descarga de un objeto de aprendizaje, ya que este debería ser descargable independientemente de que se haya decidido rellenar un metadato o no. Así pues, se plantea la posibilidad de incorporar un método que posibilite la descarga de un objeto de aprendizaje junto con su metainformación. Así pues su interfaz sería la que se enuncia en la Figura 4.9.

Method name	**downloadResource**	
Return type	Byte[]	
Parameters	*Name*	*Type*
	targetSessionID	String
	resourceID	String
	downloadType	int
Fault	NO_SUCH_SESSION INVALID_RESOURCE_ID DOWNLOAD_IO_ERROR DOWNLOAD_MODE_NOT_SUPPORTED METHOD_FAILURE	

Figura 4.9 Propuesta de formato del nuevo método de descarga de recursos de SQI v2.0

Como información del método se destaca que:

- El método devolvería o bien un array de bytes (tipo estándar utilizado en la representación de ficheros físicos en todas las plataformas de programación) o un contenido vacío en caso de utilizar técnicas avanzadas en el envío de datos a través de mensajes SOAP como se verá a continuación.
- El parámetro *targetSessionID* representa el identificador de la sesión que previamente se estableció con el repositorio.

- El parámetro *resourceID* representa el identificador del recurso que se desea descargar, el cual permite conocer a cada recurso de manera inequívoca dentro del conjunto de recursos con el que cuente cada repositorio.
- El parámetro *downloadType* representa el tipo de descarga que se desea realizar del recurso. Con valor 0, se procederá a una descarga utilizando el array de bytes comentado anteriormente y con valor 1, se procederá a enviar el fichero fuera del mensaje SOAP.

En cuanto a los posibles errores manejados por la interfaz, se destacan:

- "NO_SUCH_SESSION" en caso que el parámetro *targetSessionID* sea inválido.
- "INVALID_RESOURCE_ID" en caso que el identificador especificado no coincida con ninguno de los que maneja el repositorio.
- "DOWNLOAD_IO_ERROR" si la descarga del recurso provocara un error de entrada/salida.
- "DOWNLOAD_MODE_NOT_SUPPORTED" si el modo de descarga no estuviera soportado, o si el parámetro *downloadType* tuviera un valor incorrecto.
- "METHOD_FAILURE" si la operación falla por otro motivo.

Para la descarga del material se podrá optar por dos mecanismos:

- Utilizar un array de bytes: este sería el método de descarga del contenido más sencillo, pero menos eficiente. El recurso en cuestión se transformaría en un flujo de bytes que se transmite desde el origen al destino. El tipo de datos utilizado no supondría en ningún caso un problema de interoperabilidad entre sistemas, ya que este tipo de datos está presente en todas las plataformas de programación. Sin embargo es poco eficiente, ya que bloquearía el flujo de comunicación entre cliente y servidor hasta que el objeto se haya transmitido por completo, por lo que resultaría ineficiente para transmitir recursos muy pesados.
- Utilizar técnicas avanzadas de transmisión de datos a través de mensajes SOAP: Las últimas versiones del protocolo SOAP admiten la posibilidad de enviar datos utilizando *Streaming* de bytes, es decir los datos se van transmitiendo poco a poco y así no es necesario bloquear el proceso en destino hasta recibir todo el recurso. Existe una especificación que data de Enero de 2005 que promueve el uso de este tipo de técnicas, se denomina MTOM (SOAP Message Transmission Optimization Mechanism) [2005].

Gracias a esta especificación se podrá transmitir el contenido del recurso fuera del mensaje SOAP en vez de dentro como se haría en el caso de utilizar arrays de bytes.

Evidentemente tendrá que existir un acuerdo en este sentido para que ambas partes utilicen la misma técnica de envío/recepción de datos, por este motivo se plantean dos posibilidades para la realización de este cometido.

Estos son los dos cambios que se plantean para conseguir la especificación de una interfaz de consulta, que ahora sí que cuenta con todas las características necesarias para garantizar no solo la interoperabilidad entre sistemas, sino que ahora también se hará de manera estructurada y eficiente.

4.2.3. Propuesta de incorporación de técnicas semánticas para las búsquedas de información

Los sistemas de búsqueda actuales recurren exclusivamente a la coincidencia o no del término o términos de búsqueda en origen y destino. En el caso de los sistemas de búsqueda de objetos de aprendizaje como los presentados a lo largo de esta Tesis, buscan un concepto dentro de la metainformación del objeto, si lo encuentran dan el objeto de aprendizaje por válido (este tipo de búsqueda sería utilizando PLQL de nivel 0). En sistemas de búsqueda como LORS, ARIADNE o ADL-R, además se pueden hacer búsquedas por más de un campo de metainformación (nivel 1 de PLQL), pero aún así se buscan exclusivamente coincidencias en varios campos en lugar de uno solo, por lo que este sistema no es todo lo efectivo que debiera ser. Las búsquedas actuales que realizan los sistemas son como ilustra la Figura 4.10.

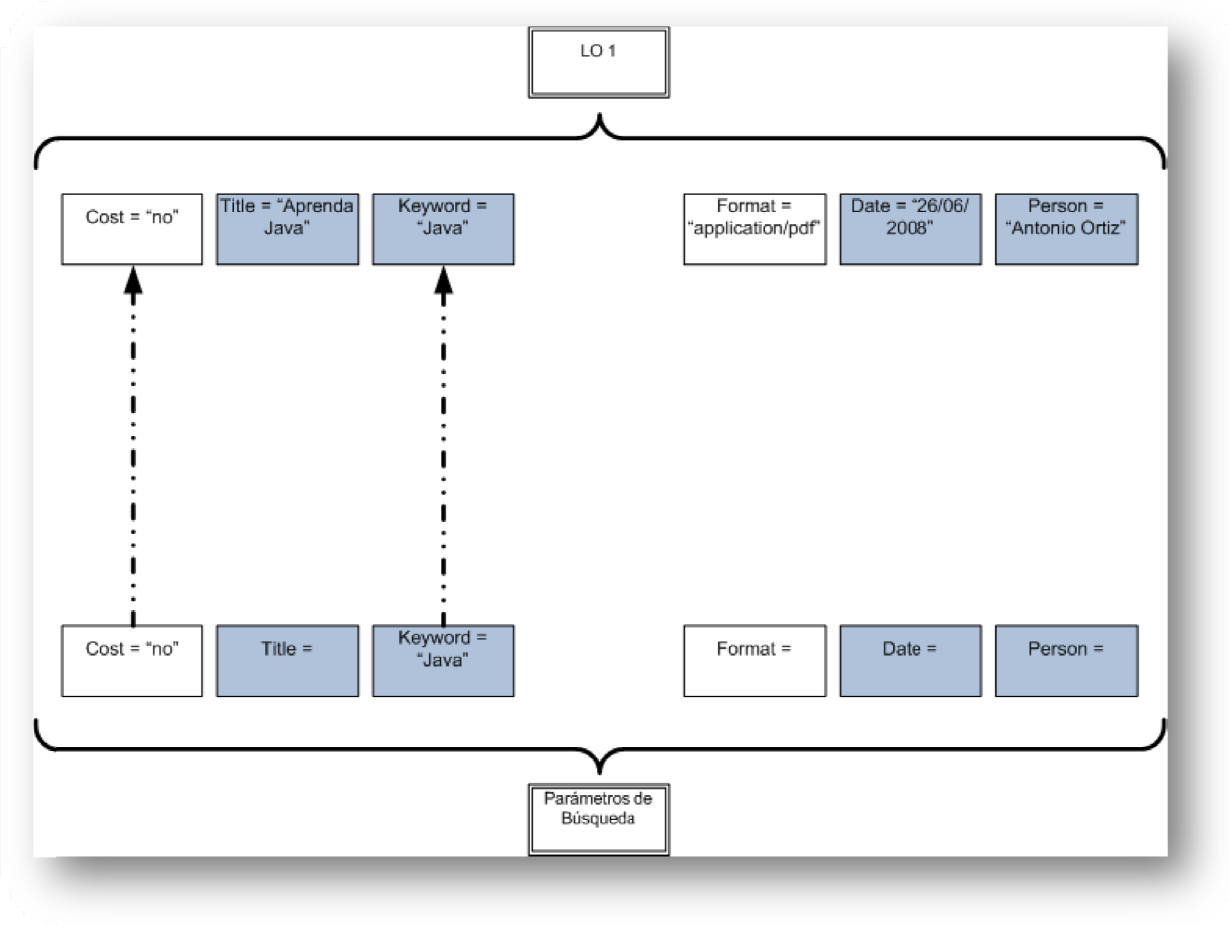


Figura 4.10 Procedimiento en los sistemas de búsqueda actuales

Como se puede ver en la Figura 4.10, los sistemas actuales de búsqueda analizan qué campos rellena el usuario en su búsqueda y, en función de esos campos, realiza la búsqueda equivalente en la metainformación de todos los objetos de aprendizaje. Es importante destacar en este punto dos factores importantes:

- Si se realizaran búsquedas en formatos de metainformación distintos, se realizarían las transformaciones pertinentes como ya se ha comentado anteriormente.
- El estándar de metadatos de objetos de aprendizaje utilizado por defecto ha sido una versión resumida de LOM [IEEE, 2002]. El resumir los metadatos que esta especificación aporta ha sido por motivos de eficiencia, ya que en la mayoría de los casos, los autores nunca rellenan todos los metadatos, así que se optó por utilizar los más representativos [Friesen, 2004]. Independientemente de esta elección, como ya se comentó anteriormente, el sistema es perfectamente adaptable a cualquier conjunto de metadatos que se quiera utilizar, por lo que en un futuro, este aspecto del sistema podrá ser fácilmente modificado.

Es interesante destacar dos tipos diferentes de metadatos. Se puede ver en la Figura 4.10, dos tipos diferentes de recuadros simulando metadatos. Los que aparecen con el fondo oscuro, representan aquellos valores en los que el usuario puede introducir la información que desee, es decir no están tipados. Por otro lado, hay otros que aparecen con el fondo claro; estos representan metadatos que pueden contener unos valores predefinidos, como por ejemplo los que representan:

- Códigos de lenguajes o países: Existen unos códigos predefinidos que identifican a cada lenguaje o país, así por ejemplo el código que representa al lenguaje Español sería el código “es”.
- Formatos de ficheros: Existe también unos tipos predefinidos de formatos de archivos. Estos se conocen con los tipos MIME (Multipurpose Internet Mail Extensions). En 1991 la IETF (Internet Engineering Task Force) comenzó a desarrollar esta norma y desde 1994 todas las extensiones MIME están especificadas de forma detallada en diversos documentos oficiales disponibles en Internet. MIME está especificado en seis RFCs (Request For Comments): RFC 2045, RFC 2046, RFC 2047, RFC 4288, RFC 4289 y RFC 2077.
- Coste por el uso de un objeto de aprendizaje: Este metadato podrá tener los valores “yes” o “no” en función de si el objeto de aprendizaje en cuestión lleva cierto coste asociado para poder realizar un uso del mismo.

- Copyright: La especificación LOM también cuenta con un metadato encargado de determinar si un objeto de aprendizaje lleva derechos de autor u otro tipo de restricciones. Para especificar esto, se podrán definir dos posibles valores “yes” o “no”.

Todos estos metadatos (y algunos otros) no son un problema para el sistema, ya que los valores que se les pueden asignar están predefinidos. El problema viene cuando se realicen búsquedas por metadatos en los que tanto el autor como el sistema de búsqueda pueden incorporar cualquier tipo de contenido; ahí se pueden producir problemas en cuanto a la interpretación de los términos. En los siguientes apartados se detallará cómo dar solución a esta problemática.

Campos de metadatos sin valores tipados

Como se ha comentado anteriormente, la principal problemática reside en aquellos metadatos en los que no se predefinen unos posibles valores, como por ejemplo el título, las palabras clave, el autor, la descripción, etc.

Los sistemas actuales de búsqueda se limitan a verificar coincidencias en los términos, en vez de analizar el significado de los mismos. Así si por ejemplo un usuario introdujera en un buscador Web la frase “vuelos para mañana por la mañana”, el término “mañana” tiene dos significados, temporalización horaria y temporalización diaria, por lo que no se podría limitar solamente a identificar coincidencias en cuanto a términos. Para poder solucionar este problema se introducen ontologías, es decir, relaciones de vocabularios con sus significados en un dominio dado. El añadir semántica en las búsquedas realizadas en la Web se conoce con el nombre de Web Semántica. Por lo tanto el cometido de este apartado es aplicar esta técnica en los sistemas de búsqueda de objetos de aprendizaje para poder dotarles de estas características.

El primer punto será definir determinados dominios de conocimiento a través de ontologías. Para ello se utilizará un lenguaje de los descritos en el apartado del Estado del Arte (Capítulo 2). Este lenguaje será OWL (Ontology Web Language) [W3C, 2004c]. A través del programa Protégé [2009] se puede visualizar de maneara gráfica este tipo de ficheros. Se muestra en la Figura 4.11 una primera aproximación de una ontología creada sobre lenguajes de programación, basada en la ontología OntoGLOSE [Hilera et al., 2005].

La ontología OntoGLOSE incluye más de 1500 conceptos sobre ingeniería del software y más de 200 relaciones entre dichos conceptos. Para este ejemplo, se ha optado por desarrollar la ontología con otro sistema de visionado, ya que Protégé solamente muestra relaciones "is-a" de manera gráfica y en este caso se necesitaba mayor nivel de detalle. Sin embargo para el desarrollo original de la ontología sí que se ha utilizado este entorno.

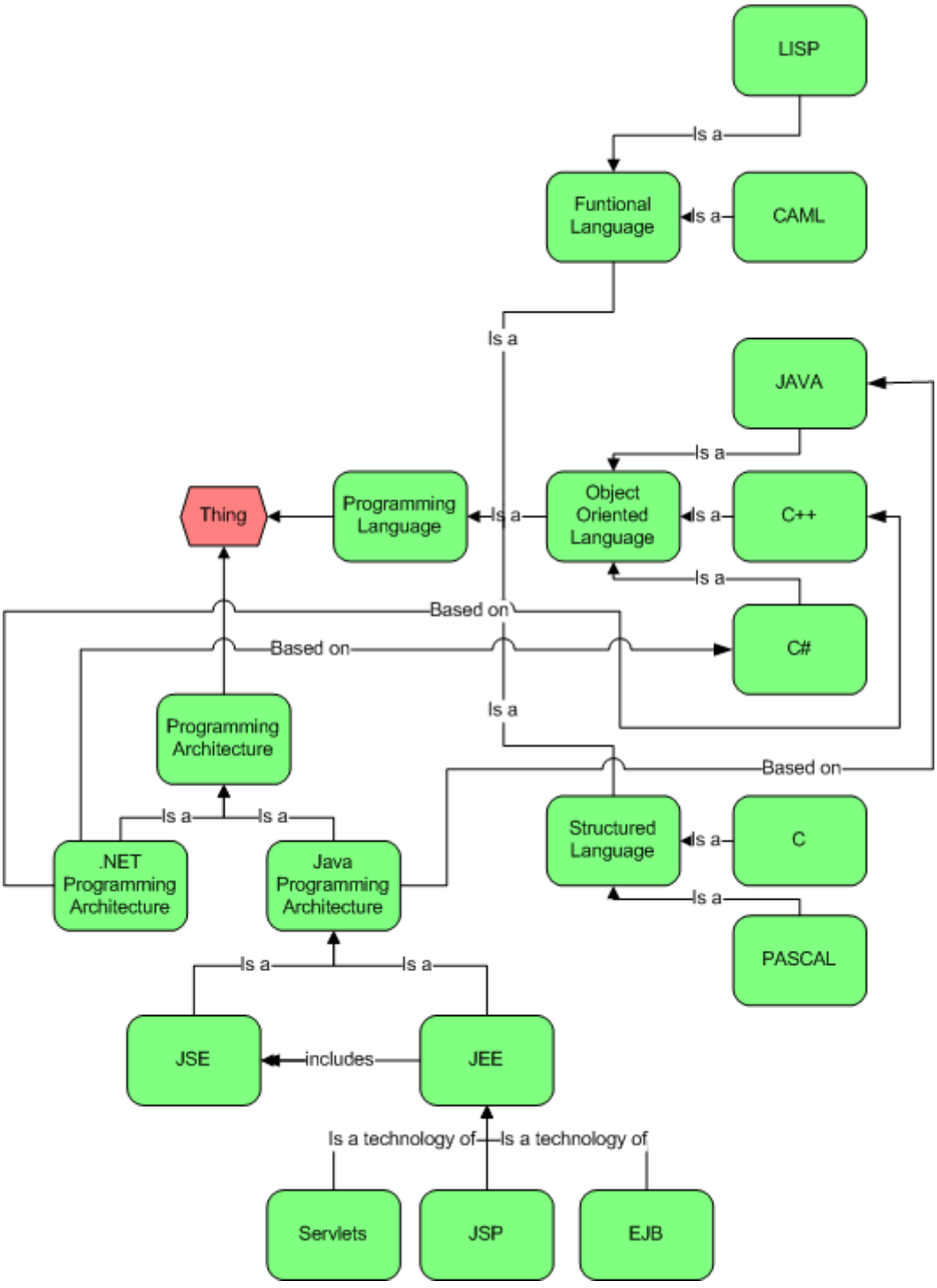


Figura 4.11 Ontología sobre lenguajes de programación

Se puede ver en la Figura 4.11 cómo se definen diferentes tipos de lenguajes de programación (denominados clases dentro de una ontología). Estas clases están relacionadas con la clase “Programming Language” a través de relaciones “is-a” (denominadas propiedades dentro de una ontología). Gracias a estas relaciones se puede saber que un “lenguaje orientado a objetos” es un “Lenguaje”. Se pueden continuar deduciendo relaciones, así como que “Java” es un “lenguaje orientado a objetos” y que por extensión también se podría decir de él que es un “lenguaje”.

Además de la estructura de lenguajes de programación, se puede ver otra estructura dentro de la ontología, y es la correspondiente a “Arquitecturas de Programación”. Aquí se destacan dos arquitecturas, la “arquitectura .NET” y la “arquitectura Java”, y se ve cómo están relacionadas con los términos “Java”, por un lado, y “C++” y “C#” por otro, sabiendo así que “Java” está basado en la “Arquitectura de Programación Java” o que “C++” o “C#” están basados en la “Arquitectura .NET”.

También se puede ver que “JSE” y “JEE” son “Arquitecturas de Programación Java” o que los “Servlets” o las páginas “JSP” es una tecnología de “JEE” que se implementa a través del uso de “Java”.

Gracias al uso de ontologías como la mostrada anteriormente, se definen una serie términos asociados en base a un dominio dado. Realizando una búsqueda en la ontología requerida (la que determinará el dominio a buscar), será posible identificar el concepto en cuestión y realizar así la búsqueda en base al significado del mismo y no en base al término en sí, como realizan hasta ahora los sistemas de búsqueda federada. A continuación se presentan los cambios necesarios que hay que llevar a cabo para adaptar la arquitectura LORA al uso de estas técnicas semánticas, sobre todo en aquellos metadatos en los que no están tipificados los valores que pueden contener.

Propuesta 3: Referencias ontológicas en la descripción de un objeto de aprendizaje

Como se ha indicado anteriormente, el punto de partida lo ponen las ontologías, que son las que determinan los dominios de los diferentes conceptos que se tratan. Por lo tanto, es necesario especificar dentro de la metainformación del objeto docente, un campo que

determine el concepto o conceptos a los que se refiere el objeto, junto con una referencia a la propia ontología, que será la que determine el término relacionado en base al dominio de conocimiento dado.

Para añadir estas dos referencias se ha utilizado el elemento 9 - Classification - con el que cuenta LOM [IEEE, 2002], tal como se recomienda en la norma RDCEO [IMS, 2002]. Este elemento se emplea para definir la categoría a la que pertenece el elemento dentro de algún sistema de clasificación o taxonomía, o en este caso, una referencia ontológica. Se podría definir una taxonomía como un sistema de clasificación compuesto por una jerarquía de taxones anidados; un tesauro como un listado de palabras o términos empleados para representar conceptos interrelacionados, y la ontología sería un tesauro en el que se amplían las relaciones que se pueden dar entre sus componentes y que además está descrito en algún formato que puede ser procesable computacionalmente, generalmente XML.

Los campos definidos (dentro de Classification) son los siguientes:

- 9.1 Purpose. Incluye el propósito por el que se clasifica el objeto docente al que referencia el registro de metadatos. Solo se puede usar un valor del siguiente vocabulario: discipline, idea, prerequisite, educational objective, accessibility, restrictions, educational level, skill level, security level, competency.
- 9.2 Taxon Path. Incluye la ruta taxonómica dentro del sistema de clasificación. Es un elemento compuesto por los elementos Source y Taxon.
    - 9.2.1 Source. Incluye el nombre del sistema de clasificación o fuente.
    - 9.2.2 Taxon. Define un elemento particular dentro del sistema de clasificación. Se entiende que es un nodo dentro de esa taxonomía que tiene una etiqueta o término. Es un elemento compuesto formado por Id y Entry.
        - 9.2.2.1 Id. Representa el identificador del taxón de acuerdo al sistema de clasificación. Idealmente debería ser único dentro de ese sistema, y puede incluir una ruta o camino dentro de alguna estructura jerárquica tal y como defina este.
        - 9.2.2.2 Entry. Es la etiqueta de texto o nombre del taxón.

LOM ofrece bastante libertad en las implementaciones con respecto a los valores y al uso que se puede hacer de los campos. Concretamente, el elemento Classification puede ser usado para encuadrar el objeto docente dentro de cualquier sistema de clasificación. Dado que en este caso se emplea para ubicarlo dentro de un sistema de búsqueda semántica, a continuación se ofrece una descripción de la interpretación de cada uno de los mencionados elementos LOM dentro del sistema propuesto:

- 9.1 Purpose. Como se indicó anteriormente, de todos los elementos del vocabulario solo se consideran tres: 'discipline', 'prerequisite' y 'education objective'. Estos son los necesarios para definir los tipos de relaciones necesarias.
- 9.2 Taxon Path. Dado que el propósito es enlazar una serie de conceptos que definan semánticamente al objeto docente, se incluirán todos aquellos datos para encontrar e identificar la ontología correspondiente.
    - 9.2.1 Source. Incluirá una URL con la dirección de la ontología en cuestión.
    - 9.2.2 Taxon. Sus subelementos especificarán cómo acceder al concepto dentro de la ontología.
        - 9.2.2.1 Id. Representa un identificador único para referenciar al concepto.
        - 9.2.2.2 Entry. Es el concepto propiamente dicho.

A continuación se muestra un fragmento de código de cómo quedaría descrito un objeto de aprendizaje en lo que a la parte semántica se refiere:

```
<lom:classification>
        <lom:purpose>discipline</lom:purpose>
        <lom:taxonPath>
                <lom:source>
                        <lom:string language="en">
                                http://www.cc.uah.es/ontologies/programming.owl
                        </lom:string>
                </lom:source>
                <lom:taxon>
                        <lom:id>1.2.1</lom:id>
                        <lom:entry>
                                <lom:string language="en">
                                        Java
                                </lom:string>
                        </lom:entry>
                </lom:taxon>
                <lom:taxon>
                        <lom:id>2.2.2</lom:id>
                        <lom:entry>
                                <lom:string language="en">
```

```
                                        JEE
                                    </lom:string>
                                </lom:entry>
                            </lom:taxon>
                            <lom:taxon>
                                <lom:id>2.2.2.3</lom:id>
                                <lom:entry>
                                    <lom:string language="en">
                                        EJB
                                    </lom:string>
                                </lom:entry>
                            </lom:taxon>
        </lom:taxonPath>
        <lom:taxonPath>
                            <lom:source>
                                <lom:string language="en">
                                    http://www.cc.uah.es/ontologies/OntoGLOSE.owl
                                </lom:string>
                            </lom:source>
                            <lom:taxon>
                                <lom:id>1.2</lom:id>
                                <lom:entry>
                                    <lom:string language="en">
                                        Object-Oriented language
                                    </lom:string>
                                </lom:entry>
                            </lom:taxon>

</lom:classification>
```

Gracias a la inclusión de los metadatos anteriormente comentados, será posible una futura búsqueda semántica en base a los conceptos que definen dicho objeto de aprendizaje. En el ejemplo anteriormente comentado, se marcaban varios conceptos de una ontología dada (la que ilustraba la Figura 4.11), pero sería posible marcar otros como también muestra el ejemplo anterior a través del uso de un término de otra ontología desarrollada por el departamento de Ciencias de la Computación de la Universidad de Alcalá, denominada OntoGLOSE [Hilera et al., 2005].

Gracias a esta técnica, si por ejemplo un usuario realiza una búsqueda por el campo título y pone “Excepciones Java” si se utilizara un sistema solo sintáctico, solamente podría buscar aquellos objetos de aprendizaje cuyo título sea “Excepciones” o “Java”, por lo que se devolverían objetos que no le interesarían al usuario, como por ejemplo uno que tuviera como título “Curso Java Servlets”. Sin embargo, si se utilizaran técnicas semánticas como las que se acaban de describir, el sistema, accediendo a la ontología anteriormente presentada, identificará que las excepciones forman parte de la arquitectura JSE de Java, de forma que podrá realizar una búsqueda en todos aquellos objetos de aprendizaje que tengan no solo por título “Excepciones” sino por objetos de

temática “Cuso JSE Java” porque sabrá que las excepciones forman parte de la arquitectura estándar de Java.

Los buscadores por lo tanto a partir de ahora deberán buscar dentro de la ontología aquellos valores de campos que no tengan sus contenidos tipificados, para así saber el significado de la terminología utilizada y realizar búsquedas mucho más precisas en base a significados y no solo a términos.

4.2.4. Propuesta de composición de cursos como agregación de objetos docentes

Por último se presenta en este apartado otra nueva funcionalidad que puede ser añadida a los sistemas de búsqueda federada. Se trata de la posibilidad de permitir la búsqueda y posterior descarga de cursos completos de aprendizaje. Para realizar esta tarea, será necesario basarse nuevamente en ontologías, de forma que se identifiquen los módulos que componen cada uno de los cursos, y así poder obtenerlos de forma individual.

Una vez que el usuario indique el tipo de curso que desea obtener, comenzará una búsqueda federada como la que se detalla a continuación. Si por ejemplo el usuario desea un curso completo de JEE (arquitectura empresarial de Java), accediendo a una ontología relacionada con los conceptos de Servlets, JSP, EJB, Servicios Web, Apache Struts y Spring, el sistema podrá realizar una búsqueda federada semántica para obtener todos estos módulos de forma individual. Según se vayan obteniendo todos estos objetos de aprendizaje individuales, se obtendrá la información docente que albergan y se irá ordenando según indique la ontología. Evidentemente se podrá obtener más de un objeto de aprendizaje para cada módulo, o que no se obtenga ningún objeto de aprendizaje para uno o varios módulos en concreto. En el caso de obtenerse más de un objeto para cada módulo se realizarán combinaciones de todos ellos, indicando para cada combinación el porcentaje de coincidencia resultante para que el usuario pueda ver cuál es el que mejor se adapta a sus necesidades. Todo este proceso, será mostrado de forma exhaustiva en el capítulo de validación de la arquitectura (capítulo 5).

A continuación se puede ver en la Figura 4.12, a grandes rasgos, cómo sería el proceso resultante:

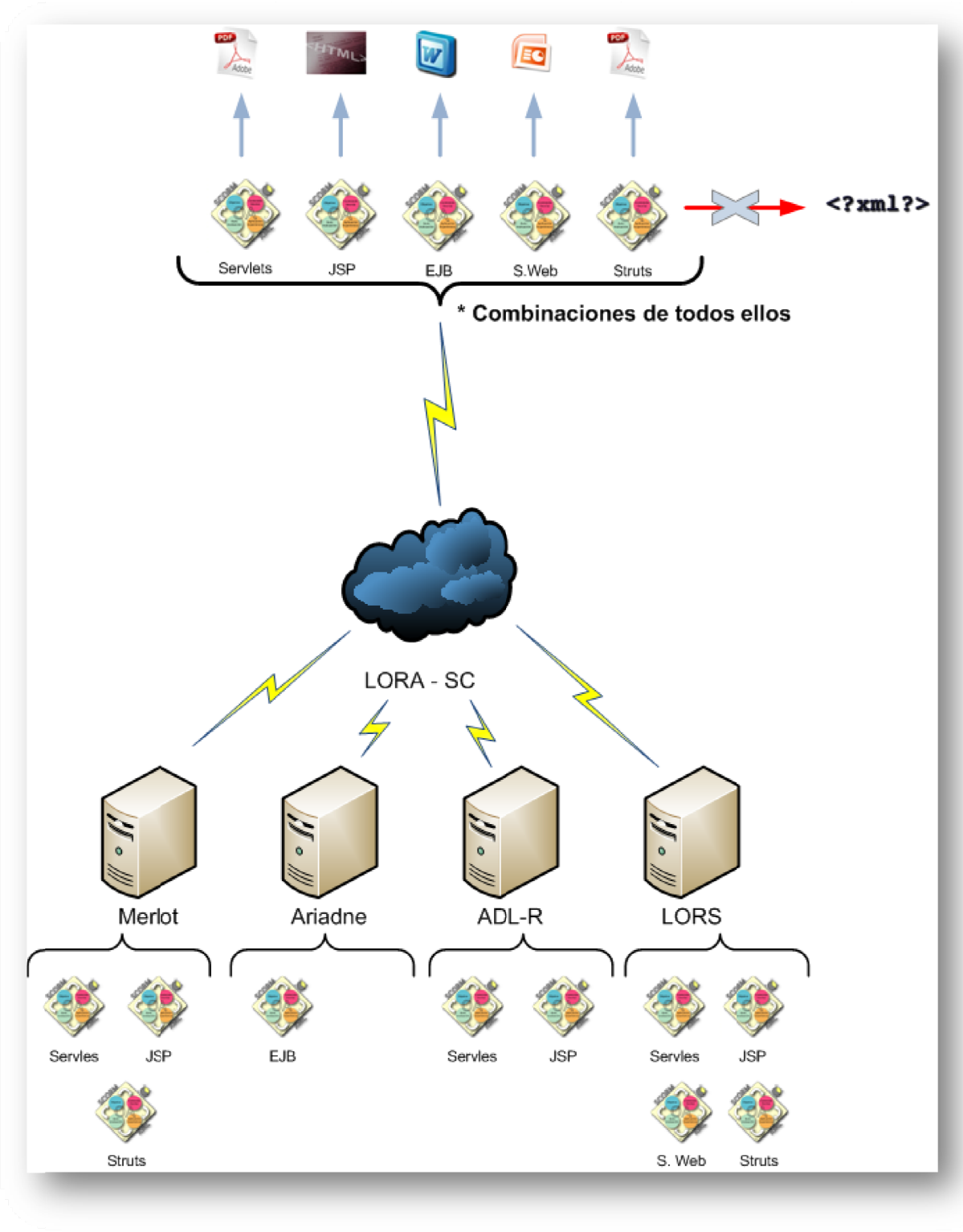


Figura 4.12 Composición automática de cursos (paso 1)

Una vez que se han recuperado los objetos de aprendizaje individuales (aquellos que se hayan podido recuperar, ya que como muestra la Figura 4.12 faltaría el objeto de aprendizaje para el módulo de Spring), se obtendrá de cada uno de ellos el contenido educativo, desperdiciando la metainformación. Con todo este contenido educativo se

generará un nuevo objeto de aprendizaje, al que habrá que incorporarle una nueva metainformación que lo describa. Para su composición se utilizará la especificación IMS CC (Common Cartridge) [IMS, 2008]. Se puede ver en la Figura 4.13 cómo sería este proceso.

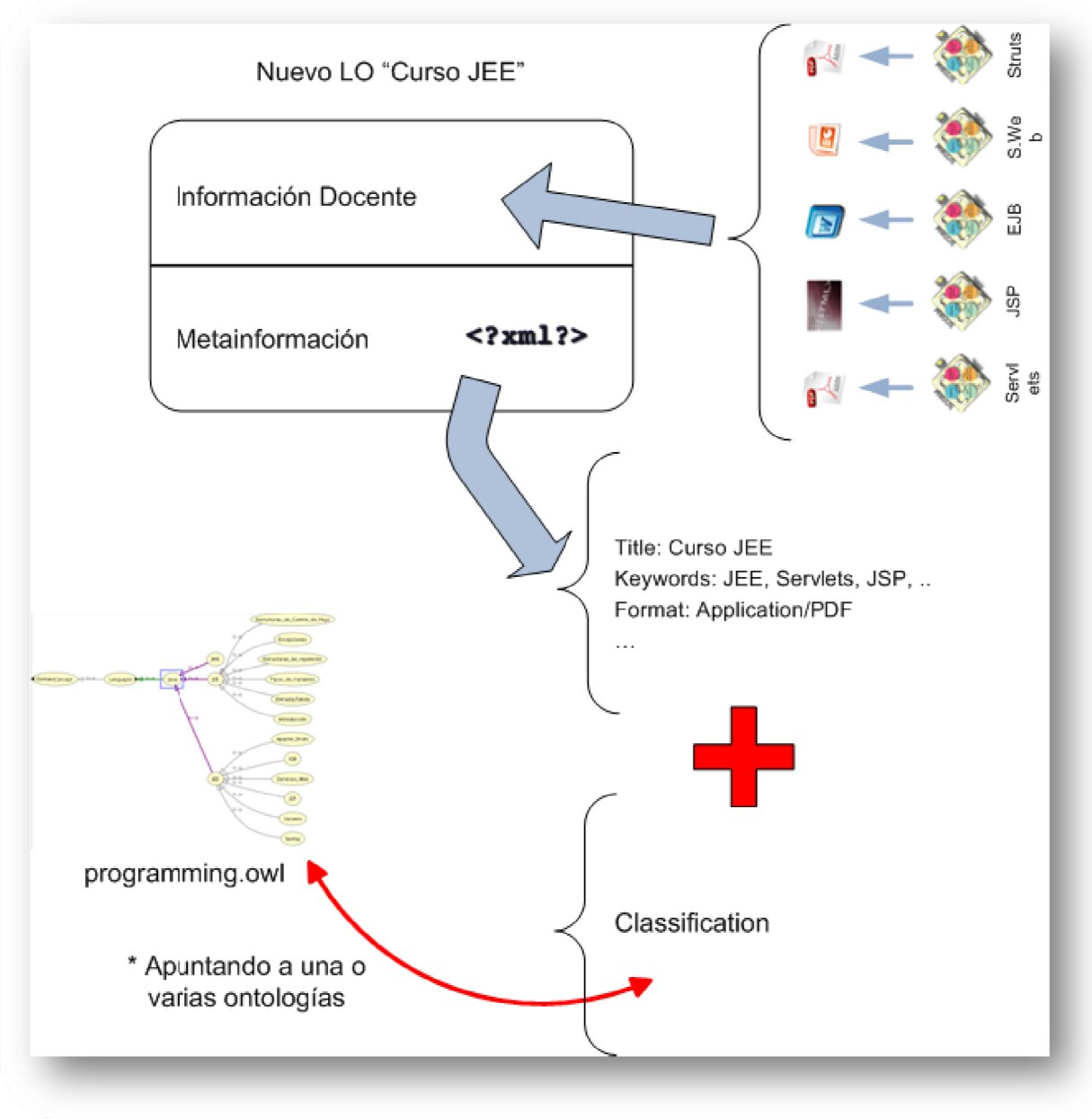


Figura 4.13 Composición automática de cursos (paso 2)

Cuando se hayan juntado todos los materiales docentes de los objetos de aprendizaje individuales, se deberá generar el archivo de metainformación correspondiente al nuevo curso. Para ello, se deberá analizar el tipo de metadato en cada

caso para darle un valor correspondiente, así por ejemplo para el título, le corresponderá el nombre del curso, en este caso “curso JEE”, para las palabras clave, lo mejor sería establecer varios valores, uno por cada uno de los módulos que lo componen, “Servlets, JSP, EJB, …” así como el nombre del curso “JEE” o “Java”, para el formato se haría el mismo proceso, añadir todos los formatos que componen el curso, “application/pdf, application/msword, application/vnd.ms-excel, …”, etc.

La norma a seguir será para aquellos campos que puedan contar con más de un valor, se incorporarán todos los valores correspondientes a cada uno de los objetos individuales que lo componen, mientras que para aquellos que pueden tener solo un valor, se optará por añadir el campo siempre y cuando todos los objetos individuales tengan el mismo valor, o no añadirlo en caso contrario. Un ejemplo podría ser el campo que especifica si el objeto tiene o no copyright; si la mitad de los objetos individuales lo tuvieran y la otra mitad no, es mejor dejar este campo vacío que aportar información incorrecta. Evidentemente además de la metainformación que identifica al objeto de aprendizaje habrá que añadir la referente a la ontología (u ontologías), la cual determinará el dominio de los términos que en ella se detallan, según se especificó en el anterior apartado.

A través de esta nueva característica en los sistemas de búsqueda federada será posible la obtención de cursos completos, beneficiando aún más la reutilización de objetos de aprendizaje y el uso de este tipo de arquitecturas. En este nuevo buscador será posible especificar otros parámetros, como que por ejemplo solamente se obtengan contenidos en un determinado idioma, sin coste, sin copyright, etc. siempre bajo un modelo de búsqueda semántica, de forma que los resultados obtenidos se aproximen lo máximo posible a las necesidades del usuario en cada momento.

El proceso completo de obtención de un curso completo será el siguiente:

- El usuario indicará los datos de búsqueda, en este caso:
  - Lenguaje: es
  - Copyright: no
  - Términos a buscar dentro de las ontologías seleccionadas
- Con esos datos, lo primero que hará el sistema es buscar la ontología (u ontologías) adecuada, en este caso la ontología de programación.

- Con la ontología se buscarán los módulos (conceptos) que debería contener el curso buscado.
- Se realizará una búsqueda federada para conseguir todos los módulos que compondrán este curso.
- Cuando se hayan obtenido todos los objetos de aprendizaje individuales, se realizarán combinaciones de todos ellos, obteniendo de cada uno de ellos la información docente que albergan y se construirá un nuevo objeto de aprendizaje como composición de todos ellos.
- Se generará la metainformación de ese nuevo objeto de aprendizaje siguiendo las siguientes directrices:
    - Para los metadatos que admitan más de un valor, se añadirán todos los valores distintos que tuvieran los objetos de aprendizaje individuales que conforman el nuevo curso.
    - Para los metadatos que no admitan más de un valor se rellenará si todos los objetos de aprendizaje tenían el mismo valor para ese metadato o se dejará sin rellenar en caso contrario.
- Una vez generado el objeto de aprendizaje se le posibilitará su descarga al usuario. El listado que le mostrará el sistema al usuario lo hará en función del nivel de coincidencia de los mismos con respecto a los términos marcados por el usuario en las distintas ontologías seleccionadas. Es posible que se le notifique que falta alguno de sus módulos si no se hubiera podido conseguir de ninguno de los repositorios asociados al sistema de búsqueda.

4.2.5. Niveles de la arquitectura

Una vez descritas todas las propuestas que se han realizado en los apartados anteriores, se muestra ahora el resultado de la arquitectura que da cabida a las mismas. En este caso, la arquitectura presentada no muestra amplios cambios con respecto a la mostrada en la Figura 4.1, ya que las propuestas realizadas no afectan a los niveles de la arquitectura anterior. Se muestra a continuación en la Figura 4.14 las capas o niveles con las que cuenta la nueva arquitectura:

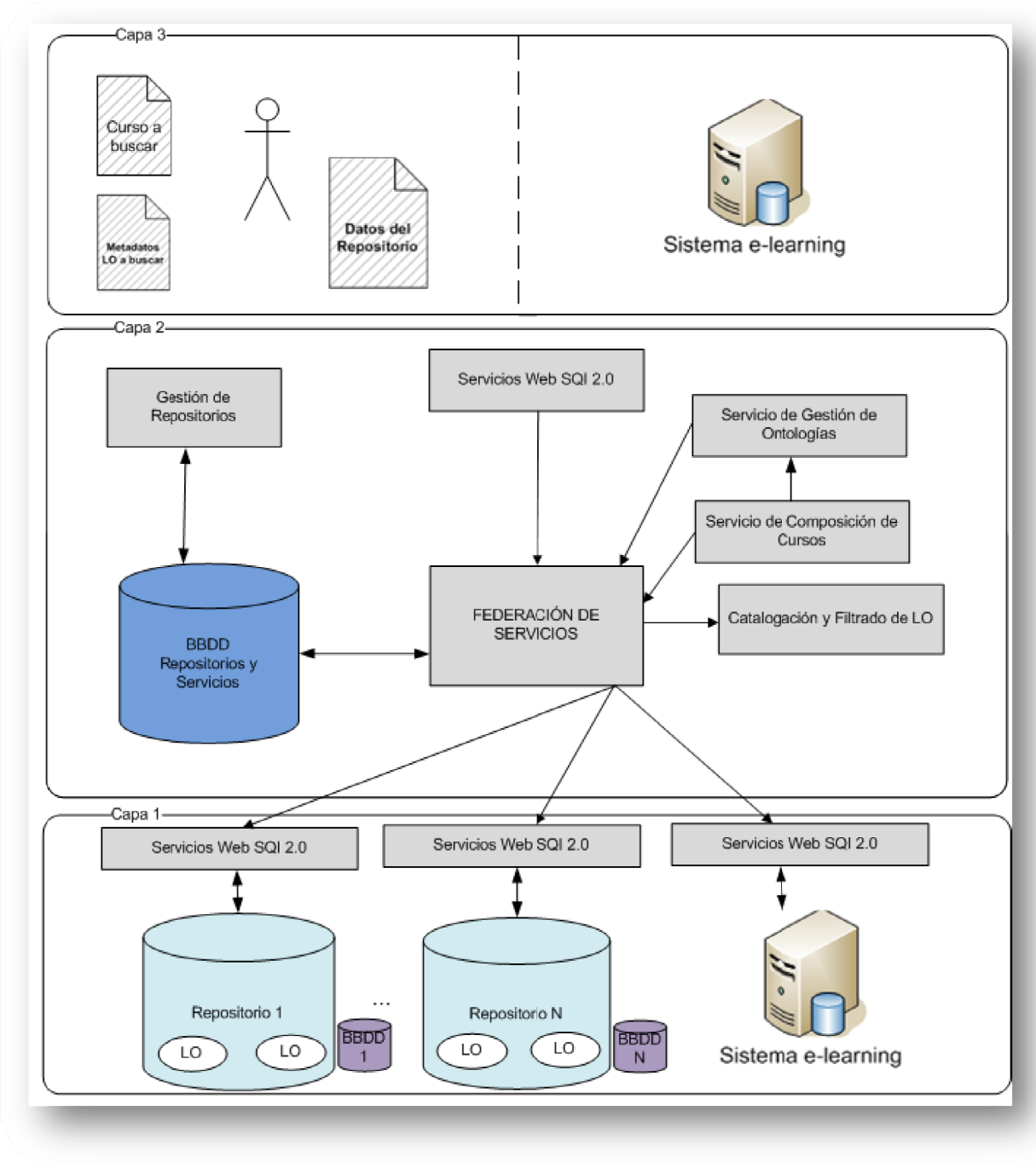


Figura 4.14 Arquitectura LORA - SC

En cuanto a los niveles presentados en la Figura 4.14:

- Capa 1: Sigue siendo la capa más baja de la arquitectura y la ligada al repositorio. Como cambio se destaca que el servicio Web asociado al repositorio ahora contará con un método de descarga que posibilitará la descarga del objeto de aprendizaje en cuestión y que los métodos de consulta han sido modificados siguiendo la propuesta descrita anteriormente; así el servicio Web asociado al repositorio se ha denominado "Servicio Web SQI 2.0" haciendo referencia al uso de la nueva propuesta de interfaz de consulta SQI 2.0.
- Capa 2: Esta capa sigue siendo la que posibilitará la búsqueda federada de objetos docentes. Como cambios se destacan:
    - La incorporación, al igual que en el caso anterior, del "servicio Web SQI 2.0" para que de esta forma la arquitectura LORA-SC pueda recibir de igual forma peticiones de consulta sobre los repositorios asociados a la misma.
    - La incorporación de un servicio de gestión de ontologías. Este servicio será el encargado de acceder a la ontología requerida, en función de la temática buscada, y así obtener el listado de clases y propiedades presentes en la misma y permitir la búsqueda semántica de términos en las consultas realizadas a través de esta arquitectura.
    - La incorporación de un servicio de composición de cursos. Este servicio será el encargado de recibir las peticiones de cursos completos y gracias al uso del servicio de gestión de ontologías, obtendrá los contenidos mínimos con los que debería contar un curso de esas características. Posteriormente a través del servicio de búsqueda federada, realizará la búsqueda individual de todos los objetos de aprendizaje que compongan el curso, los filtrará y ordenará con el servicio de catalogación y filtrado y por último se lo devolverá al usuario.
- Capa 3: Esta capa ha permanecido inalterada. Tan solo habrá que añadir las interfaces necesarias para permitir la búsqueda de cursos.

En cuanto a las cuatro propuestas realizadas afectan a:

- Propuesta 1 – Modificación de los métodos de consulta de SQI: Esta propuesta afecta a los métodos de consulta de SQI. Se deberán modificar los métodos de consulta que ofrecía cada repositorio, por lo tanto tan solo hay

que modificar internamente el servicio Web que se ubicaba justo encima de cada repositorio (capa 1). Es un cambio interno, y no estructural.

- Propuesta 2 – Añadir un método de descarga del objeto de aprendizaje en SQI: Al igual que en el caso anterior, se trata de un cambio interno y no estructural. Esta propuesta consiste en añadir un nuevo método a los que ya contiene el servicio Web anteriormente modificado (también capa 1). En la anterior versión de la arquitectura presentada en la Figura 4.1 se mostraba en la capa 2 un servicio Web encargado de procesar la descarga del objeto de aprendizaje en cuestión; ahora este método está situado al mismo nivel que el resto de métodos SQI.
- Propuesta 3 – Referencias ontológicas en la descripción de un objetos de aprendizaje: Como tal esta propuesta no afecta en nada al diseño de la arquitectura, ya que es a nivel del conjunto de metadatos que se usarán para describir a cada objeto de aprendizaje; sin embargo, viene derivada de la propuesta de incorporar técnicas semánticas a las búsquedas de información, por lo que será necesario incorporar un servicio de gestión de ontologías (capa 2) encargado de acceder a la ontología en cuestión e identificar las propiedades y clases que se definan en ella para darle un tratamiento semántico a la búsqueda en cuestión.
- Propuesta 4 – Composición de cursos como agregación de objetos docentes: Esta propuesta afecta de igual forma a la capa 2 de la arquitectura, ya que es necesario incorporar un servicio de composición de cursos, de forma que ayudándose del servicio de búsqueda federada y del servicio de gestión de ontologías pueda ofrecer al usuario la posibilidad de descargar cursos completos formados por objetos de aprendizajes individuales.

4.2.6. Conclusiones de la arquitectura LORA-SC

Gracias al uso de todas las especificaciones y recomendaciones presentadas a lo largo de estos capítulos, así como las características y técnicas descritas, se ha diseñado una arquitectura completa de reutilización de objetos de aprendizaje a la que se le ha denominado **LORA-SC** (Learning Object Reusability Architecture – Semantic & Composite). Entre sus nuevas características se destacan:

- Define una nueva interfaz de consulta, la cual amplia la funcionalidad ofrecida por SQI y a la que se ha denominado SQI 2.0. Como principales novedades ofrece:
  - Un método de consulta que ofrece unos resultados más estructurados, lo cual facilitará enormemente su análisis y procesado, siguiendo la misma filosofía de ofrecer interoperabilidad e independencia entre sistemas.
  - Un método de obtención del material educativo. En este caso se trata de un método completamente nuevo y que finaliza el proceso de reutilización de objetos de aprendizaje, ya que de poco sirve una especificación de consulta si luego no existe un método que posibilite su descarga.
- Define una nueva forma de realizar consultas, basándose en el significado de los términos y no solamente en los términos propiamente dichos. Para ello, se define una ontología (o varias) asociada al objeto de aprendizaje a través del campo *Classification* con el que cuenta LOM [IMS, 2002]. Gracias a esta ontología se podrá conocer el significado de los términos bajo un dominio dado y así realizar búsquedas más efectivas.
- Posibilita la búsqueda y descarga de cursos completos de aprendizaje. Para realizar esta tarea, el sistema se basará nuevamente en ontologías para determinar los módulos que componen un curso completo y así realizar una búsqueda federada de todos ellos. Una vez obtenidos generará un nuevo objeto de aprendizaje, creándole además su metainformación correspondiente.

Por último se presenta en la Tabla 4.1 un resumen en el que se pueden identificar todas las especificaciones y recomendaciones seguidas en cada uno de los apartados o niveles de la arquitectura, ya que uno de los objetivos de la misma, es que estuviera basada en estándares.

Categoría	Sub-Categoría	Estándar
LO		
	Metadatos	LOM [IEEE, 2002] (modificado)
	Empaquetado	IMS CC [IMS, 2008]
	Recursos	IMS RLI [IMS, 2004b]
	Vocabularios	IMS VDEX [IMS, 2004c]
Búsquedas		
	Interoperabilidad Repositorios	IMS DRI [IMS, 2003a]
	Interoperabilidad en Consultas	SQI (modificado a la v2.0) [CEN, 2005]
	Interoperabilidad en la Publicación	SPI [CEN, 2007]
Arquitecturas		
	Framework	IMS AF [IMS, 2003b]
Servicios		
	Intercambio de Datos	IMS LIS [IMS, 2007]
	Interoperabilidad	IMS GWS [IMS, 2005]

Tabla 4.1 Resumen de las especificaciones y recomendaciones utilizadas

En la Figura 4.14 se podían ver las tres capas o niveles que componían su arquitectura. A continuación se describen los estándares o recomendaciones que se siguen en cada caso:

- Capa 1: Para su diseño se ha seguido la recomendación IMS GWS (General Web Services [IMS, 2005]) que determina la forma en la que los servicios Web se construyen e interactúan entre ellos. Estos servicios Web posibilitan el acceso a los objetos de aprendizaje descritos según la especificación IMS LOM (Learning Object Metadata [IMS, 2002]) y empaquetados según IMS CC (Common Cartrigde [IMS, 2008]). Para acceder a estos datos, se utiliza por un lado la interfaz de consulta SQI (Simple Query Interface [CEN, 2005]), y por otro lado la interfaz de publicación SPI (Simple Publishing Interface [CEN, 2007]). SQI es la que determinará los métodos que serán desarrollados en los servicios Web, y que posibilitarán el acceso a la información docente que albergan y SPI los métodos que determinarán la forma en la que se publican los objetos en un repositorio. Para etiquetar estos metadatos y crear categorías de vocabularios se ha utilizado la especificación IMS VDEX (Vocabulary Definition Exchange [IMS, 2004c]), para organizar, describir e intercambiar las listas de recursos que tiene cada objeto de aprendizaje se ha

utilizado IMS RLI (Resource List Interoperability [IMS, 2004b]), y para que todo el intercambio de datos se haga de una manera efectiva se ha utilizado IMS LIS (Learning Information Services [IMS, 2007]).

- Capa 2: En el nivel dos se encuentran los servicios de búsqueda federada, es decir aquellos que utilizando los métodos desarrollados anteriormente obtienen y manejan la información ofrecida por los mismos, por lo que también utilizan la especificación SQI. Al ser también servicios Web, estos deben ser construidos de igual forma en base a IMS GWS. Como esta parte será la encargada de manejar toda la información devuelta por los repositorios, también será necesario el uso de la especificación IMS LIS. Además de los servicios que invocan a los métodos SQI, se desarrollan otros que se encargarán de la catalogación y filtrado de la información, o la gestión de los repositorios asociados al sistema de búsqueda. Por otra parte destacar que esta parte también podrá ser invocada por otros sistemas de búsqueda, por lo que también podrá ser destino de llamadas, así será al mismo tiempo origen y destino de consultas.
- Capa 3: Esta última capa es la que representa la capa de presentación de las aplicaciones que cumplan con la arquitectura propuesta. En ella se encuentran todas las interfaces que dan acceso a las funcionalidades de la aplicación en cada caso. Además de todas las especificaciones anteriormente comentadas se ha seguido la especificación IMS DRI (Digital Repositories Interoperability [IMS, 2003a]) que determina las necesidades que debe cumplir un repositorio para ser interoperable y así facilitar el acceso al contenido que alberga e IMS AF (Abstract Framework [IMS, 2003b]), que aporta una representación abstracta del conjunto de servicios que se deberían utilizar para construir un sistema e-learning en su sentido más amplio, por lo que determinó en gran parte los servicios que se necesitaban.

Se muestra en la Tabla 4.2, un resumen de todas las especificaciones y recomendaciones usadas en las diferentes capas o niveles que componen la arquitectura:

Especificación	Capa 1	Capa 2	Capa 3
IEE LOM	X		
IMS CC	X		
IMS RLI	X		
IMS VDEX	X		
IMS DRI	X	X	X
CEN SQI	X	X	
CEN SPI	X		
IMS AF	X	X	X
IMS LIS	X	X	
IMS GWS	X	X	

Tabla 4.2 Resumen de las especificaciones y recomendaciones usadas en el diseño de la arquitectura de LORA-SC

5. VALIDACIÓN DE LA ARQUITECTURA CON LA IMPLEMENTACIÓN DE UN PROTOTIPO REAL

El objetivo de este capítulo es explicar cómo se han validado las arquitecturas propuestas en el capítulo anterior, mediante la creación de prototipos reales. El resultado de la creación del prototipo ha sido un "**Sistema de Reutilización de Objetos de Aprendizaje (SROA)**" o en inglés "**Learning Objects Reusability System (LORS)**".

Como se presentaron dos arquitecturas en el anterior apartado, a continuación se presentarán dos prototipos funcionales. Realmente el segundo de los prototipos será una evolución del primero, como ya ocurría con la arquitectura.

A grandes rasgos, los objetivos a conseguir con el desarrollo del sistema han sido los siguientes:

- Permitir las búsquedas federadas a través de distintos repositorios distribuidos.
- Definir sistemas de búsqueda semánticos para garantizar una mayor efectividad en las mismas.
- Permitir la búsqueda de cursos completos como conjuntos de objetos de aprendizaje individuales.
- Permitir el registro de diversos repositorios en el sistema, así como su catalogación.
- Permitir la catalogación de los objetos de aprendizaje resultados de la búsqueda federada.
- Permitir la descarga de cada uno de los objetos de aprendizaje resultados de la búsqueda federada.

- Definir un conjunto de metadatos común, pudiendo incorporar determinadas restricciones a los campos que se consideren oportunos, y encapsularlos en un fichero XSD.
- Registrar la trazabilidad de las acciones que se realizan en el sistema mediante un fichero de log.

Por lo tanto, en este capítulo se mostrarán los prototipos generados para solventar los problemas descritos en el capítulo 3, y para validar las arquitecturas propuestas en el capítulo 4. En este capítulo se presentan dos sistemas:

- LORS-SQI: Prototipo que sigue la arquitectura LORA-SQI, la cual es una adaptación de la arquitectura inicial a las nuevas especificaciones existentes para garantizar su completa interoperabilidad con el resto de sistemas.
- LORS-SC: Prototipo que sigue la arquitectura LORA-SC, la cual es una modificación de la arquitectura anterior en la que se añaden técnicas semánticas para posibilitar tanto una mayor eficiencia en las búsquedas como nuevas funcionalidades derivadas de su uso, entre las que se destaca la de permitir la búsqueda de cursos completos de aprendizaje.

5.1. PROTOTIPO INICIAL: LORS-SQI

Para la implementación de esta versión de LORS, al igual que ocurriera con el sistema que sirve de punto de partida de esta Tesis [Otón, 2006], se ha optado por elegir Java como plataforma de desarrollo y MySQL [2009] como gestor de base de datos. Sin embargo se ha modificado el servidor de aplicaciones utilizado. En el caso anterior se utilizaba Apache Tomcat [Apache, 2009] junto con Systinet Server como servidor de servicios Web. Como ya se comentó anteriormente, este servidor ha desaparecido por lo que era necesario también modificar el sistema en este aspecto para poder contar con las últimas actualizaciones que vayan surgiendo. Se ha decidido utilizar el servidor de aplicaciones completo desarrollado por Sun Microsystems, el Glassfish Server [Glassfish, 2009]. Este servidor de aplicaciones proviene del antiguo Application Server de Sun e incorpora además del contenedor de Servlets/JSP la parte de servicios Web, por lo que bajo la misma herramienta será posible el tratamiento de todas las partes de la aplicación.

El sistema final quedaría como ilustra la Figura 5.1.

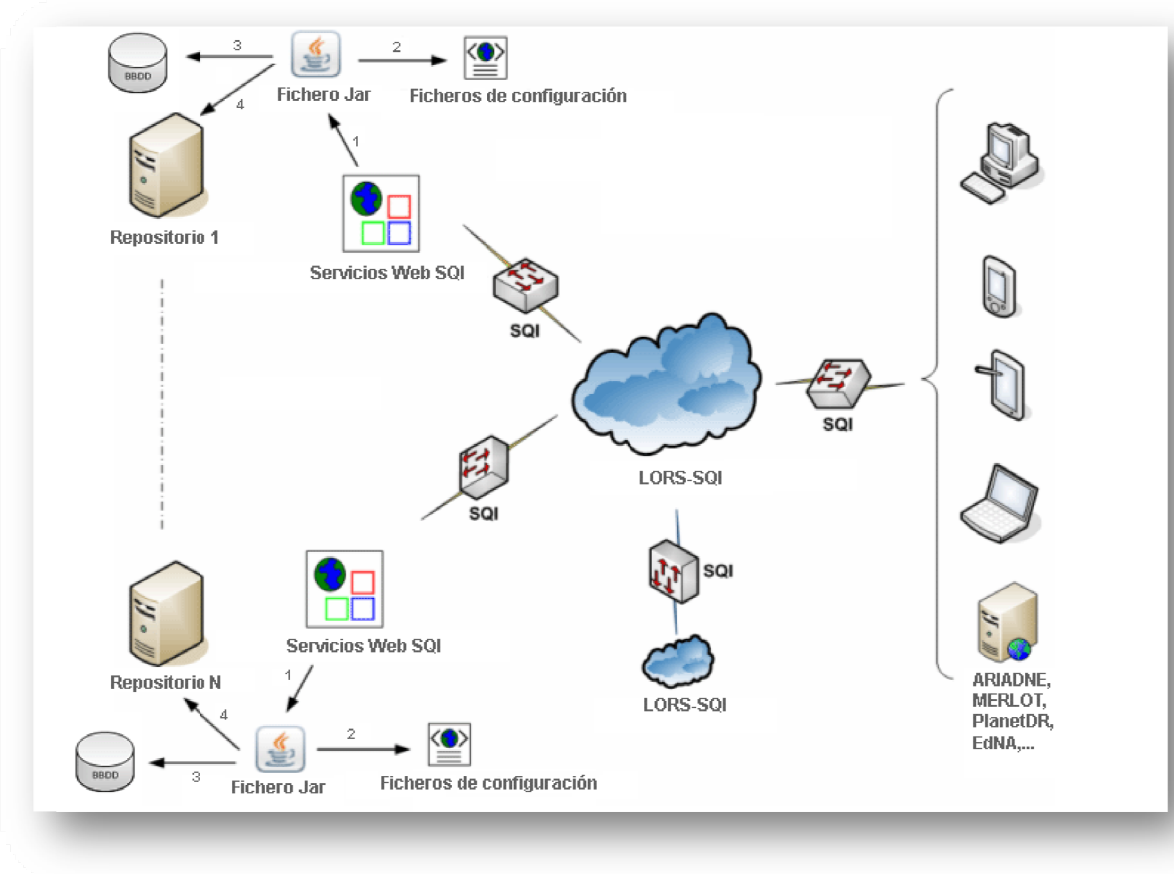


Figura 5.1 Elementos del sistema LORS-SQI

Se pueden diferenciar tres partes fundamentales, en la zona central de la imagen LORS-SQI, el sistema de búsquedas federadas que siguiendo la especificación SQI permite la comunicación, siempre a través de servicios Web SQI, con él tanto a repositorios o sistemas de Internet (parte derecha) como a repositorios implementados o desarrollados según se propone en este trabajo, mediante una librería llamada SQITL-API. En la Figura 5.2 se puede ver más detalladamente las partes de este tipo de repositorios.

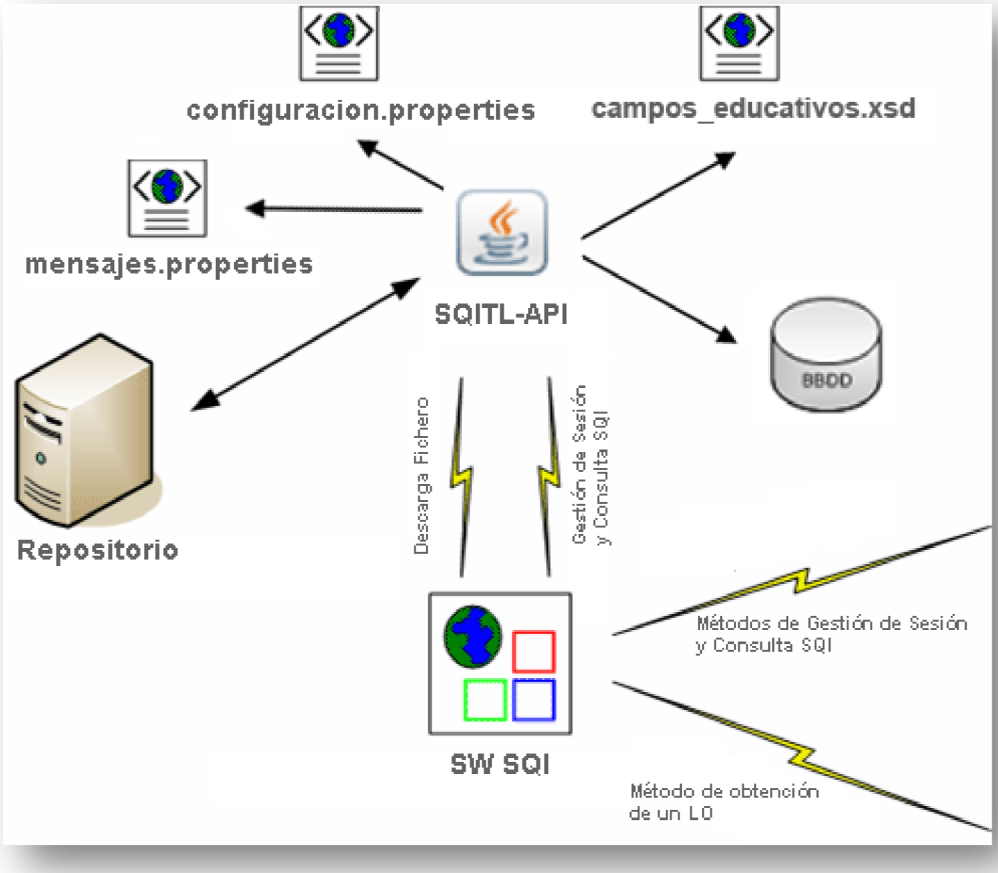


Figura 5.2 Estructura de la librería SQITL-API

Junto con el desarrollo del sistema LORS, se ha elaborado la librería denominada SQITL-API (SQI Tag Library – API). Esta librería representa toda la funcionalidad de un repositorio, de forma que si una entidad externa deseara hacer accesible una serie de objetos de aprendizaje elaborados bajo cualquier estándar de metainformación, podría implementar de manera sencilla la funcionalidad de un repositorio y hacerlos accesibles al exterior. Este repositorio puede recibir las consultas en formato SQI, de forma que es completamente accesible desde el exterior y desde cualquier otro sistema de búsqueda (no solo desde LORS). Para la adaptación de este nuevo repositorio a las particularidades

del cliente en cuestión, tan solo hay que modificar algunos ficheros de configuración externos (también se muestran en la Figura 5.2). Esta es otra de las características con las que cuenta LORS, y que hacen que sea sumamente sencillo poner a disposición (y con ello hacer reutilizables) un conjunto de objetos de aprendizaje.

El sistema desarrollado realiza una adaptación de la interfaz de consulta SQI (Simple Query Interface) [CEN, 2005], como se indicó en el Capítulo 4. Esta interfaz no presenta un método que posibilite la descarga del material educativo, por lo que las arquitecturas desarrolladas 100% conforme a la especificación tan solo pueden presentar enlaces a la página del autor, en las que muchas veces no se posibilita la descarga del material docente, o este no presenta su contenido junto con su metainformación. En el caso de LORS esto no ocurre, el sistema añade un método de descarga del contenido (aparte de los propios de SQI como se puede ver en la Figura 5.2), el que permitirá acceder a la información docente junto con su metainformación. Esta característica también es de vital importancia y a día de hoy, también es exclusiva de este sistema de búsqueda.

El repositorio contará con los propios objetos de aprendizaje que lo componen, así como con una base de datos que mantendrá la metainformación de dichos objetos y que como se ha comentado se utilizará para realizar de forma más eficiente las búsquedas. También se dispone de dos ficheros de configuración ".properties" donde se podrán configurar muchos de los parámetros referentes al repositorio, como se explicará más adelante, y por último una interfaz Web con los servicios SQI, gestión de sesión y de consultas, mediante los cuales será posible la comunicación ya sea con LORS-SQI o con cualquier sistema que implemente la especificación SQI.

El desarrollo de LORS se ha realizado conforme a las más avanzadas herramientas de desarrollo, las que le dotan de la eficiencia, flexibilidad, integridad y adaptabilidad que se esperan de una herramienta de estas características.

En los siguientes apartados se mostrarán el modelo funcional del sistema, el cual ayudará a comprender los requisitos que se van a satisfacer a través de la especificación de casos de uso, el modelo de orquestación de servicios de la aplicación, en el que se podrá ver cómo interactúan entre si los diferentes servicios del sistema y el modelo relacional, en el que se detallan las principales tablas que utilizan los sistemas gestores de base de datos del sistema. No se presenta un modelo de clases, ya que se trata de una

aplicación orientada a servicios, y por lo tanto, toda la funcionalidad del aplicativo estará representada por servicios interconectados; así pues, el modelo de orquestación de servicios aportará más información que la pudiera aportar el modelo de clases.

5.1.1. Modelo funcional

Como se ha indicado anteriormente, la descripción de una arquitectura debe hacerse desde diferentes puntos de vista [IEEE, 2000]. En este apartado se adoptará un enfoque funcional, para describir los requisitos funcionales generales que debe ofrecer el sistema de reutilización de objetos de aprendizaje que se presenta en este apartado (LORS-SQI), así como los actores que participarán en este sistema.

Para describir la funcionalidad soportada por la arquitectura, en primer lugar se realizará una presentación de los requisitos generales que debe satisfacer LORS-SQI, construido siguiendo la arquitectura LORA-SQI anteriormente presentada. Después se detallará cada requisito en un caso de uso donde se explicará con más detalle los pasos a seguir para completar con éxito la funcionalidad requerida. A continuación se presentará el modelo del dominio del problema y por último los diagramas de interacción más importantes.

Especificación de requisitos generales

Como se indicó en el capítulo 3, una de las tareas que primero se deben realizar a la hora de definir la arquitectura de un sistema, es establecer la especificación de sus requisitos. Mediante estos requisitos se podrán conocer los procesos que se llevan a cabo en el dominio del problema y establecer los objetivos que debe cumplir la arquitectura.

A continuación se muestra en la Tabla 5.1 un resumen con los requisitos funcionales más importantes que tiene que cumplir la arquitectura, y por extensión el sistema:

REQUISITOS	DESCRIPCIÓN
R1	Realizar la búsqueda de contenidos
R2	Realizar la catalogación de contenidos
R3	Proporcionar los contenidos solicitados
R4	Realizar el intercambio de contenidos entre repositorios
R5	Realizar la publicación de contenidos
R6	Definir un conjunto de metadatos común, que se adapte a los estándares existentes como LOM, SCORM o IMS
R7	Realizar la conversión de metadatos para adaptarlos a la especificación del solicitante de contenidos
R8	Realizar un catálogo de los repositorios que forman parte de LORS-SQI
R9	Facilitar la integración con plataformas de aprendizaje
R10	Realizar el registro de usuarios
R11	Realizar la autenticación de usuarios
R12	Gestionar los derechos de los contenidos
R13	Registrar la trazabilidad de las acciones que se realizan en LORS-SQI
R14	Permitir la interoperabilidad entre LORS-SQI y cualquier otro sistema de búsqueda basado en SQI.

Tabla 5.1 Especificación de requisitos del sistema LORS-SQI

Actores

Antes de describir cada uno de los requisitos en forma de casos de uso es necesario detallar qué actores utilizarán el sistema. Partiendo de la especificación de IMS DRI (Digital Repository Interoperability) [IMS, 2003a], explicada en el capítulo 2, se recuerda que existían roles predeterminados que podían tomar los usuarios de los repositorios digitales y que, por tanto, serán los mismos que interactúen con LORS-SQI; estos roles son: Bibliotecario, Contribuyente, Prestatarios, Usuarios Casuales, Administrador y

Agentes Software. A continuación se muestran cada uno de estos roles y su cabida en el sistema LORS-SQI:

- El rol de bibliotecario lo tendrá el propio LORS-SQI que será el encargado de la catalogación y gestión de los repositorios y su contenido docente; por lo tanto, representa un rol interno de LORS-SQI.
- El rol de contribuyente se da en LORS-SQI pero de una manera especial. Tal y como se explica en [IMS, 2003a] son aquellas personas que introducen los recursos (objetos de aprendizaje) en el repositorio y se encargan de la generación de metainformación. En este caso es aquel que registra su repositorio en LORS-SQI y por tanto, de forma indirecta, da acceso a su contenido.
- El rol de prestatario también se da en LORS-SQI, ya que son aquellos que adquieren objetos de aprendizaje de los repositorios registrados en LORS-SQI de manera regular, y que suelen personalizar la interfaz con la que interactúan con el repositorio; así como preservar y dejar constancia de las búsquedas realizadas en las sesiones con el repositorio.
- Los usuarios casuales también tienen cabida en LORS-SQI, ya que representan usuarios invitados que pueden tener permisos para buscar, explorar o descargar objetos del repositorio pero sin tener su espacio personalizado propio. Generalmente estos usuarios no tienen que estar registrados como usuarios habituales; por eso SQI posee un método de obtención de la sesión de forma anónima.
- El rol de administrador estará representado en LORS-SQI, ya que tiene la responsabilidad de gestionar los usuarios del repositorio y de establecer la configuración de LORS-SQI.
- El rol de agente software también se da, ya que representa cualquier sistema que puede consultar el repositorio y descargar contenidos de LORS-SQI.

Casos de uso

Para describir cada uno de los casos de uso correspondientes a los requisitos funcionales presentados anteriormente, se va a utilizar la siguiente plantilla, basada en la utilizada por IMS en DRI [IMS, 2003a] para describir sus casos de uso:

CAMPO	VALOR
Funcionalidad	Nombre del caso de uso.
Objetivo	Descripción informal de los objetivos.
Actores	Actores que intervienen: principales y secundarios.
Precondiciones	Condiciones que deben cumplirse para que pueda llevarse a cabo.
Pasos (o Flujos Básicos)	Secuencia de pasos necesarios para que se desarrolle con éxito. Se muestran las interacciones de los actores y las acciones del sistema.
Extensiones (o Flujos Alternativos)	Puntos de extensión.

Utilizando la plantilla anterior, se muestra a continuación cada una de las funcionalidades enunciadas anteriormente:

Funcionalidad	***R1: Realizar la búsqueda de contenidos.***
Objetivo	Buscar contenidos educativos a través de diversos repositorios distribuidos.
Actores	Prestatarios, usuarios casuales, agentes software y bibliotecario.
Precondiciones	Detallar la metainformación de los contenidos a buscar.
Pasos	1. Especificar el estándar utilizado por el cliente (SCORM, IMS, etc.). 2. Seleccionar el tipo de búsqueda que se desea realizar (sencilla o avanzada). a. En el caso de la búsqueda sencilla, rellenar la *query*. b. En el caso de la búsqueda avanzada, rellenar los campos educativos. 3. Realizar la búsqueda en todos los repositorios registrados en LORS-SQI (búsqueda federada). 4. Catalogar los contenidos resultado de la búsqueda. 5. Presentar los resultados de la búsqueda al solicitante.
Extensiones	Si se produce un error en el proceso de búsqueda se le comunicará al usuario. Si no se encuentra ningún contenido que coincida con los parámetros de búsqueda se le comunicará al usuario.

Funcionalidad	***R2: Realizar la catalogación de contenidos***
Objetivo	Catalogar los contenidos educativos de diversos repositorios distribuidos.
Actores	Bibliotecario.
Precondiciones	Haber realizado una búsqueda en la que existan resultados.
Pasos	1. Realizar la búsqueda en todos los repositorios registrados en LORS-SQI. 2. Descartar objetos de aprendizaje duplicados. 3. Descartar objetos de aprendizaje con bajo índice de coincidencia.

	4. Clasificar los objetos de aprendizaje por índice de coincidencia.
Extensiones	Si se produce un error en el proceso de catalogación se le comunicará al usuario.

Funcionalidad	***R3: Proporcionar los contenidos solicitados.***
Objetivo	Poder descargar los contenidos educativos de diversos repositorios distribuidos.
Actores	Prestatarios, usuarios casuales, agentes software y bibliotecario.
Precondiciones	Haber realizado una búsqueda en la que existan resultados.
Pasos	1. Realizar la búsqueda en todos los repositorios registrados en LORS-SQI. 2. Catalogar los contenidos resultado de la búsqueda. 3. Presentar al cliente los resultados de la búsqueda. 4. Descargar los objetos de aprendizaje (convirtiéndolos al estándar de metainformación que utilice el cliente si fuera necesario).
Extensiones	Si se produce un error en el proceso de descarga se le comunicará al usuario.

Funcionalidad	***R4: Realizar el intercambio de contenidos entre repositorios.***
Objetivo	Poder descargar los contenidos educativos de diversos repositorios distribuidos e integrarlos en un repositorio exterior.
Actores	Prestatarios, usuarios casuales, agentes software y bibliotecario.
Precondiciones	Haber realizado una búsqueda en la que existan resultados y haber descargado los contenidos.
Pasos	1. Realizar la búsqueda en todos los repositorios registrados en LORS-SQI. 2. Catalogar los contenidos resultado de la búsqueda. 3. Presentar al cliente los resultados de la búsqueda. 4. Descargar los objetos de aprendizaje (convirtiéndolos al estándar de metainformación que utilice el cliente si fuera necesario). 5. Integrar los objetos de aprendizaje en un repositorio externo.
Extensiones	Ninguna.

Funcionalidad	***R5: Realizar la publicación de contenidos.***
Objetivo	Poder registrar en LORS-SQI un repositorio.
Actores	Contribuyente y bibliotecario.
Precondiciones	Ser el propietario de un repositorio con contenidos educativos.
Pasos	1. Realizar un servicio Web que de acceso al repositorio. 2. Incorporar sus referencias en la Base de Datos correspondiente.
Extensiones	Si se produce un error en el proceso de registro del repositorio se le comunicará al usuario.

Funcionalidad	***R6: Definir un conjunto de metadatos común, que se adapte a los estándares existentes como LOM, SCORM o IMS***
Objetivo	Definir un conjunto de campos de metainformación común a la gran mayoría de especificaciones adaptable y evolutivo que facilite la búsqueda de contenidos y su conversión de un estándar a otro.
Actores	Administrador.
Precondiciones	Ninguna.
Pasos	1. Determinar los campos de metainformación más utilizados e importantes de las diversas especificaciones. 2. Clasificarlos por categoría. 3. Crear un fichero XML (XSD) con las categorías y campos seleccionados. 4. Permitir la modificación del fichero para futuras ampliaciones o especificaciones.
Extensiones	Ninguna.

Funcionalidad	***R7: Realizar la conversión de metadatos para adaptarlos a la especificación del solicitante de contenidos***
Objetivo	Permitir la conversión de los metadatos de un objeto de aprendizaje del estándar utilizado en el repositorio origen al necesitado por el cliente.
Actores	Bibliotecario.
Precondiciones	Ninguna.
Pasos	1. Determinar la especificación de metadatos que usa el cliente. 2. Determinar la especificación de metadatos de cada uno de los objetos de aprendizaje resultado de la búsqueda. 3. Si alguna especificación de los metadatos de los objetos de aprendizaje recuperados no coincide con la utilizada por el cliente, realizar la conversión.
Extensiones	Si se produce un error en el proceso de conversión se le comunicará al usuario.

Funcionalidad	***R8: Realizar un catálogo de los repositorios que forman parte de LORS-SQI***
Objetivo	Catalogar en LORS-SQI los repositorios registrados.
Actores	Bibliotecario.
Precondiciones	El repositorio debe estar registrado en LORS-SQI.
Pasos	1. Cuando se registra un nuevo repositorio en LORS-SQI se deben completar todos los datos necesarios para describirlo. 2. Catalogar los servicios que dan acceso a los repositorios registrados en LORS-SQI. 3. Mantener actualizado el estado de los servicios (activados o desactivados).
Extensiones	En caso de fallo en la localización del servicio cambiar el estado del servicio.

Funcionalidad	*R9: Facilitar la integración con plataformas de aprendizaje*
Objetivo	Dar acceso a LORS-SQI desde una plataforma de aprendizaje.
Actores	Agentes software.
Precondiciones	Ser el administrador de una plataforma de aprendizaje.
Pasos	1. Conocer la URL de LORS-SQI. 2. Dar acceso a LORS-SQI desde la plataforma de aprendizaje. 3. Realizar búsquedas de contenidos o registrar el repositorio de la plataforma.
Extensiones	Ninguna.

Funcionalidad	*R10: Realizar el registro de usuarios*
Objetivo	Registro de los usuarios de LORS-SQI.
Actores	Prestatarios, contribuyente, agentes software y administrador.
Precondiciones	Los nombres de usuario deben ser únicos en LORS-SQI
Pasos	1. Pedir el nombre de usuario y la contraseña al cliente para su registro en LORS-SQI. 2. Comprobar el nombre de usuario para evitar duplicados. 3. Activar la cuenta de usuario con el perfil correspondiente. 4. Realizar acciones de administración básica: Altas, bajas y modificaciones.
Extensiones	Si ya existe un nombre de usuario en LORS-SQI igual al que se quiere dar de alta se le avisará del error.

Funcionalidad	*R11: Realizar la autenticación de usuarios*
Objetivo	Autenticación de los usuarios de LORS-SQI.
Actores	Prestatarios, contribuyente, agentes software y administrador.
Precondiciones	Para autenticarse el cliente tiene que estar registrado.
Pasos	1. Pedir el nombre de usuario y la contraseña al cliente. 2. Comprobar el nombre de usuario y la contraseña en LORS-SQI. 3. Cargar el perfil del usuario.
Extensiones	Si no existe un nombre de usuario en LORS-SQI igual al que se ha introducido se le avisará del error. Si la contraseña no es correcta para el nombre de usuario se le avisará del error. Se permitirá el acceso restringido a usuarios invitados.

Funcionalidad	*R12: Gestionar los derechos de los contenidos*
Objetivo	Determinar que contenidos están sujetos a derechos de autor y si es necesario el pago por su uso.
Actores	Prestatarios, contribuyente, agentes software y bibliotecario.
Precondiciones	Los contenidos deben tener información sobre derechos.
Pasos	1. El contribuyente ha de determinar que contenidos de su repositorio

	están sujetos a derechos de autor. Esto se determinará con los metadatos correspondientes. 2. El bibliotecario cuando recupera los objetos de aprendizaje de los repositorios determina cuales están sujetos a derechos y cuál es su coste. 3. Cuando se presenta la clasificación final de los resultados de una búsqueda se indicarán estos datos y si es necesario se cobrará por los contenidos.
Extensiones	Si un usuario tiene que pagar por el uso del contenido no se permitirá la descarga del mismo hasta que no se hayan activado los mecanismos de pago determinados.

Funcionalidad	***R13: Registrar la trazabilidad de las acciones que se realizan en LORS-SQI***
Objetivo	Guardar un archivo de "log" con todas las acciones que se desencadenan cuando un cliente realiza una acción en LORS-SQI
Actores	Todos.
Precondiciones	Realizar alguna acción en LORS-SQI
Pasos	1. Si no existe el fichero de log se debe crear. 2. Si existe el fichero de log ir añadiendo anotaciones. 3. Ir anotando los resultados de todas las operaciones que se van realizando en LORS-SQI. 4. Realizar acciones de administración básica: Búsqueda, borrado, etc.
Extensiones	Ninguna.

Funcionalidad	***R14: Permitir la interoperabilidad entre LORS-SQI y cualquier otro sistema de búsqueda basado en SQI***
Objetivo	Permitir que cualquier sistema de búsqueda pueda interoperar con LORS-SQI, así como que el sistema de búsqueda de LORS-SQI pueda realizar búsquedas en cualquier sistema.
Actores	Todos.
Precondiciones	Ninguna.
Pasos	1. Desarrollar el sistema conforme a la especificación SQI. a. Desarrollar un SW de gestión de la sesión b. Desarrollar un SW de consultas (síncronas y asíncronas).
Extensiones	Ninguna.

En la Figura 5.3 se representa el diagrama UML de los casos de uso generales y todos los actores implicados. La Figura 5.4 y la Figura 5.5 muestran en detalle los dos casos de uso más importantes, ya que como se señala en [Larman, 2001] lo importante de los casos de uso no son los diagramas sino la explicación textual de los mismos.

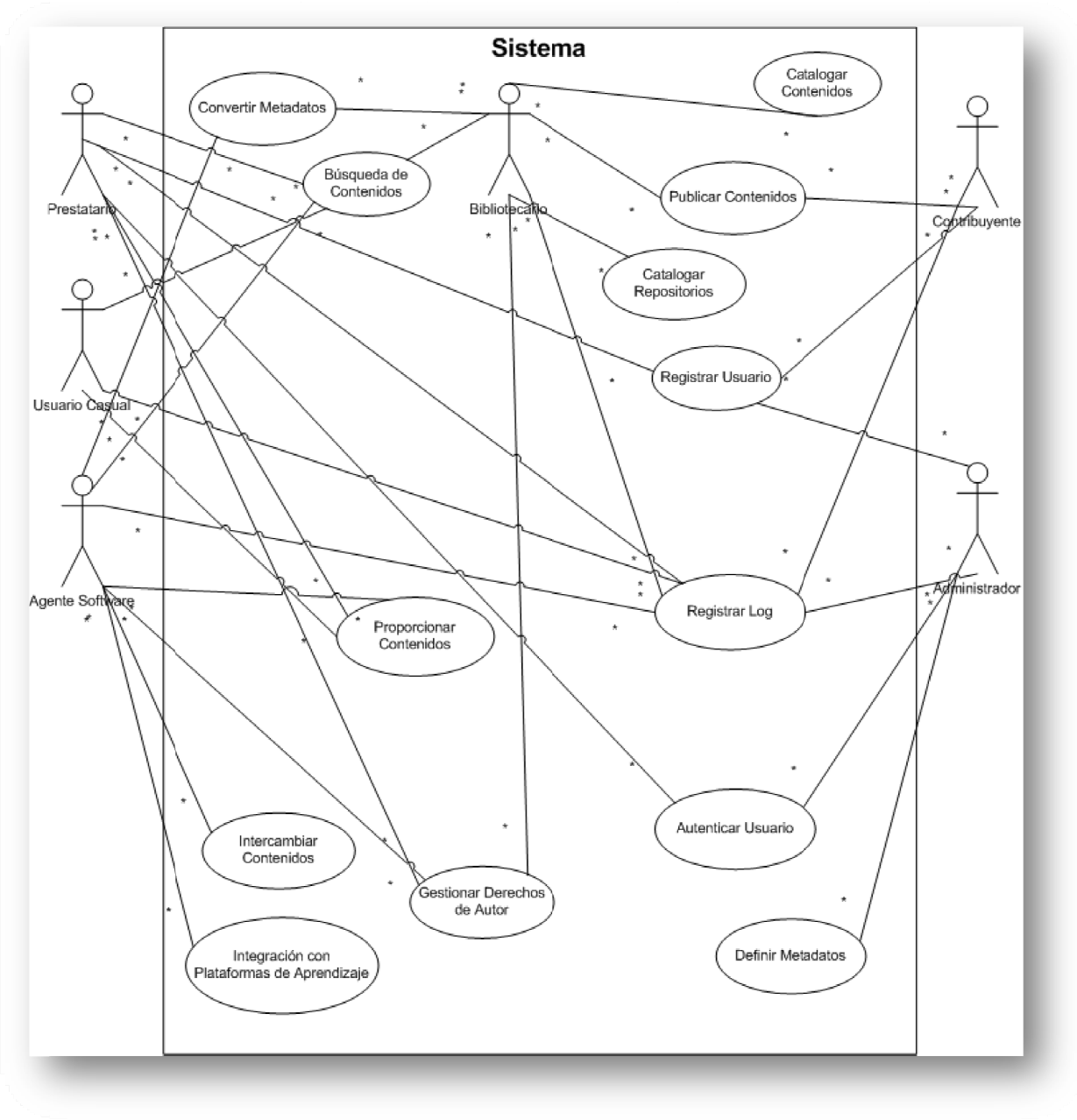


Figura 5.3 Diagrama de casos de uso generales

El primer caso de uso más importante, mostrado en la Figura 5.4 es el de la búsqueda de contenidos en LORS-SQI, correspondiente al requisito R1. Esta acción desencadena una búsqueda federada en diversos repositorios distribuidos a través de un servicio asociado a cada uno de los repositorios que previamente se han registrado en LORS-SQI.

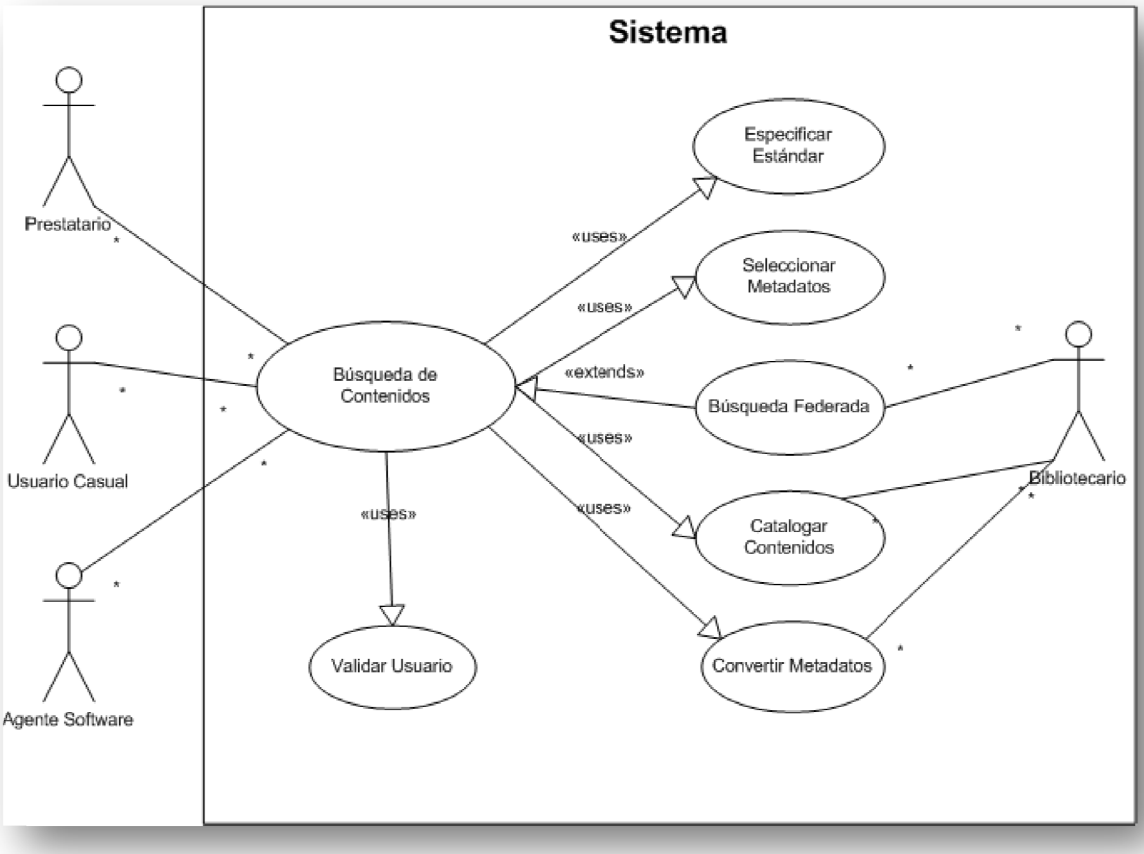


Figura 5.4 Caso de uso: Búsqueda de contenidos

El siguiente caso de uso, mostrado en la Figura 5.5, es el que describe cómo se pueden publicar contenidos en LORS-SQI, correspondiente al requisito R5. Para ello lo que se necesita es que el Contribuyente registre su repositorio en LORS-SQI, donde quedará catalogado. Los pasos previos para poder hacerlo, es haber desarrollado los dos servicios Web que determina la especificación SQI (sesión y consulta).

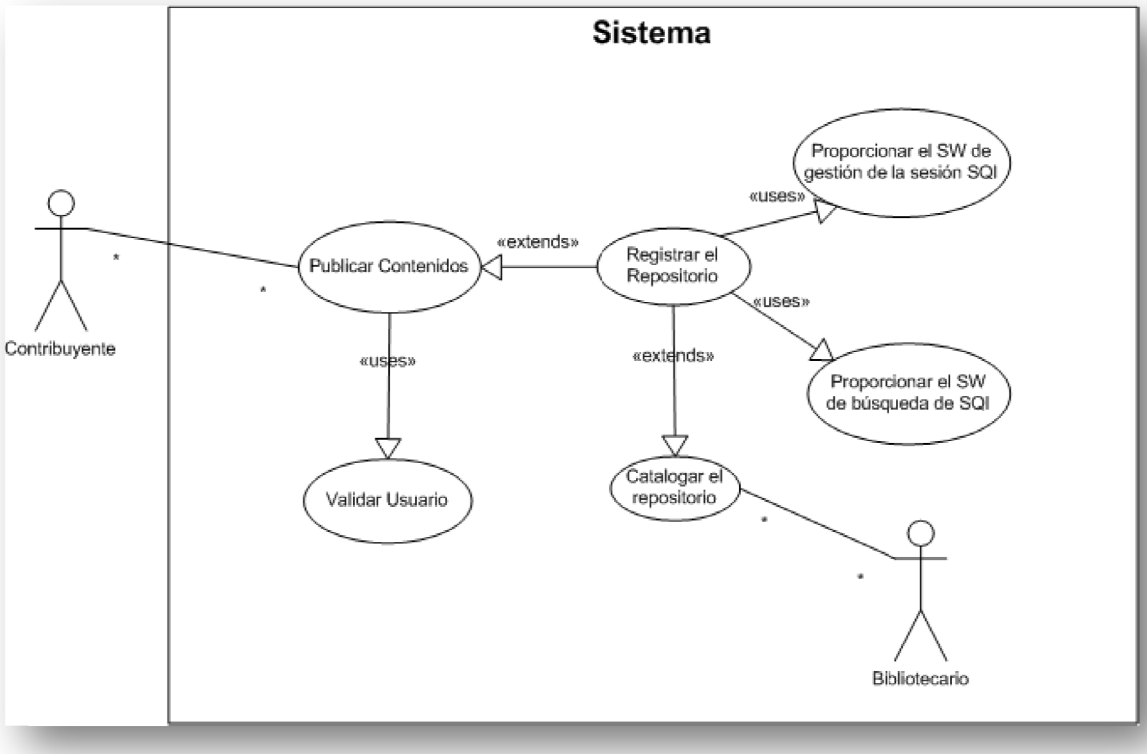


Figura 5.5 Caso de uso: Publicación de contenidos

Modelo del Dominio

El modelo del dominio del problema muestra los conceptos más importantes representados en forma de clases del dominio. En este caso el problema a resolver consiste en la reutilización de objetos de aprendizaje que residen en repositorios distribuidos haciendo interoperables estos repositorios.

Como se aprecia en la Figura 5.6 se ha considerado que el sistema LORS-SQI está compuesto por un conjunto de repositorios que relaciona y un conjunto de usuarios del sistema. Estos repositorios contienen objetos de aprendizaje que a su vez están compuestos por contenidos docentes y metadatos que describen estos contenidos utilizando una especificación determinada como LOM, SCORM o IMS.

Las operaciones principales que se pueden llevar a cabo en LORS-SQI son las de realizar búsquedas y publicar contenidos. La operación de búsqueda desencadena una

serie de acciones detalladas en los casos de uso, cuya principal característica es la de realizar una búsqueda federada en diversos repositorios. La operación de publicación de contenidos en LORS-SQI consiste en registrar y catalogar un repositorio y, por lo tanto, dar acceso a los objetos de aprendizaje que contiene.

Como se puede apreciar en el diagrama cada repositorio tendrá sus propias operaciones de búsqueda y publicación. Por lo tanto cuando LORS-SQI realiza una búsqueda en un repositorio realmente se está llamando a la operación de búsqueda interna del repositorio en cuestión, estandarizada gracias al uso de SQI, el cual permitirá la completa interoperabilidad entre sistemas.

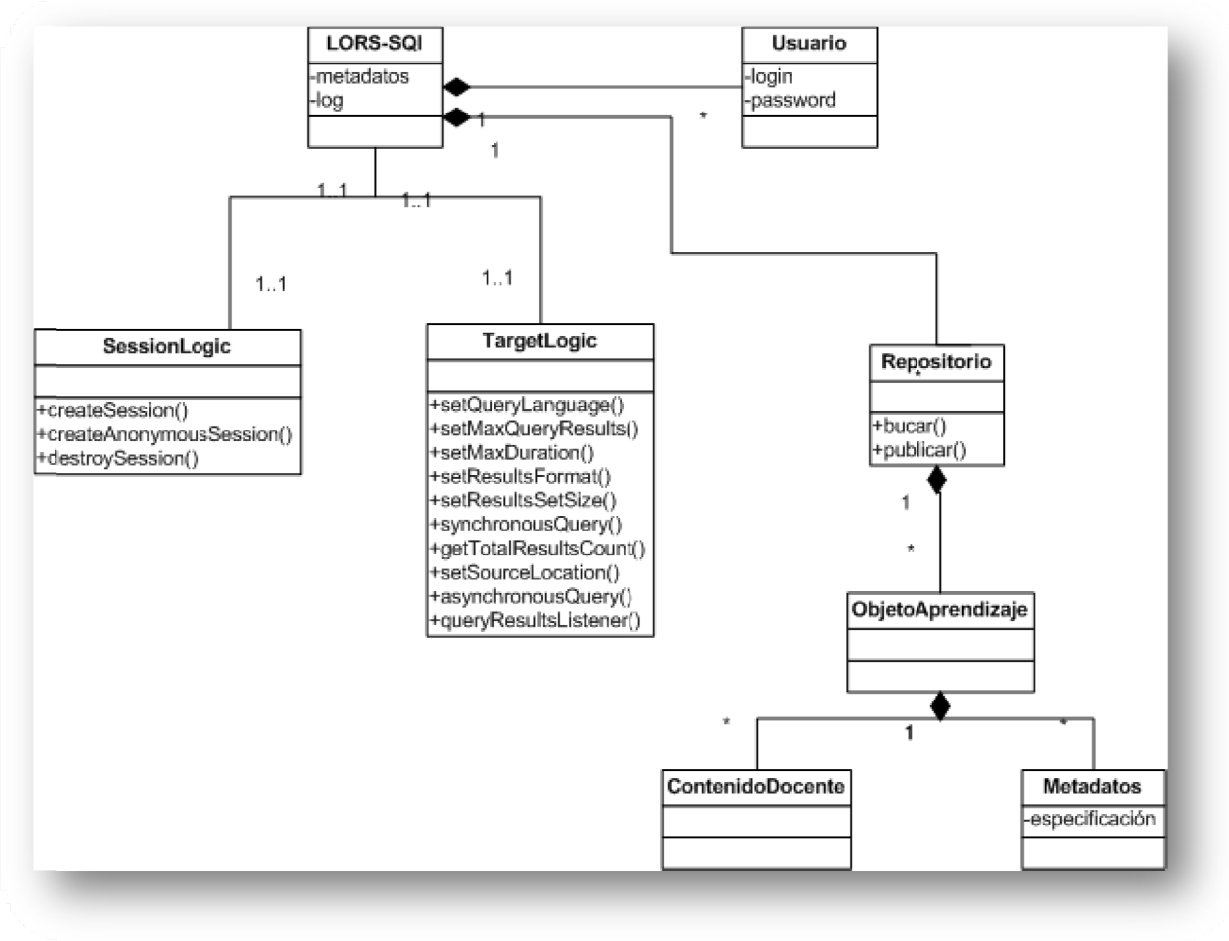


Figura 5.6 Modelo del dominio del problema

Diagramas de interacción

Para completar el estudio de la funcionalidad que ha de ser soportada por la arquitectura propuesta, se presentan a continuación los diagramas de interacción (a través de diagramas de secuencia) que se corresponden con las acciones más importantes que se pueden realizar con LORS-SQI (búsqueda federada y publicación de contenidos).

En la Figura 5.7 se muestra el diagrama de secuencia para la búsqueda de contenidos. En este caso el usuario lo primero que realiza es la autenticación en LORS-SQI (se supone que es un usuario registrado, o de lo contrario podrá establecer una sesión anónima) y, a continuación, le manda a LORS-SQI un mensaje para que active la operación de búsqueda (en función del tipo que sea) con la información sobre los metadatos que le interesan y la especificación que utiliza. LORS-SQI redirige la búsqueda hacia los repositorios distribuidos que le devolverán los objetos de aprendizaje coincidentes con los metadatos seleccionados. Una vez que LORS-SQI ha recibido los objetos de aprendizaje realizará su catalogación, para por último, devolverle el listado completo de objetos coincidentes con sus parámetros de búsqueda.

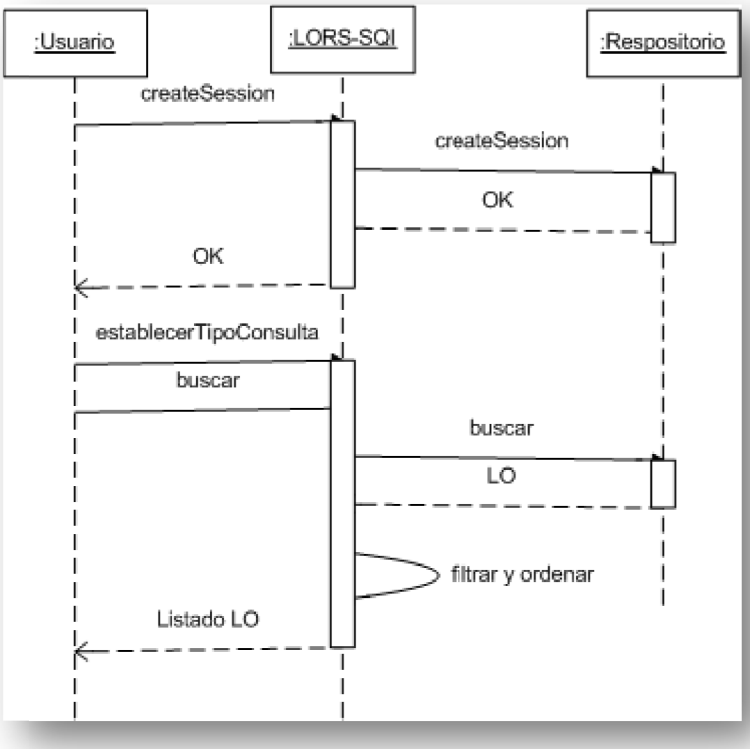


Figura 5.7 Diagrama de Secuencia: Búsqueda de contenidos

En la Figura 5.8 se muestra el diagrama de secuencia para la publicación de contenidos. Como se comentó en los casos de uso, la publicación de contenidos conlleva el registro en LORS-SQI de un repositorio SQI para dar acceso al mismo. Lo primero que debe hacer el usuario es realizar la autenticación en LORS-SQI (se supone que es un usuario registrado), y a continuación enviar a LORS-SQI un mensaje para que active la operación de publicación con la información sobre el repositorio que quiere publicar. LORS-SQI realizará una comprobación de localización positiva del repositorio y a continuación lo catalogará dentro del mismo, con todos sus datos, dando una respuesta positiva al usuario si todo ha sido correcto.

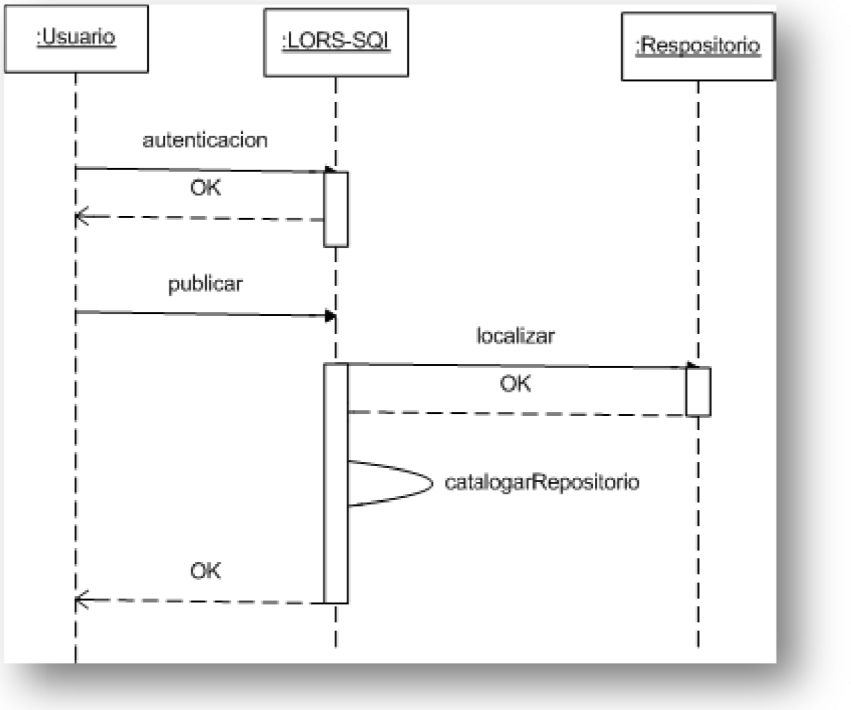


Figura 5.8 Diagrama de Secuencia: Publicación de contenidos

5.1.2. Modelo de servicios

En este apartado se describirá con detalle cada uno de los servicios que componen el sistema LORS-SQI. Para ello se utilizará las recomendaciones que IMS hace en su Abstract Framework [IMS, 2003b] y en su especificación sobre General Web Services [IMS, 2005] cuando describe un servicio.

De forma esquematizada realizaremos una tabla como la siguiente para describir cada servicio:

CAMPO	VALOR
Servicio	Nombre del servicio.
Descripción	Descripción informal del servicio.
Punto de Acceso al Servicio (PAS)	Signatura de las operaciones que realiza el servicio.
Operaciones	Descripción de cada una de las operaciones con los parámetros de entrada y los valores de retorno.
Dependencias	Dependencias de otros servicios.
Excepciones	Principales errores gestionados por el servicio.

A continuación se presentan los servicios más importantes agrupados por capas o niveles:

Capa 1 – Repositorios SQI

Servicio	**SessionLogic (Capa 1)**
Descripción	Servicio asociado a cada uno de los repositorios que están registrados en el sistema. Debe permitir gestionar la sesión con el mismo.
Punto de Acceso al Servicio (PAS)	String createSession(userID, password) String createAnonymousSession() void destroySession(sessionID)
Operaciones	► *createSession:* ✓ Parámetros de entrada: el identificador del usuario y su contraseña. ✓ Resultado: Un identificador de la sesión que el usuario utilizará en todas sus consultas. ► *createAnonymousSession:*

	✓ Parámetros de entrada: ninguno ✓ Resultado: Un identificador de la sesión que el usuario utilizará en todas sus consultas. ► *destroySession:* ✓ Parámetros de entrada: Identificador de la sesión del usuario. ✓ Resultado: ninguno.
Dependencias	Ninguna
Excepciones	Problemas relacionados con el procesado de la sesión, como por ejemplo los datos referentes al usuario.

Servicio	**TargetLogic (Capa 1)**
Descripción	Servicio asociado a cada uno de los repositorios que están registrados en el sistema. Debe permitir realizar una serie de acciones referentes a la consulta de información.
Punto de Acceso al Servicio (PAS)	void setQueryLanguage (sessionID, queryLanguageID) void setResultsFormat(sessionID, resultsFormat) void setMaxQueryResults(sessionID, maxQueryResults) void setMaxDuration(sessionID, maxDuration) void setResultsSetSize(sessionID, resultsSetSize) String sychronousQuery(sessionID, queryStatement, startResult) int getTotalResultsCount(sessionID, queryStatement) void asynchronousQuery(sessionID, queryStatement, queryID) void setSourceLocation(sessionID, sourceLocation) void queryResultsListener(sessionID, queryResults)
Operaciones	► setQueryLanguage*:* ✓ Parámetros de entrada: Identificador de la sesión del usuario y el identificador del lenguaje de consulta. ✓ Resultado: ninguno ► *setResultsFormat:* ✓ Parámetros de entrada: Identificador de la sesión del usuario y el formato de resultados esperado. ✓ Resultado: ninguno. ► *setMaxQueryResults:* ✓ Parámetros de entrada: Identificador de la sesión del usuario y el número máximo de resultados obtenidos por consulta. ✓ Resultado: ninguno. ► *setMaxDuration:* ✓ Parámetros de entrada: Identificador de la sesión del usuario y la duración máxima de la consulta. ✓ Resultado: ninguno ► *setResultsSetSize:* ✓ Parámetros de entrada: Identificador de la sesión del usuario y el tamaño máximo de resultados que se esperan obtener. ✓ Resultado: ninguno

	► *sychronousQuery:* ✓ Parámetros de entrada: Identificador de la sesión del usuario, la consulta y el primer registro que se desea obtener. ✓ Resultado: los datos coincidentes ► *getTotalResultsCount:* ✓ Parámetros de entrada: Identificador de la sesión del usuario y la consulta. ✓ Resultado: total de resultados coincidentes. ► *asynchronousQuery:* ✓ Parámetros de entrada: Identificador de la sesión del usuario, la consulta y el identificador de la consulta. ✓ Resultado: ninguno ► *setSourceLocation:* ✓ Parámetros de entrada: Identificador de la sesión del usuario y la localización de la fuente. ✓ Resultado: ninguno ► *queryResultsListener:* ✓ Parámetros de entrada: Identificador de la sesión del usuario y los resultados de la consulta. ✓ Resultado: ninguno
Dependencias	SessionLogic
Excepciones	Problemas relacionados con el procesado de las consultas

Servicio	**Descarga de contenidos (Capa 1)**
Descripción	Servicio encargado de descargar el LO seleccionado por el usuario.
Punto de Acceso al Servicio (PAS)	LO *descargarLO*(LO_seleccionado)
Operaciones	► *descargarLO:* ✓ Parámetros de entrada: El LO que el usuario ha seleccionado. ✓ Resultado: El LO
Dependencias	Servicio de cobro por derechos de autor.
Excepciones	Errores derivados del envío del LO a la máquina del cliente.

Capa 2 – Federación de servicios

Servicio	**Catalogación de repositorios (Capa 2)**
Descripción	Servicio encargado de mantener actualizada toda la información de los repositorios asociados al sistema (en base de datos), a través de los servicios utilizados para su encapsulación.
Punto de Acceso al Servicio (PAS)	bool *actualizarServicioRepositorio*(servicio, nuevo_estado) estado *comprobarEstado*(servicio)

	bool *registrarRepositorio*(descripción_repositorio)
Operaciones	► *Actualizar Servicio asociado a un Repositorio:* ✓ Parámetros de entrada: Servicio Web a actualizar, así como su nuevo estado (actualizado/desfasado). ✓ Resultado: Booleano con valores verdadero o falso dependiendo de si se logra actualizar el servicio Web. ► *Comprobar Estado:* ✓ Parámetros de entrada: Servicio Web a comprobar. ✓ Resultado: Estado del servicio Web (actualizado/desfasado). ► *Registrar repositorio:* ✓ Parámetros de entrada: La descripción del repositorio ✓ Resultado: Booleano con valores verdadero o falso dependiendo de si se logra registrar el repositorio.
Dependencias	Ninguna.
Excepciones	Errores de base de datos al acceder o modificar la información. Errores de acceso o búsqueda sobre el repositorio en cuestión.

Servicio	**Búsqueda federada (Capa 2)**
Descripción	Servicio encargado de realizar las llamadas a los distintos servicios de búsqueda y sesión SQI de los repositorios a los que se tiene acceso.
Punto de Acceso al Servicio (PAS)	String *búsquedaFederada*(metadatos, especificación, tipo_busqueda, porcentaje_coincidencia)
Operaciones	► *Búsqueda federada:* ✓ Parámetros de entrada: conjunto de metadatos a buscar, especificación utilizada por el cliente (para proceder a la posterior conversión de los LO), así como el tipo de búsqueda a realizar en los repositorios y el mínimo nivel de coincidencia que deberán cumplir los LO para ser considerados como válidos. ✓ Resultado: Todos los LO (de todos los repositorios distribuidos especificados en el sistema), cuyos metadatos coincidan con los especificados en la búsqueda.
Dependencias	Servicio de catalogación de repositorios. Servicio de catalogación de objetos de aprendizaje. Servicio de conversión de objetos de aprendizaje. Servicio de adaptación del fichero XEL.
Excepciones	Búsqueda sin resultados. Errores de acceso a algún repositorio.

Servicio	Catalogación de objetos de aprendizaje (Capa 2)
Descripción	Servicio encargado de filtrar y ordenar los objetos de aprendizaje recuperados de varios repositorios.
Punto de Acceso al Servicio (PAS)	lista_LO *filtrarLO*(lista_LO) lista_LO *ordenarLO*(lista_LO)
Operaciones	► *Filtrar objetos de aprendizaje:* ✓ Parámetros de entrada: Todos los LO recuperados por la búsqueda federada. ✓ Resultado: Todos los LO resultado de eliminar los duplicados. ► *Ordenar objetos de aprendizaje:* ✓ Parámetros de entrada: Todos los LO filtrados. ✓ Resultado: Todos los LO ordenados por índice de coincidencia.
Dependencias	Servicio de búsqueda federada.
Excepciones	Errores relacionados con la ordenación o eliminación de LO.

Servicio	Conversión de objetos de aprendizaje (Capa 2)
Descripción	Servicio encargado de la conversión de los objetos de aprendizaje al formato de metadatos esperado por el cliente.
Punto de Acceso al Servicio (PAS)	LO *convierteLO*(especificación, LO)
Operaciones	► *Convierte objeto de aprendizaje:* ✓ Parámetros de entrada: La especificación a la que queremos convertir el LO y el propio LO. ✓ Resultado: El LO convertido a la especificación.
Dependencias	Servicio de búsqueda federada.
Excepciones	Errores derivados de la conversión.

Capa 3 – Acceso y presentación

Servicio	Búsqueda de contenidos (Capa 3)
Descripción	Servicio encargado de desencadenar una búsqueda federada.
Punto de Acceso al Servicio (PAS)	String *búsquedaContenidos*(metadatos, especificación, tipo_busqueda, porcentaje_coincidencia)
Operaciones	► *Búsqueda de contenidos:* ✓ Parámetros de entrada: conjunto de metadatos a buscar, especificación utilizada por el cliente (para proceder a la posterior conversión de los LO), así como el tipo de búsqueda a realizar en los repositorios y el mínimo nivel de coincidencia que

	deberán cumplir los LO para ser considerados como válidos. ✓ Resultado: Todos los LO catalogados y convertidos, cuyos metadatos coincidan con los especificados en la búsqueda.
Dependencias	Servicio de búsqueda federada.
Excepciones	Búsqueda sin resultados. Errores de acceso a algún repositorio.

Servicio	**Recuperación de metadatos comunes (Capa 3)**
Descripción	Servicio encargado de leer de un fichero XSD, el conjunto de campos educativos con los que desea trabajar el usuario, así como las restricciones asociadas a dichos campos en caso de encontrarse.
Punto de Acceso al Servicio (PAS)	lista_metadatos *recuperarMetadatos*(fichero_XSD)
Operaciones	► *Recuperar metadatos:* ✓ Parámetros de entrada: El fichero XSD. ✓ Resultado: Lista con los metadatos.
Dependencias	Ninguna.
Excepciones	No se encuentra el fichero XSD o está mal formado.

Servicio	**Registro de repositorios (Capa 3)**
Descripción	Servicio encargado de registrar cada uno de los repositorios del sistema.
Punto de Acceso al Servicio (PAS)	bool *registrarRepositorio*(descripción_repositorio)
Operaciones	► *Registrar repositorio:* ✓ Parámetros de entrada: La descripción del repositorio. ✓ Resultado: Booleano con valores verdadero o falso dependiendo de si se logra registrar el repositorio.
Dependencias	Servicio de catalogación de repositorios.
Excepciones	Errores de base de datos al dar de alta el repositorio.

Servicio	**Registro de usuarios (Capa 3)**
Descripción	Servicio encargado de registrar a cada uno de los usuarios del sistema y establecer sus privilegios.
Punto de Acceso al Servicio (PAS)	bool *registrarUsuario*(nombre, contraseña, privilegios)
Operaciones	► *Registrar usuario:* ✓ Parámetros de entrada: Nombre de usuario,

	contraseña y privilegios dentro del sistema ✓ Resultado: Booleano con valores verdadero o falso dependiendo de si se logra registrar al usuario.
Dependencias	Ninguna.
Excepciones	Errores relacionados con el registro de usuarios.

Servicio	**Autenticación de usuarios (Capa 3)**
Descripción	Servicio encargado de autentificar a los usuarios cuando quieren utilizar el sistema y establecer sus privilegios.
Punto de Acceso al Servicio (PAS)	bool *autenticarUsuario*(nombre, contraseña)
Operaciones	► *Autenticar usuario:* ✓ Parámetros de entrada: Nombre de usuario y contraseña con las que se registró en el sistema ✓ Resultado: Booleano con valores verdadero o falso dependiendo de si se logra autenticar al usuario.
Dependencias	Ninguna.
Excepciones	Errores relacionados con la autenticación de usuarios.

Servicio	**Cobro por derechos de autor de los contenidos (Capa 3)**
Descripción	Servicio encargado de cobrar a los usuarios cuando quieren descargar contenidos sujetos a derechos de autor que tengan que ser pagados.
Punto de Acceso al Servicio (PAS)	Boolean *cobrarDerechos*(coste_derechos)
Operaciones	► *Cobrar derechos:* ✓ Parámetros de entrada: Coste de los derechos a cobrar. ✓ Resultado: Booleano con valores verdadero o falso dependiendo de si se logra cobrar los derechos de autor.
Dependencias	Sistemas de pago.
Excepciones	No se puede cobrar.

Servicio	**Registro de la trazabilidad de las acciones que se realizan en el sistema (Capa 3)**
Descripción	Servicio encargado de registrar todas las acciones que se realizan en el sistema y quedarán reflejadas en un fichero de *log*.
Punto de Acceso al Servicio (PAS)	*registrarAcción*(fecha, descripción_acción) fichero_log *examinarLog*()
Operaciones	► *Registrar acción:*

	✓ Parámetros de entrada: Fecha y descripción de la acción a registrar en el fichero de log. ✓ Resultados: Ninguno. ► *Examinar log:* ✓ Resultado: Fichero log con todas las acciones registradas.
Dependencias	Ninguna.
Excepciones	Errores de escritura/acceso en el fichero *log*.

A continuación se presenta en la Tabla 5.2, un resumen con los servicios presentados y su distribución por capas o niveles.

CAPA	SERVICIOS	NOMBRE EN IMPLEMENTACIÓN
1 - Repositorios SQI	Servicio de gestión de la sesión Servicio de consultas Servicio de descarga de contenidos	*SessionLogic* *TargetLogic* *DescargaContenidos*
2 - Federación de servicios	Servicio de catalogación de repositorios Servicio de búsqueda federada Servicio de catalogación de objetos de aprendizaje Servicio de conversión de objetos de aprendizaje Servicio de gestión de la sesión Servicio de consultas Servicio de descarga de contenidos	*CatalogaRepositorio* *BusquedaFederada* *CatalogacionLO* *ConversionLO* *SessionLogic* *TargetLogic* *DescargaContenidos*
3 - Acceso y presentación	Servicio de recuperación de metadatos comunes. Servicio de búsqueda de contenidos. Servicio de descarga de contenidos. Servicio de registro de repositorio. Servicio de registro de la trazabilidad de las acciones que se realizan en LORS-SQI	*ProcesarFicheroXSD* *BuscarContenidos* *DescargaContenidos* *CatalogaRepositorio* *UtilidadesFichero*

Tabla 5.2 Resumen de servicios generados por capas

5.1.3. Modelo de orquestación de servicios

Para una mayor comprensión de la implementación realizada de la arquitectura, se va a explicar el modelo BPEL elaborado como diseño de la orquestación de servicios, que se ha realizado con la herramienta Active BPEL [ActiveBPEL, 2009]. En la Figura 5.9 se muestra parte de este modelo, la correspondiente a la búsqueda federada en repositorios, ya que es la parte más importante de la arquitectura.

Como se puede ver, la orquestación de servicios comienza a través del servicio Web ***ProcesarFicheroXSD***, encargado de procesar el fichero XSD de campos educativos, y obtener el conjunto de campos educativos, así como las distintas restricciones impuestas a los mismos, para que el sistema pueda generar la interface de entrada de información, totalmente adaptada a dichos campos en cada momento. Cuando el usuario rellene la información, se comprobará la validez de la misma y en caso de ser correcta, se contactará con el servicio Web de gestión de la sesión (***SessionLogic***). Este servicio Web establecerá la misma sesión con todos los repositorios asociados al sistema de búsqueda, es decir, si el usuario crea una sesión anónima, esta se creará de igual forma en todos los repositorios asociados al sistema LORS-SQI. A partir de aquí comenzaría el sistema de búsqueda federada propiamente dicho, gracias a la funcionalidad ofrecida por el servicio ***BusquedaFederada***.

El sistema entraría en una estructura repetitiva, la cual, incluye contactar con el servicio Web de gestión de la sesión (para establecer una sesión con todos y cada uno de los repositorios distribuidos asociados al sistema), para posteriormente enviar una consulta SQI sobre el servicio Web de consulta del repositorio (***TargetLogic***). De él obtendrá un listado con todos los objetos de aprendizaje (su metainformación) que han resultado coincidentes con los parámetros marcados por el usuario. Cuando este proceso iterativo concluya se retornarán al servicio ***BusquedaFederada*** todos listado de datos de cada repositorio o un error si no se han obtenido datos.

Si alguno de estos repositorios produjera algún error, eso no supondría, evidentemente, la cancelación de la operación, sino que simplemente se le notificará al usuario de dicho proceso, continuando con la búsqueda si aún quedaran más repositorios por analizar. Para la invocación de los distintos servicios de búsqueda el sistema creará un hilo que se encargará de su procesamiento, siendo este proceso totalmente concurrente.

Cuando el proceso de búsqueda haya finalizado, será momento de catalogar toda esa información. El encargado de realizar el filtrado de los objetos de aprendizaje, eliminando todos aquellos repetidos, así como la ordenación de los mismos, para que de esta manera se le muestre una lista al usuario ordenada por nivel de coincidencia en sentido descendente, será el servicio de filtrado denominado ***CatalogacionLO***. Una vez que estos datos estén filtrados el servicio de devolución de datos se los mostrará al usuario.

Posteriormente a esta acción (ya no mostrado en el modelo), el usuario seleccionará de la lista mostrada un objeto de aprendizaje, invocando con ello al servicio Web de descarga asociado al repositorio que lo contenía (***DescargaContenidos***), el cual se encargará de enviarlo hasta su máquina local. Dicho fichero contendrá además de la información docente del mismo, la metainformación asignada por el autor. Esto constituye una de las principales diferencias con respecto a los principales repositorios analizados en el capitulo cuatro, ya que a través de este prototipo podremos obtener el objeto de aprendizaje completo.

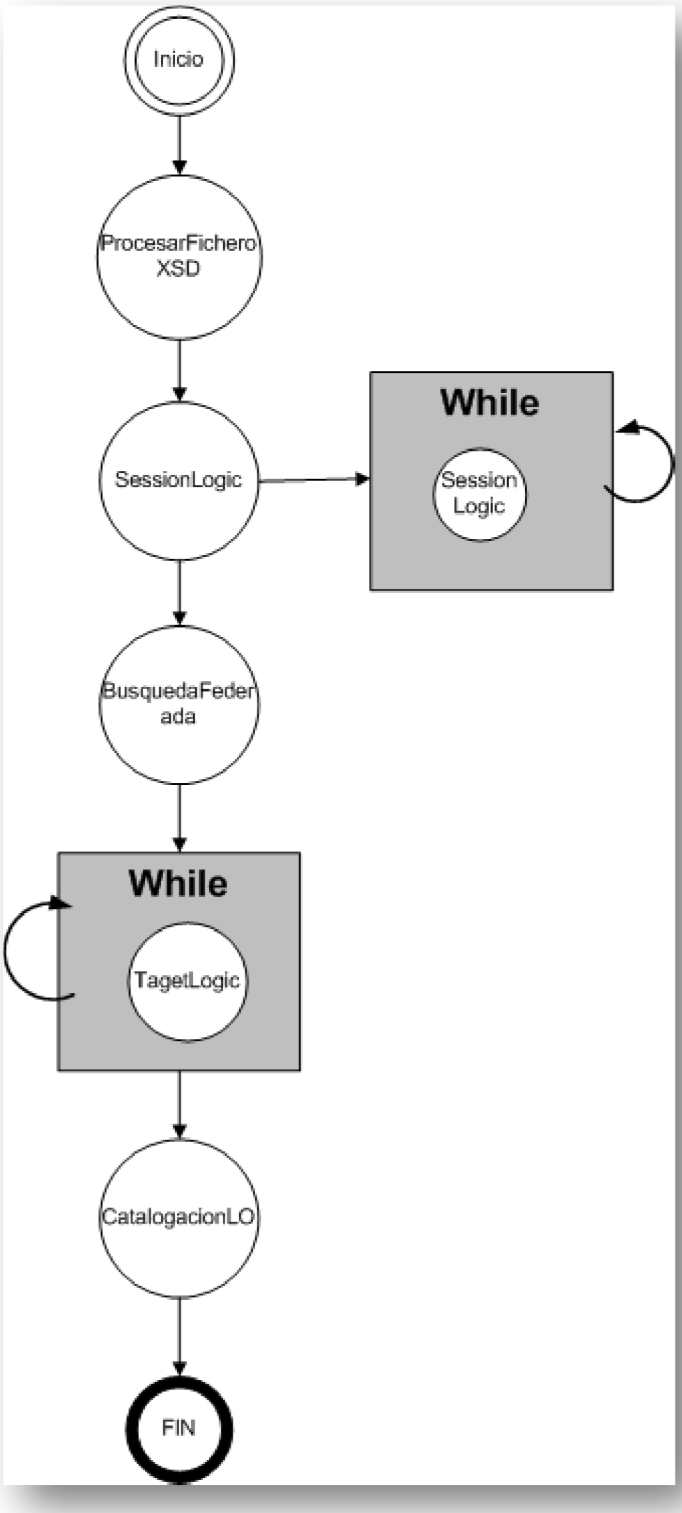


Figura 5.9 Modelo BPEL del servicio de búsqueda federada

Clases que implementan los servicios

En cuanto a la parte de las clases que componen el servicio de búsqueda federada (el más importante del sistema), se comentarán las principales clases que aportan la funcionalidad SQI, dividida en dos clases principalmente: SessionLogic y TargetLogic.

La lógica de implementación de los métodos de SQI tanto de Gestión de Sesión (SessionLogic) como de Consultas (TargetLogic) en cuanto a la distribución de dichos métodos sobre los repositorios registrados en el sistema, es la misma dependiendo, como se comprobará, si el método recibe o devuelve un identificador de sesión SQI.

La Clase "Session.Logic" se encarga de la parte de Gestión de Sesión SQI. Como se puede observar en la Figura 5.10, cuenta con los métodos de Gestión de Sesión SQI correspondientes.

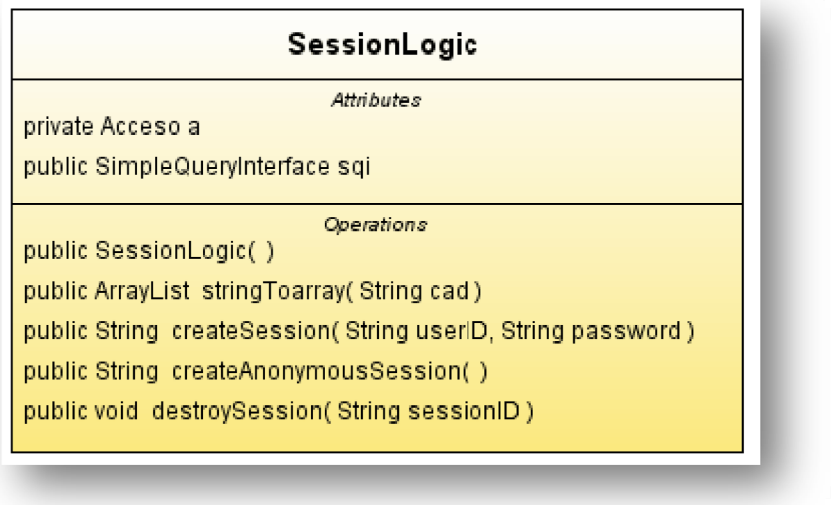


Figura 5.10 Diagrama de clases de la clase SessionLogic

Mediante la Clase "TargetLogic" se implementa la parte de Consultas SQI de manera distribuida sobre los repositorios presentes en el sistema de búsqueda federada. Como se pude observar en la Figura 5.11 cuenta con los métodos de Consulta SQI correspondientes.

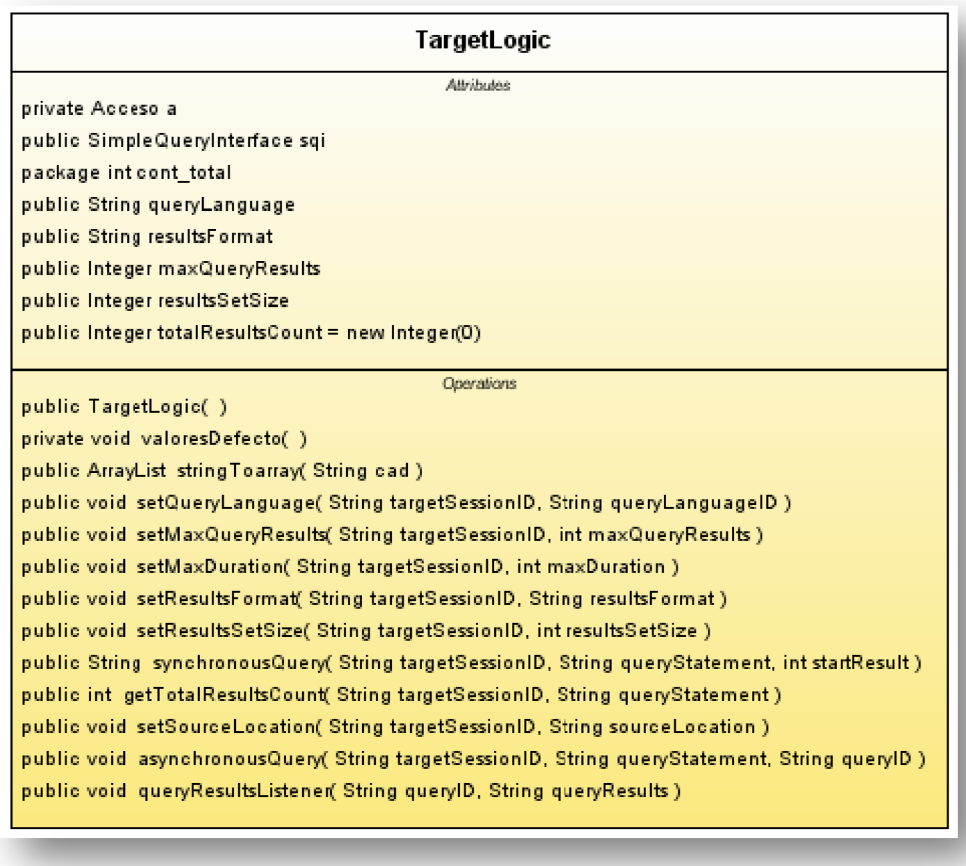
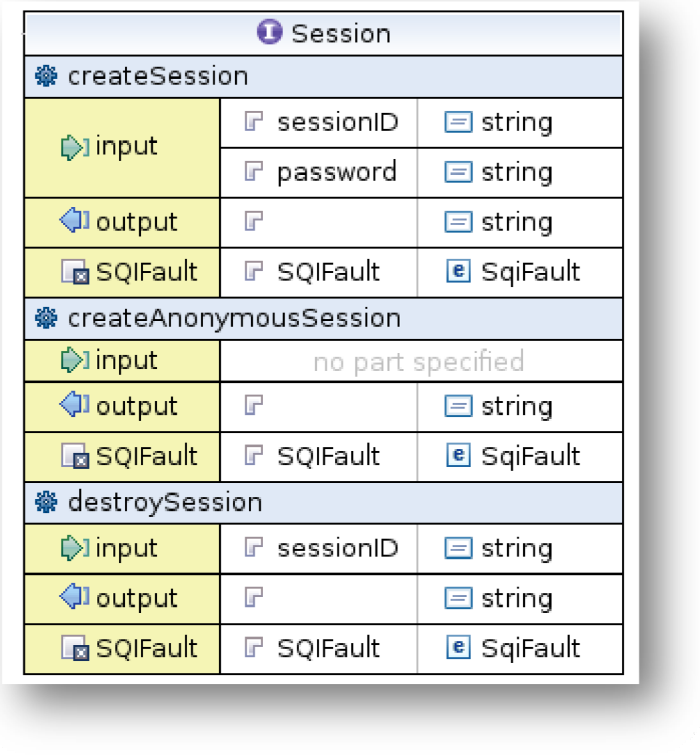


Figura 5.11 Diagrama de clases de la clase TargetLogic

Interfaz WSDL de los servicios

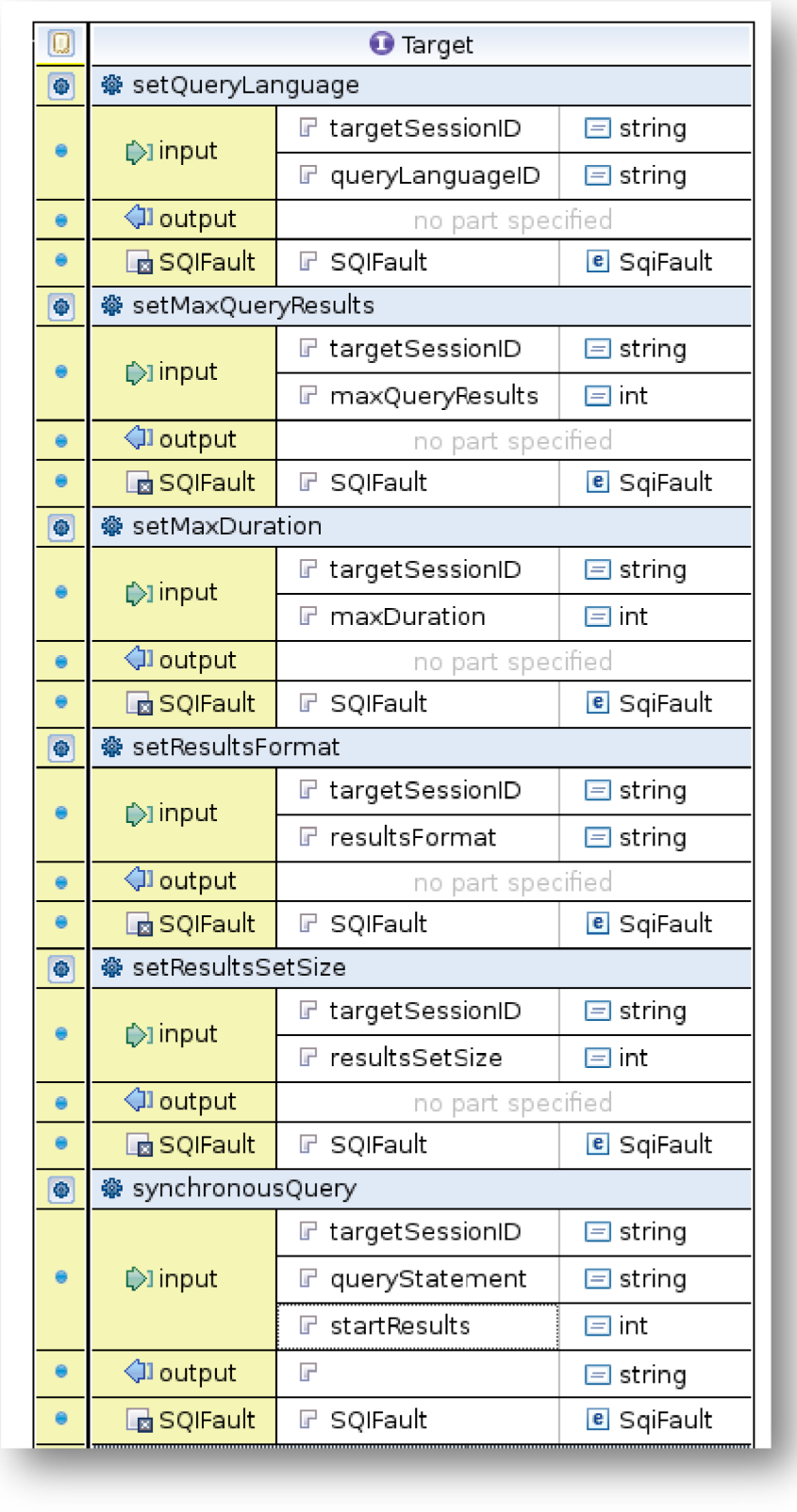
A continuación se muestran las interfaces WSDL de los dos servicios anteriormente comentados. Se puede ver en la Figura 5.12 la interfaz WSDL del servicio Web SessionLogin, con los tres métodos anteriormente comentados, y que servían para gestionar la sesión con el sistema de búsqueda.



Session		
createSession		
input	sessionID	string
	password	string
output		string
SQIFault	SQIFault	SqiFault
createAnonymousSession		
input	no part specified	
output		string
SQIFault	SQIFault	SqiFault
destroySession		
input	sessionID	string
output		string
SQIFault	SQIFault	SqiFault

Figura 5.12 Interfaz WSDL del servicio Web SessionLogic

Y por último se muestra en la Figura 5.13 y en la Figura 5.14 la interfaz WSDL del servicio Web de consulta, con todos los métodos SQI que permiten tanto las consultas síncronas como asíncronas.



Target		
setQueryLanguage		
input	targetSessionID	string
	queryLanguageID	string
output	no part specified	
SQIFault	SQIFault	SqiFault
setMaxQueryResults		
input	targetSessionID	string
	maxQueryResults	int
output	no part specified	
SQIFault	SQIFault	SqiFault
setMaxDuration		
input	targetSessionID	string
	maxDuration	int
output	no part specified	
SQIFault	SQIFault	SqiFault
setResultsFormat		
input	targetSessionID	string
	resultsFormat	string
output	no part specified	
SQIFault	SQIFault	SqiFault
setResultsSetSize		
input	targetSessionID	string
	resultsSetSize	int
output	no part specified	
SQIFault	SQIFault	SqiFault
synchronousQuery		
input	targetSessionID	string
	queryStatement	string
	startResults	int
output		string
SQIFault	SQIFault	SqiFault

Figura 5.13 Interfaz WSDL del servicio Web TargetLogic (parte 1)

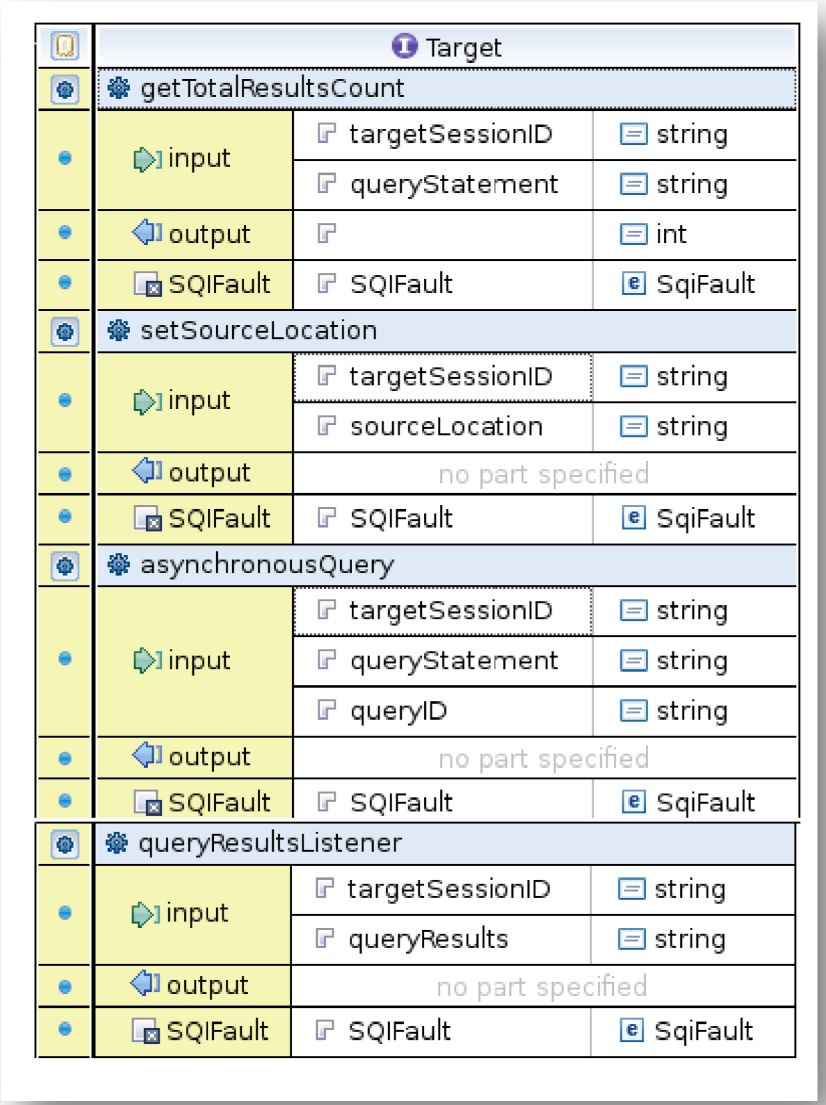


Figura 5.14 Interfaz WSDL del servicio Web TargetLogic (parte 2)

5.1.4. Modelo relacional

Como ya se comentó en el capítulo anterior, la arquitectura propuesta y con ello el prototipo presentado, se apoyan en un sistema gestor de base de datos para aportarle una mayor eficacia a las consultas realizadas, gracias a que se ahora se contará con la indexación de datos.

Siguiendo la arquitectura LORA-SQI, se podían identificar dos bases de datos dentro de la misma; una asociada a cada repositorio, donde se especificarán por ejemplo los campos educativos con los que trabaja, su diccionario de términos y las sesiones que establecen los usuarios con el mismo, y otra asociada al sistema de búsqueda federada, donde se hará referencia a todos los repositorios registrados, así como las sesiones con los mismos.

A continuación se presentan las tablas de las dos bases de datos con las que contará el prototipo LORS-SQI.

Base de Datos del Repositorio

La Base de Datos que va a tratar cada repositorio (a través del uso de la librería SQITL-API como ya se comentó anteriormente), es uno de los elementos más importantes puesto que en ella va a residir la metainformación de todos los objetos que componen un repositorio y sobre la cual se realizarán las consultas SQI transformadas a SQL. Así mismo el sistema también se apoyará en la base de datos para gestionar los identificadores de sesión necesarios y requeridos en todas las operaciones de SQI.

Todos los parámetros de configuración de la base de datos pueden ser modificados en el fichero de configuración “configuracion.properties”, como se puede ver en la Figura 5.2. A continuación se muestran las entidades con las que cuenta la base de datos de esta parte del sistema:

- Tabla “nombre_repositorio”: Esta tabla tendrá como nombre el valor indicado en el parámetro nombre_repositorio del fichero “configuracion.properties”. Esta tabla en principio se crea únicamente con el atributo “Id”, y al analizar el fichero “campos_educativos.xsd” y obtener sus campos, se van añadiendo

como atributos de la tabla. De esta forma una vez finalizado el análisis del fichero XSD se dispone de una tabla preparada para recibir la metainformación de cada uno de los objetos de aprendizaje que pertenecen al repositorio. Dicha información será insertada cada vez que se descomprima un objeto, se analice su fichero de metainformación y se extraigan los datos.

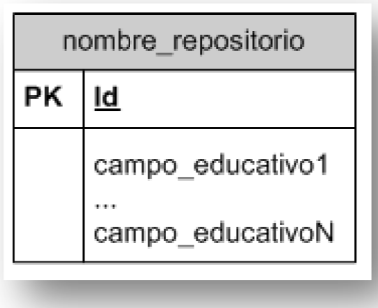


Figura 5.15 Tabla nombre_repositorio

- Tabla “diccionario_nombre_repositorio”: Esta tabla tomará el nombre del repositorio indicado en el fichero de configuración precedido de “diccionario_”. El objetivo de esta tabla es mantener una relación entre el nombre de un campo educativo, su contenido y el número de veces que esta pareja de valores se repite en la metainformación del conjunto de todos los objetos de aprendizaje incluidos en el repositorio. Esta relación se utilizará para dar prioridad en las consultas, en cuanto a que si un término de consulta aparece como valor en varios campos educativos se dará prioridad en la búsqueda al campo educativo que mayor número de apariciones del término tenga. En cuanto a los atributos de la tabla:
    - Id: Campo clave de la Tabla.
    - Termino: valor de campo educativo en la metainformación de un objeto de aprendizaje.
    - Atributo: nombre del campo educativo.
    - Numero: cantidad de veces que la pareja Termino-Atributo se repite considerando todos los ficheros de metainformación de los objetos de aprendizaje pertenecientes al repositorio.

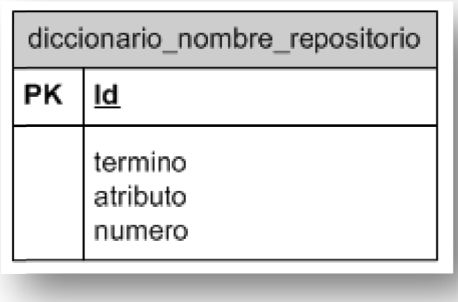


Figura 5.16 Tabla diccionario_nombre_repositorio

- Tabla "sesiones_nombre_repositorio": Como las anteriores tablas, esta también contendrá en su nombre de tabla el nombre del repositorio fijado en el fichero de configuración pero esta vez precedido de "sesiones_". Esta tabla es la encargada de mantener las sesiones creadas y destruidas por medio de la parte de Gestión de Sesión de SQI. La misma tabla es utilizada tanto para sesiones con credenciales como anónimas, puesto que en el caso de ser una sesión anónima se inserta un registro con user=anony y password=anony. En cuanto a los atributos de la tabla:
  - Id: Campo clave de la Tabla.
  - User: Nombre de usuario en el caso de sesión con credenciales, "anony" en el caso de sesión anónima.
  - Password: Contraseña en el caso de sesión con credenciales, "anony" en el caso de sesión anónima.
  - Status: Indica si una sesión está libre ("free") o por el contrario en uso ("busy").
  - Sessionid: Identificador de sesión que será la concatenación del nombre del repositorio y el valor del atributo Clave de la fila en cuestión

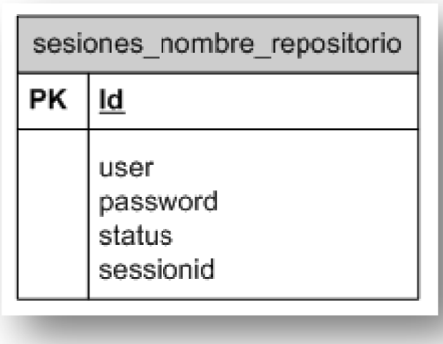


Figura 5.17 Tabla sesiones_nombre_repositorio

- Tabla “registro_nombre_repositorio”: Esta tabla mantendrá a los usuarios registrados en el sistema y sus respectivas contraseñas. Estos usuarios serán los que pueden iniciar sesión con credenciales, en cualquier otro caso se deberá optar por sesión anónima. NOTA: La librería no ofrece la funcionalidad de alta/baja de usuarios, únicamente comprueba que un usuario está registrado y que su contraseña es válida. En cuanto a los atributos de la tabla:
    - User: usuario registrado.
    - Password: contraseña de usuario registrado.

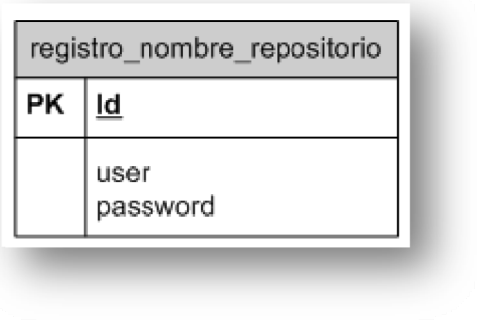


Figura 5.18 Tabla registro_nombre_repositorio

Base de Datos del Sistema de Búsqueda Federada

La base de datos con la que cuenta el sistema de búsqueda federada está formada por las siguientes entidades:

- Tabla “repositorios”: Esta tabla mantiene los repositorios registrados en el sistema de búsqueda federada así como la información relevante sobre los mismos. La información es insertada en esta tabla en la parte de Gestión de Repositorios con la que cuenta el sistema y es consultada en muchos de los procesos del mismo. En cuanto a los atributos de la tabla:
    - Id: Campo clave de la Tabla.
    - Nombre: Nombre del repositorio, elegido por el usuario del sistema.
    - Tipo: Este atributo indica la tecnología o estándar que utiliza el repositorio para mantener sus objetos de aprendizaje, de momento es simplemente de carácter informativo.
    - Session: Punto de acceso al servicio Web de Gestión de Sesión SQI del repositorio.
    - Target: Punto de acceso al servicio Web de Consultas SQI del repositorio.

- RutaFisica: En el caso de los repositorios desarrollados a través del uso de la librería SQITL-API, es decir, que dispongan del servicio Web destinado al envío de ficheros, este atributo contendrá la ruta física dónde se encuentran los recursos del repositorio. En caso contrario contendrá una URL relativa al repositorio. La inserción de valores en este campo no la realiza el usuario como en el caso de los anteriores, sino que se realiza automáticamente.

Figura 5.19 Tabla repositorios

- Tabla “sesiones_ repositorios”: Esta tabla es la encargada de mantener las sesiones creadas por medio de la parte de Gestión de Sesión de SQI. La misma tabla es utilizada tanto para sesiones con credenciales como anónimas utilizando la misma lógica que en la librería. En cuanto a los atributos de la tabla:
  - Id: Campo clave de la Tabla.
  - User: Nombre de usuario en el caso de sesión con credenciales, “anony” en el caso de sesión anónima.
  - Password: Contraseña en el caso de sesión con credenciales, “anony” en el caso de sesión anónima.
  - Status: Indica si una sesión está libre (“free”) o por el contrario en uso (“busy”).
  - Sessionid: Identificador de sesión en el sistema de búsqueda federada que será la concatenación de “repositorios” y el valor del atributo Clave de la fila en cuestión
  - Relations: Lista con las parejas formadas por los nombres de los repositorios registrados en el sistema y los identificadores de sesión SQI obtenidos en ellos y que se hayan obtenido mediante el identificador de sesión SQI del sistema de búsqueda correspondiente

al atributo Sessionid. Este atributo es utilizado para mantener la relación entre un identificador de sesión SQI creado en el sistema de búsqueda federada y los identificadores de sesión de los repositorios registrados en él.

Figura 5.20 Tabla sesiones_repositorios

- Tabla “registro_ repositorios”: Esta tabla mantendrá a los usuarios registrados en el sistema y sus respectivas contraseñas. Estos usuarios serán los que pueden iniciar sesión con credenciales, en cualquier otro caso se deberá optar por sesión anónima. NOTA: El sistema de búsqueda federada no ofrece la funcionalidad de alta/baja de usuarios, únicamente comprueba que un usuario está registrado y que su contraseña es válida.
    - User: usuario registrado.
    - Password: contraseña de usuario registrado.

Figura 5.21 Tabla registro_repositorios

5.1.5. Interfaces del sistema

Para validar la implementación de la arquitectura presentada en el capítulo anterior, se ha desarrollado una aplicación que hace uso de los servicios creados. Se trata de una aplicación Web que en su página principal ofrece al usuario la posibilidad de realizar una búsqueda federada en distintos repositorios distribuidos o la de registrar un nuevo repositorio en el sistema. A través de la Figura 5.22 se puede ver la nueva interfaz de entrada al sistema:

Figura 5.22 Interfaz de entrada en LORS-SQI

Dentro de la parte de gestión de repositorios será posible añadir nuevos repositorios para que las búsquedas tengan efecto también sobre los mismos. Esta es una de las nuevas y más importantes características con las que cuenta esta nueva versión, ya que al seguir todos los repositorios la misma interfaz de consulta, la interoperabilidad entre los mismos está garantizada. Se puede ver en la Figura 5.23 cómo sería el proceso de incorporación de nuevos repositorios.

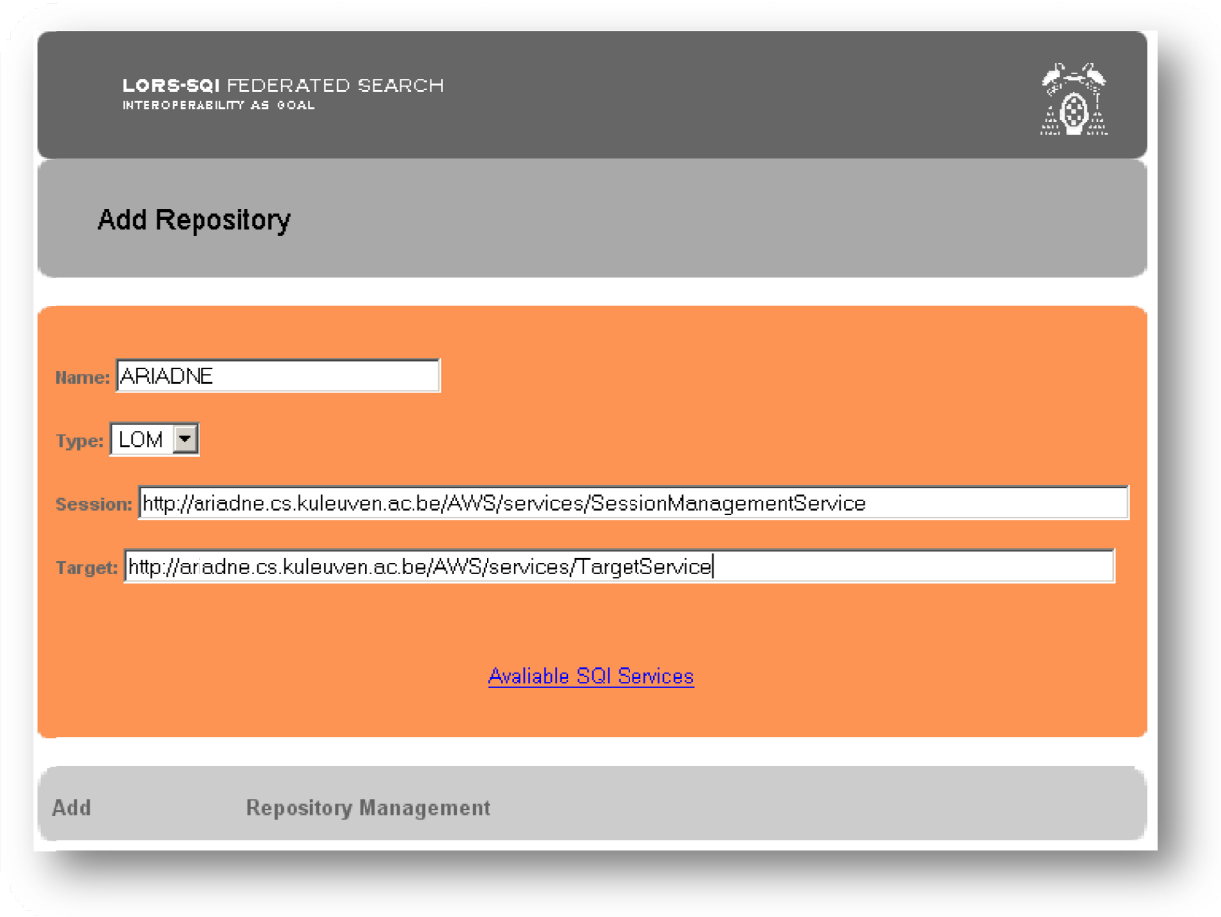


Figura 5.23 Añadiendo un nuevo repositorio en LORS-SQI

La otra opción que ofrecía la página Web inicial de la aplicación era la búsqueda. Las búsquedas que se realizan en este sistema, como ya se ha comentado, se realizan siguiendo la especificación SQI; por lo que antes de empezar a buscar es necesario crear una sesión ya sea anónima o con credenciales. El sistema no dispone de un mecanismo para gestionar usuarios, sino de gestión de sesiones SQI, por lo que si se desea añadir usuarios será modificando directamente la tabla correspondiente de la base de datos. A continuación se muestra en la Figura 5.24 cómo sería esta interfaz de entrada al sistema.

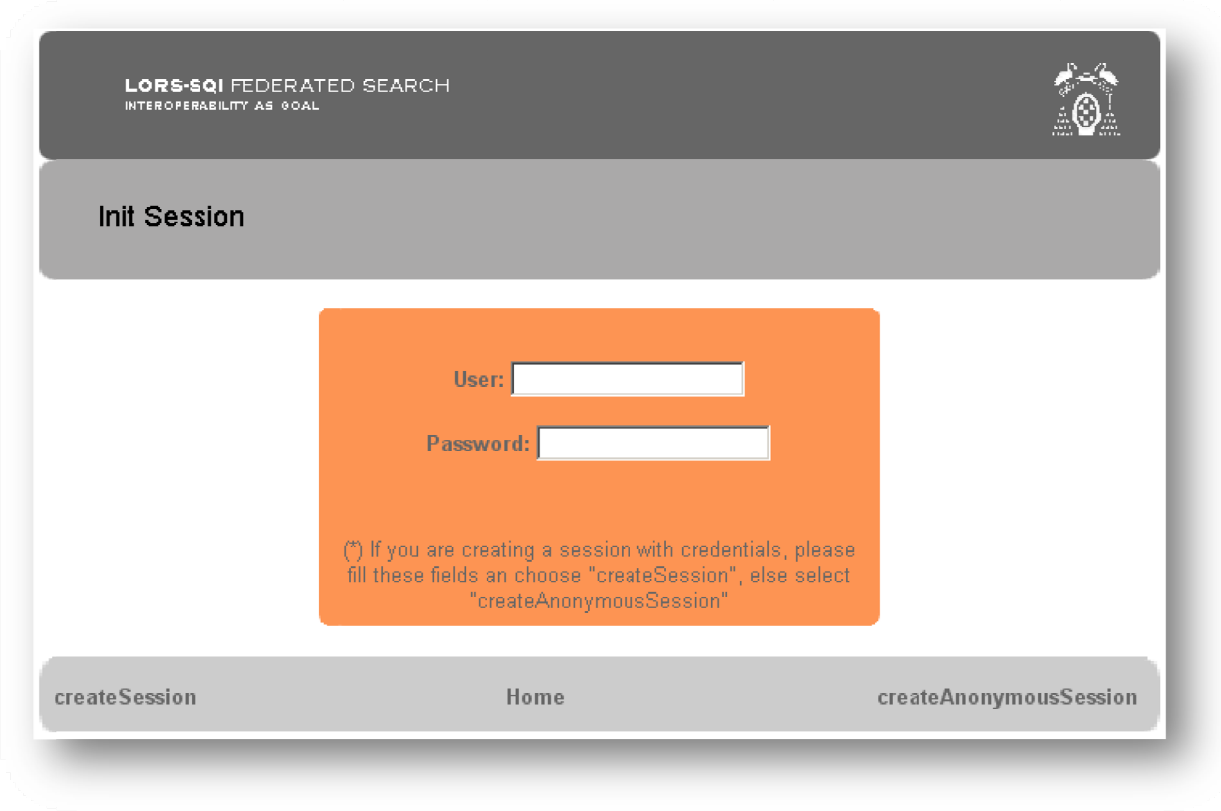


Figura 5.24 Inicio de sesión en LORS-SQI

Una vez que nos validemos contra el sistema, se seguirá el mismo proceso de validación frente a todos los repositorios registrados en LORS; de forma que si optamos por crear una sesión anónima, esta se creará de igual modo con todos los repositorios a los que LORS tiene acceso. Una vez que se ha creado la sesión, el sistema ofrece dos posibilidades de búsqueda: sencilla y avanzada.

La búsqueda avanzada o también llamada búsqueda exacta, ofrece la posibilidad de configurar varios parámetros SQI como el número máximo de resultados, el tamaño del conjunto de resultados o el resultado inicial. Al tratarse de una búsqueda exacta se realiza la búsqueda en base a campos educativos. Dichos campos están definidos en el fichero XSD (esta funcionalidad era el sistema de búsqueda con el que contaba la anterior versión de LORS). A continuación se puede ver en la Figura 5.25 cómo sería esa interfaz de consulta:

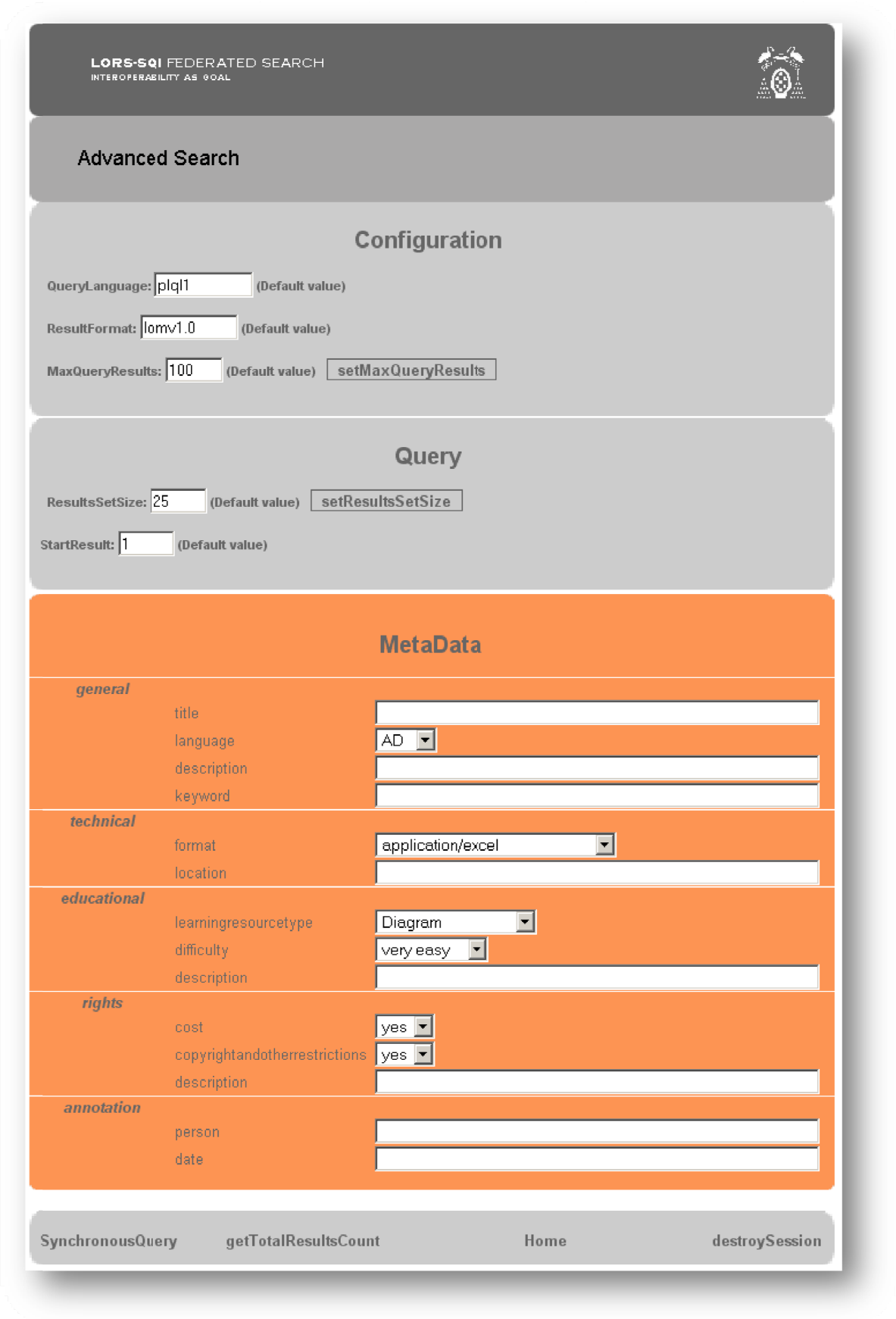


Figura 5.25 Interfaz de consulta exacta de LORS-SQI

En el caso de la búsqueda simple o búsqueda inexacta, las consultas se realizarán en función de cuántas veces aparezcan y con qué relevancia los términos introducidos en la declaración de la consulta en la metainformación de los objetos de aprendizaje. Al igual que en el caso anterior, también se pueden establecer los parámetros SQI: número máximo de resultados, el tamaño del conjunto de resultados y el resultado inicial. A continuación se muestra en la Figura 5.26 la interfaz de consulta resultante.

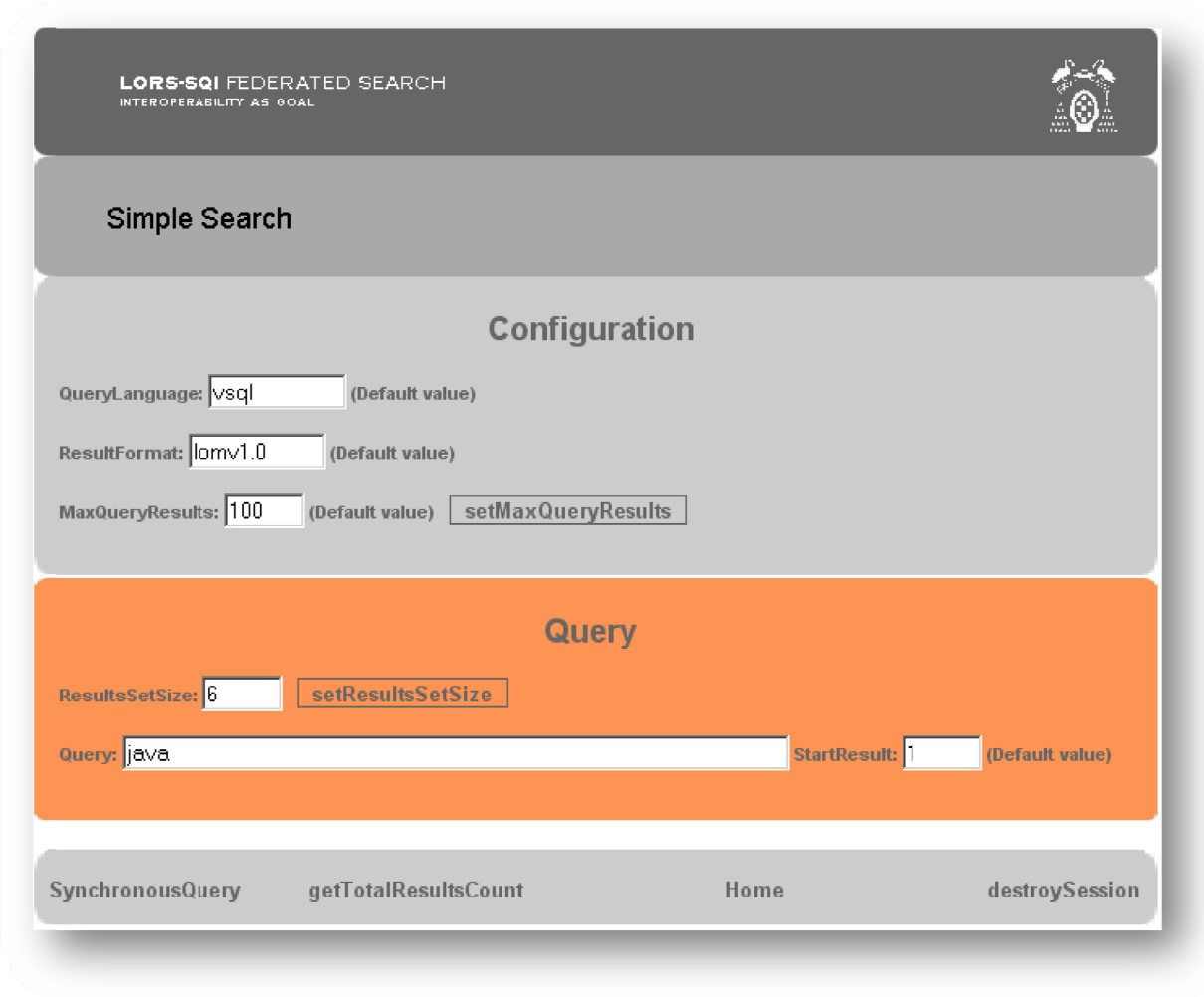


Figura 5.26 Interfaz de consulta inexacta de LORS-SQI

Tras la realización de la consulta, el sistema ofrecerá un listado ordenado con todos aquellos objetos de aprendizaje que hayan cumplido las especificaciones marcadas por el usuario. Se puede ver en la Figura 5.27 el listado ofrecido. El usuario podrá proceder a su descarga haciendo clic en el enlace. Una de las principales diferencias que se comentaron anteriormente y que ahora se recuerda, es que el repositorio asociado a LORS cuenta con un servicio extra que no pertenece a la especificación SQI de forma que se posibilita la

descarga de material educativo directamente, en lugar de enlazar con la página del autor como hacen otros repositorios como ARIADNE.

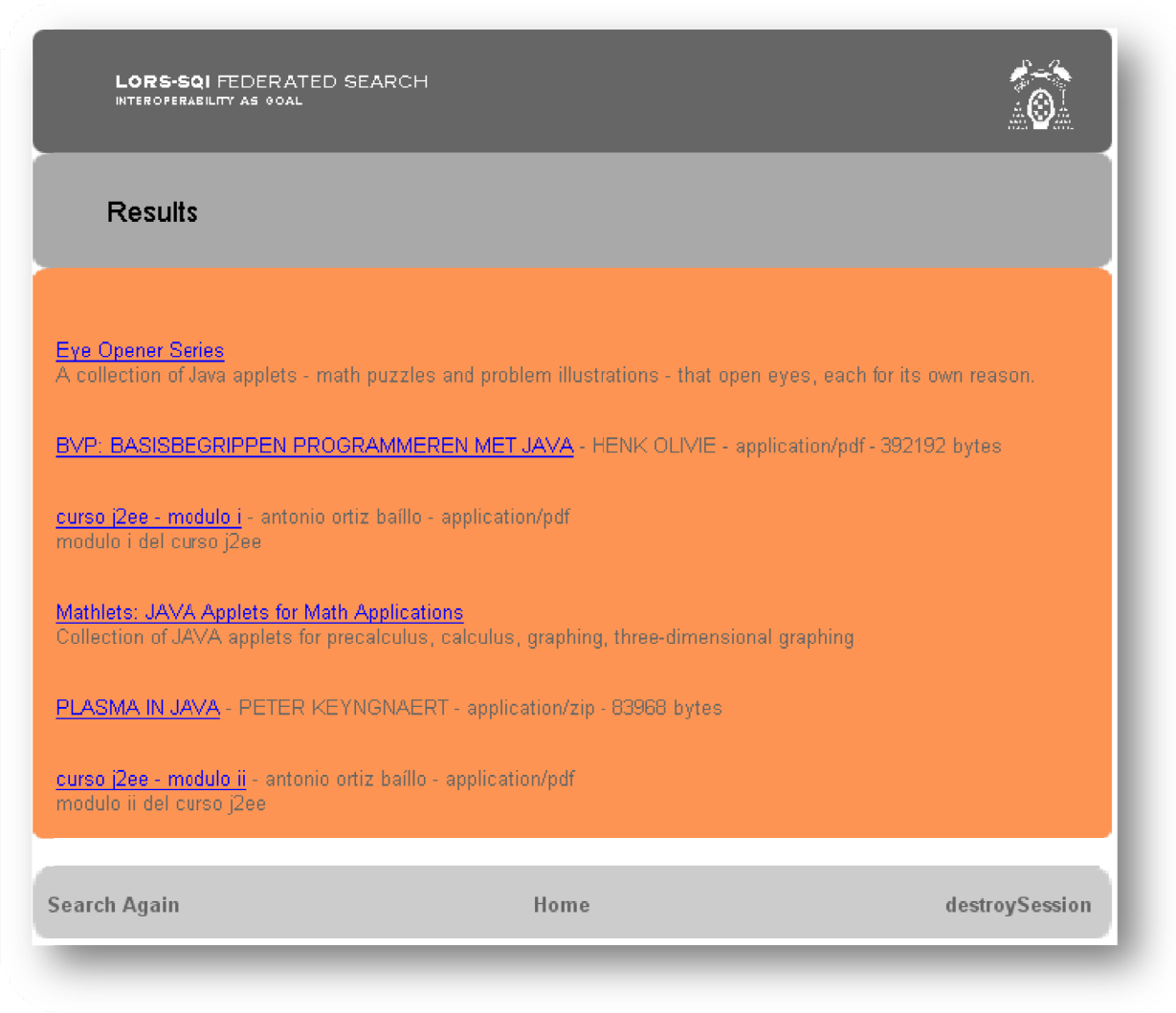


Figura 5.27 Interfaz de resultados obtenidos de LORS-SQI

5.2. PROTOTIPO AMPLIADO: LORS-SC

Durante este apartado se detallarán las características del sistema resultante de la arquitectura LORA-SC, ampliación de la arquitectura LORA-SQI, y por lo tanto del sistema anteriormente presentado y denominado LORS-SQI. En el capítulo anterior se definió la arquitectura LORA-SC, que como se indicó, suponía un cambio funcional importante. La arquitectura LORA-SC presentaba entre sus propuestas:

- Definir una nueva interfaz de consulta, la cual amplia la funcionalidad ofrecida por SQI y a la que se ha denominado SQI 2.0. Como principales novedades ofrece:
    - Un método de consulta que ofrece unos resultados más estructurados, lo cual facilitará enormemente su análisis y procesado, siguiendo la misma filosofía de ofrecer interoperabilidad e independencia entre sistemas.
    - Un método de obtención del material educativo. En este caso se trata de un método completamente nuevo y que finaliza el proceso de reutilización de objetos de aprendizaje, ya que de poco sirve una especificación de consulta si luego no existe un método que posibilite su descarga.
- Define una nueva forma de realizar consultas, basándose en el significado de los términos y no solamente en los términos propiamente dichos. Para ello, se definen una serie de conceptos de una o varias ontologías (asociada al objeto de aprendizaje a través de una serie de campos con los que ya contaba la especificación LOM [IMS, 2005]). Gracias a esta ontología se podrá conocer el significado de los términos bajo un dominio dado y así realizar búsquedas más efectivas.
- Posibilita la búsqueda y descarga de cursos completos de aprendizaje. Para realizar esta tarea, el sistema se basará nuevamente en ontologías para determinar los módulos que componen un curso completo y así realizar una búsqueda federada de todos ellos. Una vez obtenidos generará una serie de nuevos objetos de aprendizaje, creando además para cada uno de ellos su metainformación correspondiente.

A continuación se muestra en la Figura 5.28 un esquema de cómo quedaría el sistema LORS-SC:

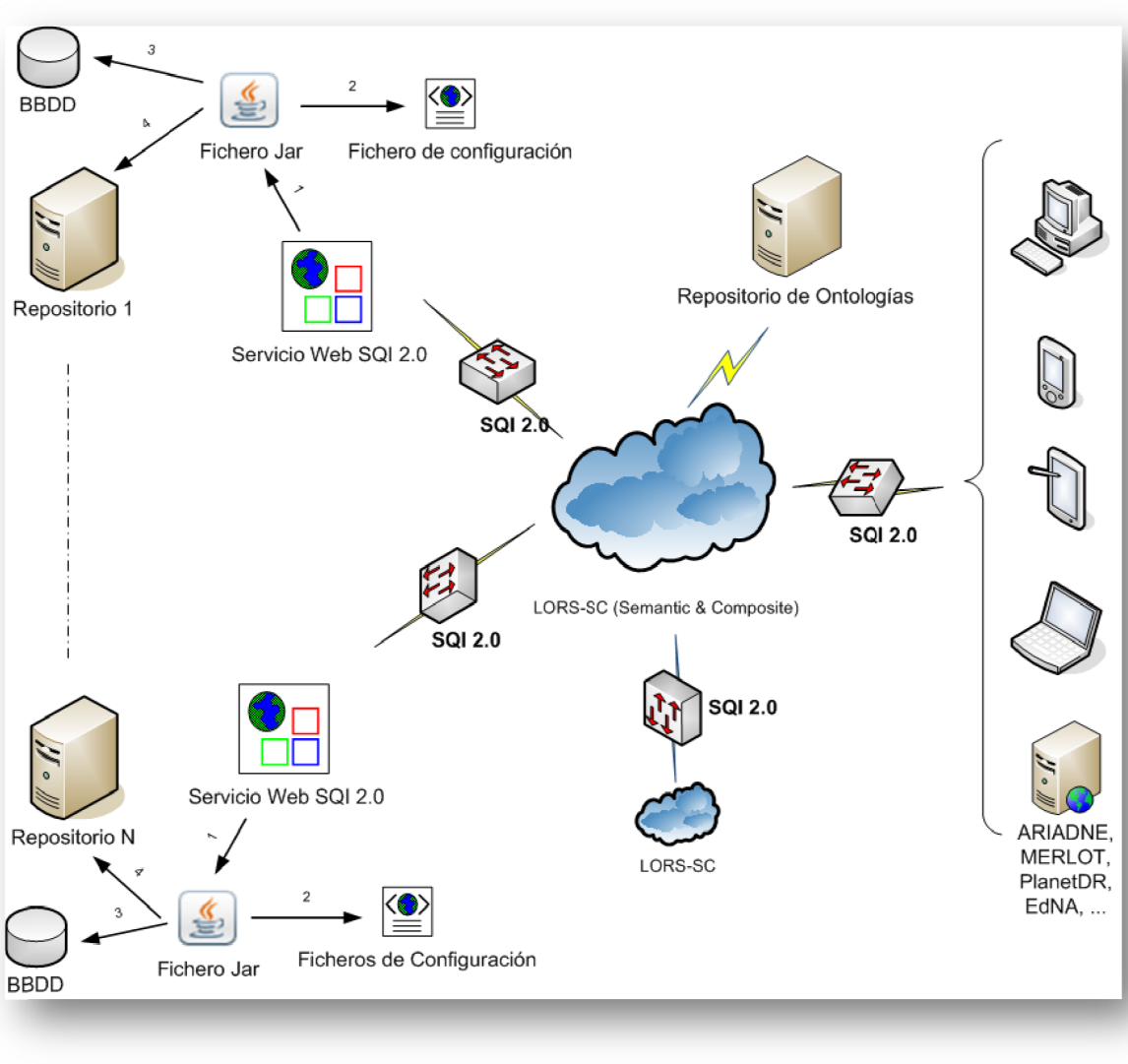


Figura 5.28 Elementos del sistema LORS-SC

Los cambios fundamentales se han producido en:

- La interfaz de consulta utilizada: a partir de ahora se utilizará SQI 2.0 en la comunicación entre el sistema de búsqueda y los repositorios. Este cambio afecta al sistema anterior en:
  - Para la devolución de los datos se utilizará ahora un método eficiente de almacenaje, a través del uso de estructuras eficientes de programación, como pueden ser listados de objetos, lo cual repercutirá favorablemente en el tratamiento y análisis de la información.
  - Para la obtención del objeto de aprendizaje se utilizará o bien la interfaz MTOM [2005] o bien arrays de bytes en caso de contar con

sistemas antiguos que no permitan el envío de datos adjuntos fuera del mensaje SOAP.

- Los sistemas de búsqueda de cada repositorio utilizarán técnicas semánticas en el acceso a los metadatos que describen a cada objeto de aprendizaje, como se comentó en el capítulo anterior. Además hay que tener en cuenta que ahora cada objeto de aprendizaje estará descrito además con el campo Classification de LOM, el cual reflejará una serie de conceptos extraídos de una o varias ontologías.
- El sistema LORS-SC contará con una nueva interfaz para permitir la búsqueda de cursos completos de aprendizaje, por lo que el sistema tendrá un nuevo servicio Web que dará tratamiento a esta funcionalidad.

A continuación se mostrarán, como en el caso anterior, el modelo funcional con los cambios establecidos en esta ampliación, el modelo de orquestación de servicios, el diseño de una ontología de ejemplo para la validación de las pruebas y algunas de las interfaces del mismo. En este caso no se muestra el modelo relacional, ya que la base de datos utilizada en este caso coincide plenamente con la del sistema anterior. Una de las características principales que tiene este sistema es que es adaptable completamente a cambios que pudieran surgir en cuanto a los metadatos utilizados, por eso, al añadir el campo Classification (para identificar los conceptos de la o las ontologías de cada objeto de aprendizaje), el sistema no ha tenido que sufrir cambios en este aspecto.

5.2.1. Modelo funcional

Como se ha indicado anteriormente, los cambios introducidos en esta ampliación del sistema son básicamente tres:

- Modificación de los métodos de consulta para hacerlos más eficientes: Funcionalmente no conlleva cambios, ya que los cambios se producen a nivel interno de los métodos desarrollados; por lo tanto, este cambio afectará al modelo de servicios principalmente.
- Añadir técnicas semánticas en las consultas: Funcionalmente tampoco conlleva cambios, ya que al igual que antes, se modificarán internamente los métodos de consulta.
- Posibilitar la descarga de cursos completos de aprendizaje: Funcionalmente esta tarea conlleva más cambios, ya que será necesario añadir nuevos servicios que posibiliten esta tarea.

Para describir estas nuevas funcionalidades, en primer lugar se realizará una presentación de los requisitos generales que debe satisfacer LORS-SC, después se detallará cada requisito en un caso de uso donde se explicará con más detalle los pasos a seguir para completar con éxito la funcionalidad requerida, a continuación se presentará el modelo del dominio del problema y por último los diagramas de interacción de la nueva funcionalidad presentada.

Especificación de requisitos generales

Al igual que se hizo para LORS-SQI, una de las tareas que primero se deben realizar a la hora de definir la arquitectura de un sistema, es establecer la especificación de sus requisitos. Mediante estos requisitos se podrán conocer los procesos que se llevan a cabo en el dominio del problema y establecer los objetivos que debe cumplir la arquitectura.

A continuación se muestra en la Tabla 5.3 un resumen con los nuevos requisitos que posee el sistema LORS-SC con respecto a LORS-SQI:

REQUISITOS	DESCRIPCIÓN
R1	Modificar los sistemas de búsqueda de SQI para que retornen información más estructurada.
R2	Añadir un método a la especificación SQI para que posibilite la descarga del material docente.
R3	Modificar los métodos de consulta para que manejen ontologías y de esta forma se afinen en mayor medida las consultas
R4	Posibilitar la búsqueda y descarga de cursos completos de aprendizaje

Tabla 5.3 Especificación de requisitos del sistema LORS-SQI

Actores

Antes de describir cada uno de estos nuevos requisitos en forma de casos de uso, es necesario detallar qué actores utilizarán el sistema. Al igual que ocurría con el sistema LORS-SQI, se partirá de la especificación IMS DRI (Digital Repository Interoperability) [IMS, 2003a] explicada en el capítulo 2. Esta especificación determinaba los roles de Bibliotecario, Contribuyente, Prestatarios, Usuarios Casuales, Administrador y Agentes Software.

Los tres primeros requisitos no afectan en nada a este apartado de actores, ya que no suponen una nueva funcionalidad, sino que se trata de modificaciones de funcionalidades ya existentes en el sistema anterior. El cuarto requisito sí que supone una nueva funcionalidad, en este caso una nueva tipología de búsqueda que será utilizada por los actores Bibliotecario, Prestatario, Agente Software y Usuario Casual.

Casos de uso

Para describir cada uno de los casos de uso correspondientes a los requisitos funcionales presentados anteriormente, se va a utilizar la misma plantilla que se utilizó para la descripción de los casos de uso del sistema LORS-SQI, que como se recuerda estaba basada en la utilizada por IMS en DRI [IMS, 2003a] para describir sus casos de uso.

En el tema de los casos de usos ocurre igual que con los actores, realmente de los cuatro requisitos que se enunciaban al principio del apartado, solamente el último es un requisito totalmente nuevo, por lo que, utilizando la plantilla anterior, se muestra a continuación la funcionalidad que permitirá una búsqueda y obtención de cursos completos de aprendizaje.

Funcionalidad	***R4: Posibilitar la búsqueda y descarga de cursos completos de aprendizaje***
Objetivo	Permitir que un usuario pueda realizar una consulta de un curso completo (en base a una o varias ontologías que determinarán los contenidos de dicho curso) y el sistema le presente un listado ordenado de cursos completos que cuenten con los respectivos módulos unitarios seleccionados por el usuario.
Actores	Prestatarios, usuarios casuales, agentes software y bibliotecario.
Precondiciones	Detallar la metainformación de los contenidos a buscar.
Pasos	1. Especificar el estándar utilizado por el cliente (SCORM, IMS, etc.). 2. Seleccionar los campos estándar a buscar (como idioma de los contenidos, derechos de autor, tipo de ficheros, dificultad, etc.) 3. Seleccionar los contenidos de dicho curso navegando por las distintas ontologías a las que se tiene acceso. 4. Realizar la búsqueda federada unitaria en todos los repositorios registrados en LORS-SC 5. Generar los distintos cursos realizando combinaciones de todos los recursos unitarios encontrados en los diferentes repositorios distribuidos. 6. Catalogar los contenidos resultado de la búsqueda. 7. Presentar los resultados de la búsqueda al solicitante.
Extensiones	Si se produce un error en el proceso de búsqueda se le comunicará al usuario. Si no se encuentra ningún contenido que coincida con los parámetros de búsqueda se le comunicará al usuario.

En la Figura 5.29 se representa el diagrama UML del caso de uso comentado anteriormente

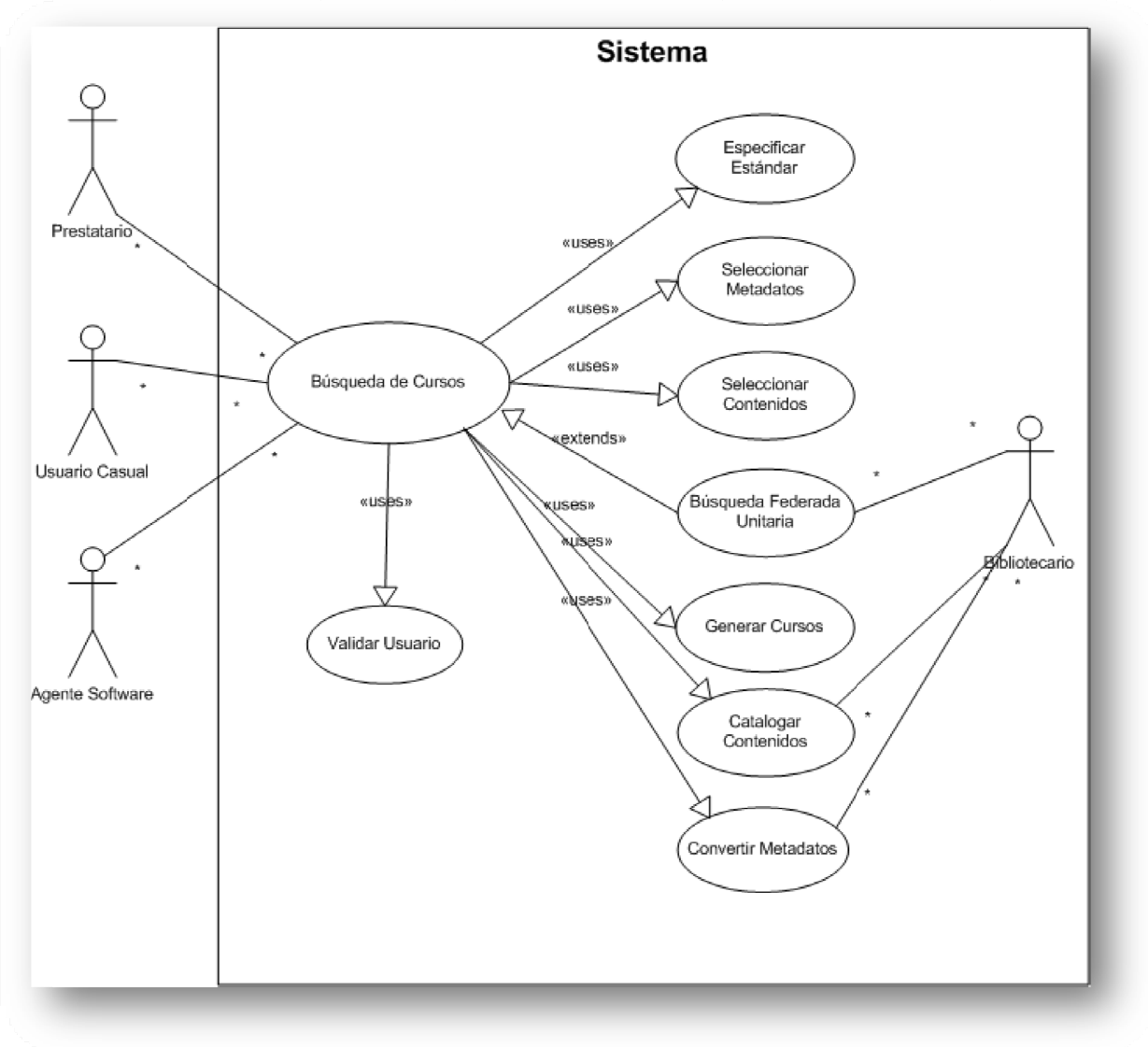


Figura 5.29 Caso de uso: Búsqueda de cursos

Modelo del Dominio

El modelo del dominio tampoco se verá modificado, ya que la nueva funcionalidad añadida es una tipología de búsqueda que se generará en base a varias búsquedas individuales de contenido docente, por lo que no afectará al modelo del dominio.

Diagramas de interacción

Para completar el estudio de la funcionalidad que ha de ser soportada por la arquitectura propuesta, se presenta a continuación el diagrama de interacción de esta nueva tipología de búsqueda.

En la Figura 5.30 se muestra el diagrama de secuencia para la búsqueda de cursos. En este caso el usuario lo primero que realiza es la autenticación en LORS-SC (se supone que es un usuario registrado, o de lo contrario podrá establecer una sesión anónima) y, a continuación, le manda a LORS-SC un mensaje para que active la operación de búsqueda con la información sobre los metadatos que le interesan (como idioma, derechos de autor, formato de los ficheros, etc.), así como la especificación que utiliza. Posteriormente seleccionará todos aquellos módulos individuales que le interesan, de una o varias ontologías a las que el sistema tendrá acceso (a través de su URL). LORS-SC redirige la búsqueda hacia los repositorios distribuidos que le devolverán los objetos de aprendizaje coincidentes con los metadatos seleccionados. Una vez que LORS-SC ha recibido los objetos de aprendizaje, generará combinaciones de cursos con todos estos objetos individuales, para posteriormente realizar su catalogación. Por último, le devolverá el listado completo de cursos coincidentes con sus parámetros de búsqueda formados por aproximaciones[3] a los contenidos individuales marcados por el usuario.

[3] Se ha detallado "aproximaciones" ya que en la mayor parte de las ocasiones será posible que se encuentre un objeto de aprendizaje superior o inferior en función de la jerarquía que determine la ontología.

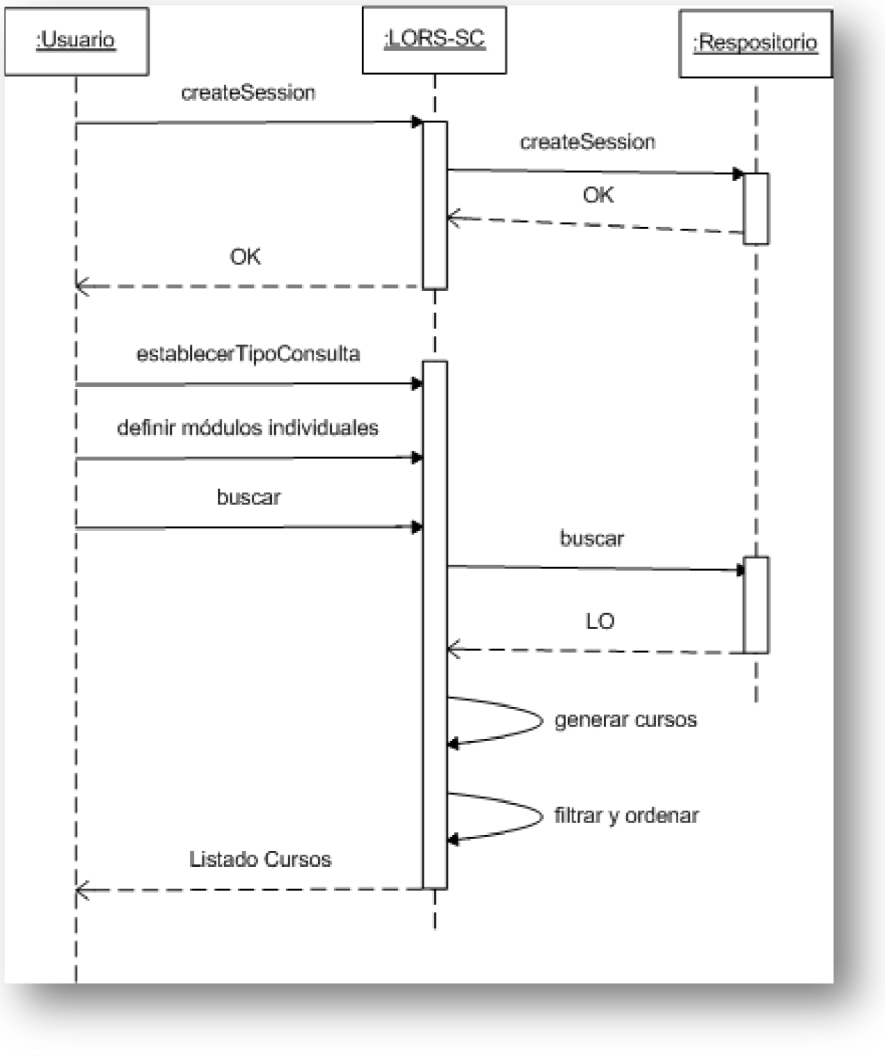


Figura 5.30 Diagrama de Secuencia: Búsqueda de cursos

5.2.2. Modelo de servicios

En este apartado se describirá con detalle cada uno de los servicios nuevos que se han incorporado al sistema LORS-SC (o que se han visto modificados). Para ello se utilizará, al igual que en el caso anterior, las recomendaciones que IMS hace en su Abstract Framework [IMS, 2003b] y en su especificación sobre General Web Services [IMS, 2005] cuando describe un servicio. La tabla esquemática de descripción de cada servicio será la misma que en el caso anterior.

A continuación se presentan los servicios más importantes agrupados por capas o niveles:

Capa 1 – Repositorios SQI

Servicio	**TargetLogic (Capa 1)**
Descripción	Servicio asociado a cada uno de los repositorios que están registrados en el sistema. Debe permitir realizar una serie de acciones referentes a la consulta de información. Se ven modificados los métodos de consulta (tanto síncrona como asíncrona) para que en lugar de manejar cadenas de caracteres como formato para devolver la información, utilicen formatos más estructurados. Además se incorpora un método que posibilita la descarga del contenido docente.
Punto de Acceso al Servicio (PAS)	Class sychronousQuery(sessionID, queryStatement, startResult) void queryResultsListener(sessionID, queryResults) byte[] downloadResource (sessionID, resourceID, downloadType)
Operaciones	► *sychronousQuery:* ✓ Parámetros de entrada: Identificador de la sesión del usuario, la consulta y el primer registro que se desea obtener. ✓ Resultado: los datos coincidentes en formato estructurado. ► *queryResultsListener:* ✓ Parámetros de entrada: Identificador de la sesión del usuario y los resultados de la consulta en formato estructurado. ✓ Resultado: ninguno ► *downloadResource:* ✓ Parámetros de entrada: Identificador de la sesión del usuario, identificador del recurso y tipología de descarga (sencilla – byte[] o avanzada – MTOM) ✓ Resultado: byte[] si se opta con la descarga

	sencilla o ineficiente.
Dependencias	SessionLogic
Excepciones	Problemas relacionados con el procesado de las consultas

Capa 2 – Federación de servicios

Servicio	**Búsqueda de cursos (Capa 2)**
Descripción	Servicio encargado de realizar las llamadas al servicio de búsqueda federada tantas veces como contenidos individuales desee el usuario para su curso
Punto de Acceso al Servicio (PAS)	Class *búsquedaCursos*(metadatos, especificación, contenidos individuales)
Operaciones	► *Búsqueda Cursos:* ✓ Parámetros de entrada: conjunto de metadatos a buscar, especificación utilizada por el cliente (para proceder a la posterior conversión de los LO de los cursos), así como el conjunto de contenidos individuales que desea el usuario en su curso. ✓ Resultado: Todos los cursos que el sistema ha conseguido generar en base a esos contenidos individuales.
Dependencias	Servicio de búsqueda federada.
Excepciones	Búsqueda sin resultados. Errores de acceso a algún repositorio.

Servicio	**Composición de cursos (Capa 2)**
Descripción	Servicio encargado de realizar las composiciones de los cursos en función de los objetos de aprendizaje individuales que se han obtenido.
Punto de Acceso al Servicio (PAS)	Class *composicionCursos*(metadatos, especificación, contenidos individuales, listadoObjetosIndividuales) Byte[] descargaCurso (idObjetosIndividuales, tipoDescarga)
Operaciones	► *Búsqueda Cursos:* ✓ Parámetros de entrada: conjunto de metadatos a buscar, especificación utilizada por el cliente (para proceder a la posterior conversión de los LO de los cursos), conjunto de contenidos individuales que desea el usuario en su curso y los objetos individuales que se han obtenido. ✓ Resultado: Todos los cursos que el sistema ha conseguido generar en base a esos contenidos individuales. ► *DescargaCurso:*

	✓ Parámetros de entrada: identificador de todos los objetos individuales que componían el curso y el tipo de descarga que se desea realizar, exactamente igual que la comentada para objetos individuales. ✓ Resultado: byte[] si se opta con la descarga sencilla o ineficiente.
Dependencias	Servicio de composición de cursos
Excepciones	Búsqueda sin resultados. Errores en la composición de los cursos. Errores de acceso a algún repositorio.

Servicio	**Catalogación de cursos (Capa 2)**
Descripción	Servicio encargado de filtrar y ordenar los cursos generados por el anterior servicio.
Punto de Acceso al Servicio (PAS)	Class *filtrarCurso*(lista_cursos) Class *ordenarCurso*(lista_cursos)
Operaciones	► *Filtrar cursos:* ✓ Parámetros de entrada: Todos los cursos generados por el sistema. ✓ Resultado: Todos los cursos resultado de eliminar los duplicados. ► *Ordenar cursos:* ✓ Parámetros de entrada: Todos los cursos filtrados. ✓ Resultado: Todos los cursos ordenados por índice de coincidencia.
Dependencias	Servicio de composición de cursos.
Excepciones	Errores relacionados con la ordenación o eliminación de cursos.

Capa 3 – Acceso y presentación

Servicio	**Búsqueda de cursos (Capa 3)**
Descripción	Servicio encargado de desencadenar una búsqueda de cursos.
Punto de Acceso al Servicio (PAS)	Class *búsquedaCursos*(metadatos, especificación, contenidos individuales)
Operaciones	► *Búsqueda Cursos:* ✓ Parámetros de entrada: conjunto de metadatos a buscar, especificación utilizada por el cliente (para proceder a la posterior conversión de los LO de los cursos), así como el conjunto de contenidos individuales que desea el usuario en su curso. ✓ Resultado: Todos los cursos que el sistema ha conseguido generar en base a esos contenidos

	individuales.
Dependencias	Servicio de búsqueda de cursos de la capa 2.
Excepciones	Búsqueda sin resultados. Errores de acceso a algún repositorio.

A continuación se presenta en la Tabla 5.4, un resumen con los servicios presentados y su distribución por capas o niveles.

CAPA	SERVICIOS	NOMBRE EN IMPLEMENTACIÓN
1 - Repositorios SQI	Servicio de consultas	*TargetLogic*
2 - Federación de servicios	Servicio de búsqueda de cursos Servicio de composición de cursos Servicio de catalogación de cursos Servicio de consultas	*BusquedaCursos* *ComposicionCursos* *CatalogaCursos* *TargetLogic*
3 - Acceso y presentación	Servicio de búsqueda de cursos	*BusquedaCursos*

Tabla 5.4 Resumen de servicios generados por capas (LORS-SC)

5.2.3. Modelo de orquestación de servicios

En cuanto a la orquestación de servicios del sistema LORS-SC, comentar que no ha sufrido cambios sustanciales en cuanto a lo que la búsqueda federada sencilla se refiere. Los cambios se han producido a nivel interno de los servicios Web, como por ejemplo, modificando los datos que devolverán los métodos de consulta SQI (SQI 2.0 a partir de ahora), o añadiendo un nuevo método que posibilita la descarga del objeto de aprendizaje. Los cambios más importantes se han producido al añadir una nueva funcionalidad, como es la posibilidad de realizar una búsqueda de cursos completos como conjuntos de objetos de aprendizaje individuales. Por este motivo se presenta en este apartado cómo sería la orquestación de servicios de esta funcionalidad.

Para la búsqueda de un curso completo, el sistema recibirá en primer lugar la especificación de metadatos que utilizará el usuario y el conjunto de campos tipados que desee utilizar (como idioma, restricciones de los objetos, tipos de ficheros, dificultad, etc.). A continuación el usuario irá seleccionado de las diferentes ontologías disponibles (a través de su URL) los contenidos que desea para su curso. Podrá seleccionar tantos contenidos como desee de tantas ontologías como quiera. Esta funcionalidad será ofrecida por el servicio ***BusquedaCursos*** de la capa 3. Este servicio Web se pondrá en contacto con el servicio de la capa 2, también denominado ***BusquedaCursos***, el cual comenzará una búsqueda federada unitaria para cada uno de los módulos que desea el usuario para su curso. La búsqueda federada será realizada por el servicio Web ***BusquedaFederada*** como se vio anteriormente.

De este servicio Web se obtendrán todos los módulos unitarios que se encuentren en los repositorios distribuidos. Será el servicio Web ***ComposicionCursos*** el que se encargará de su composición. Como ya se ha explicado anteriormente, para la composición de los cursos se realizará una combinación de todos los módulos posibles, indicando en todo momento (en la fase de presentación) las composiciones realizadas y el grado de aproximación a los requisitos marcados por el usuario que se ha conseguido.

Dependiendo de la similitud entre la descripción del objeto de aprendizaje en cuestión y de la selección marcada por el usuario en la interfaz del entrada, se obtendrá una valoración del mismo. El servicio Web ***CatalogacionCursos*** deberá filtrar y ordenar todas esas combinaciones de cursos realizadas para mostrarle un listado ordenado al

usuario en el que no existan cursos iguales (formados por los mismos módulos – objetos de aprendizaje).

Una vez que el servicio de ***CatalogacionCursos*** finalice con su proceso, el usuario podrá seleccionar uno de ellos para su descarga. El servicio Web ***ComposicionCursos*** del nivel 2, compondrá la metainformación del mismo y lo empaquetará siguiendo la especificación IMS CC (Common Cartridge) [IMS, 2008]. Para la descarga de todos los objetos de aprendizaje individuales irá contactando con el servicio ***TargetLogic*** como se vio en el sistema LORS-SQI. Por último el servicio ***ComposicionCursos*** posibilitará el envío del curso al usuario siguiendo el mismo funcionamiento que se haría en caso de tratarse de un objeto individual, ya que, exteriormente se trata de un objeto de aprendizaje exactamente igual que el resto.

A continuación se muestra en la Figura 5.31 cómo sería el diagrama BPEL de esta parte del sistema:

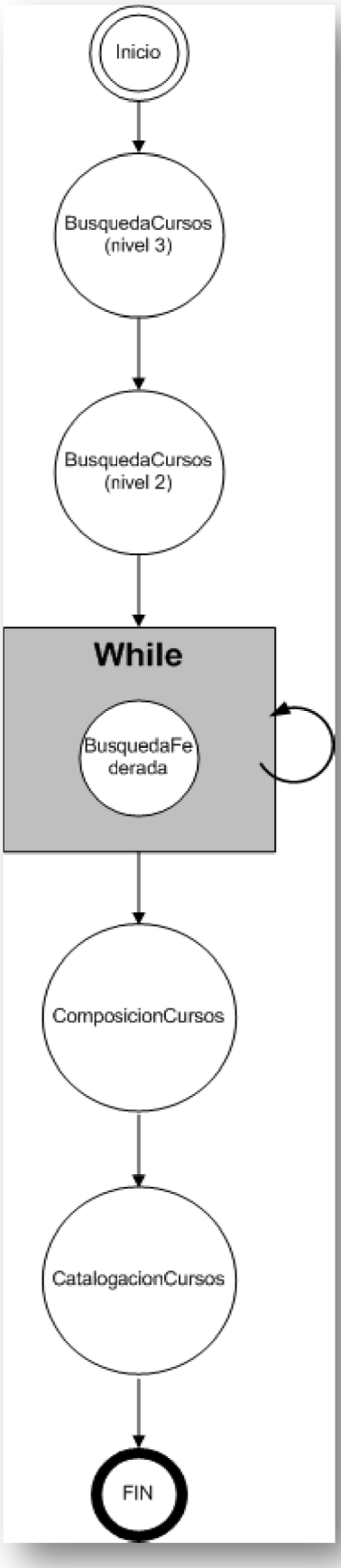


Figura 5.31 Modelo BPEL del servicio de búsqueda de cursos

Clases que implementan los servicios

En lo referente a las clases que implementan los servicios no existe ninguna reseña destacable, ya que la mayor parte de los cambios se han producido a nivel interno. Para los tres primeros requisitos que se comentaron en el apartado 5.2.1, los cambios se han producido en la clase TargetLogic, como se verá a continuación en su diagrama WSDL y el último requisito se acaba de ver en su modelo de orquestación.

Interfaz WSDL de los servicios

Como se acaba de comentar, el servicio Web asociado a cada repositorio ha sufrido algunas modificaciones para adaptarlo a una nueva interfaz de consulta a la que se ha denominado SQI 2.0, que recordemos que como cambios tenía:

- Unos métodos de consulta que ofrecen unos resultados más estructurados, lo cual facilitará enormemente su análisis y procesado, siguiendo la misma filosofía de ofrecer interoperabilidad e independencia entre sistemas.
- Un método de obtención del material educativo. En este caso se trata de un método completamente nuevo y que finaliza el proceso de reutilización de objetos de aprendizaje, ya que de poco sirve una especificación de consulta si luego no existe un método que posibilite su descarga.

En la Figura 5.32 se muestran los cambios acometidos contra el servicio Web TragetLogic para adaptarlo a SQI 2.0.

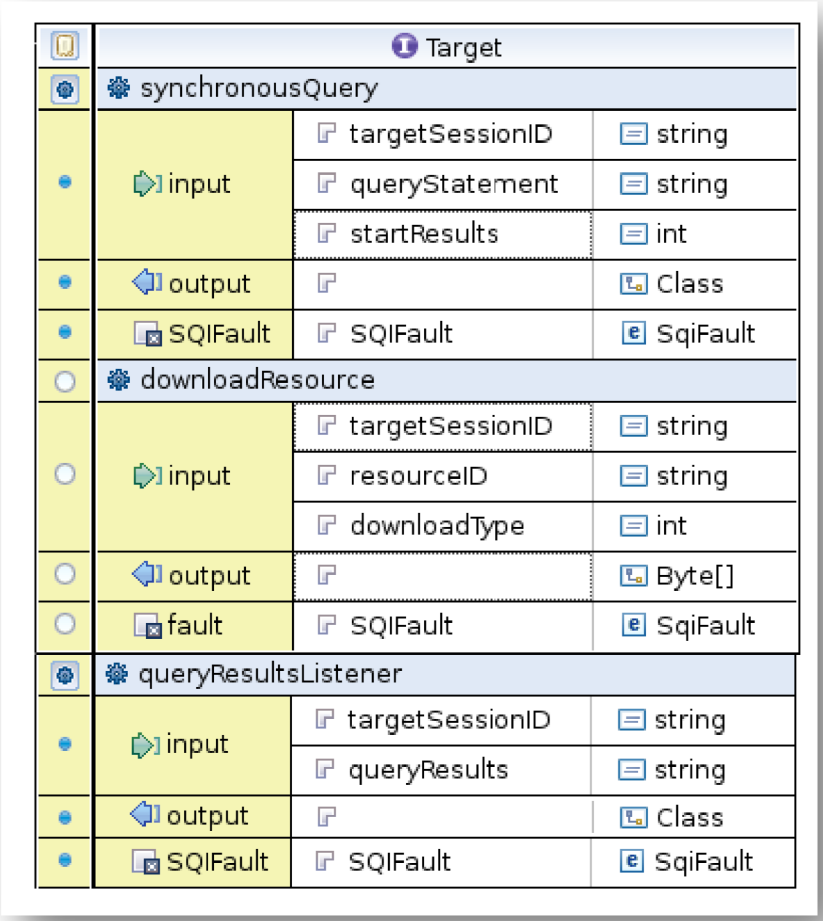


Figura 5.32 Diagrama WSDL del servicio Web de consultas SQI 2.0 (parte 1)

En la Figura 5.32 se podía ver que se han modificado los servicios de consulta (tanto síncrona como asíncrona) para mejorar la estructura de datos utilizada en la devolución de información y se ha añadido un método de recuperación del objeto de aprendizaje en sí. En los métodos de consulta se ha modificado el tipo utilizado para la devolución de datos, de forma que ahora en lugar del anterior String, se utilizará una estructura (Clase) para devolver los datos más estructurados y con ello ganar en ordenación, accesibilidad a los datos, mejoras en la reordenación, eliminación, etc. En lo que se refiere al método de descarga, recordar que el usuario haciendo uso del mismo podrá seleccionar entre descargar el fichero en formato estándar (array de bytes) o avanzado (enviando el fichero fuera del mensaje SOAP y posibilitar así el *Streaming* de bytes, mucho más eficiente).

Biblioteca de clases utilizada para el procesamiento ontológico (JENA)

Otro de los cambios que ha sufrido el sistema LORS-SC con respecto al anterior LORS-SQI, ha sido la modificación de sus sistemas de búsqueda para adaptarlos al uso de técnicas semánticas. En lo que al desarrollo de estos sistemas de búsqueda se refiere, destacar que se ha hecho uso del framework Jena [2009]. Se trata de un framework que cuenta con una API de programación para el manejo de ficheros OWL [W3C, 2004c] (Full, DL y Lite), RDF [W3C, 1999] y DAML+OIL [W3C, 2001c]. Como característica adicional, destacar que además de lo anterior, con el uso del framework Jena es posible crear modelos persistentes (persistidos de forma transparente en una base de datos relacional). Entre los gestores que soporta (MySQL, Oracle o PostgreSQL), se eligió MySQL por ser el gestor de base de datos utilizado en el desarrollo del sistema LORS-SQI.

A continuación se muestra un fragmento de código en el que se puede ver cómo se ha realizado en el sistema LORS-SC la lectura y procesado de ontologías:

```
Model model = ModelLoader.loadModel(fileOWL);
OntModel ontModel =
ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM_RULE_INF,model);

Individual ind = ontModel.getIndividual(indName);
if (ind == null) {
        …
} else {
        for (ExtendedIterator i= ind.listRDFTypes(false); i.hasNext(); ) {
                Resource r = (Resource) i.next();
                if (!r.isAnon()) … [tratamiento1]
        }

        for (StmtIterator st = ind.listProperties(); st.hasNext();) {
                Statement s = (Statement) st.next();
                Property  p = s.getPredicate();
                if (ontModel.contains(p,RDF.type,OWL.ObjectProperty)) {
                        Resource res = (Resource) s.getObject();
                        if (!res.isAnon())
                                [tratamiento2]
                }
        }
}
```

La lectura del modelo se realiza a través de la clase *Model*, la cual permitirá realizar una instancia del fichero físico que representa la ontología. El proceso iterativo irá recorriendo todos los recursos (clases), para obtener y dar tratamiento a cada uno de ellos (marcado en el código como tratamiento1). A continuación para cada propiedad, se obtendrá su predicado (en otro proceso iterativo) y de este predicado, se tendrá acceso al nombre, en lo que en el código se ha denominado tratamiento2.

5.2.4. Ontología sobre conceptos de la arquitectura empresarial de Java (JEE)

En este apartado se van a detallar los conceptos y actividades más importantes que se llevaron a cabo para la elaboración de una ontología de ejemplo basada en la arquitectura empresarial de Java (JEE) , la cual será utilizada para que el sistema sea capaz de recuperar cursos completos de esta temática, conociendo los módulos individuales que componen el dominio.

Para el desarrollo de la ontología, se siguieron los siete pasos que se recomiendan en [Noy, 2001].

Paso 1: Determinar el dominio y alcance de la ontología

Desde [Noy, 2001] sugieren comenzar el desarrollo de una ontología definiendo su dominio y alcance. Es decir, responder a varias preguntas básicas como:

- ¿Cuál es el dominio que la ontología cubriría?
- ¿Para qué será usada la ontología?
- ¿Para qué tipos de preguntas la información en la ontología debería proveer respuestas?
- ¿Quién usaría y mantendría la ontología?

Para la ontología que se desea construir, el dominio será el conjunto de módulos que componen todas las tecnologías de la arquitectura empresarial de Java. Naturalmente, los conceptos que describen diferentes tecnologías, irán ordenados a través de relaciones que determinen qué conocimientos previos son necesarios para poder empezar a estudiar una determinada tecnología. Así por ejemplo, para poder estudiar el concepto de los filtros con los que cuentan los Servlets, es necesario además de tener conocimientos de Java (relación con otra ontología sobre conceptos de la arquitectura estándar de Java – JSE), tener conocimientos del funcionamiento del ciclo de vida de los Servlets.

Si la ontología que se va a diseñar será usada para ayudar en el procesamiento del lenguaje natural de artículos, tales como un libro, sería importante incluir sinónimos e información de las varias clases de palabras a las cuales una palabra puede ser asignada para los conceptos de la ontología. Uno de los usos principales de este tipo de ontologías,

es la de saber qué módulos componen un determinado curso (JEE en este caso); así por lo tanto, será posible incluir en cada uno de ellos términos como dificultad, conocimientos previos, aptitudes necesarias, software que se necesita, etc. También es interesante tener en cuenta que si la gente que mantendrá la ontología describe el dominio en un lenguaje que es diferente del lenguaje que usan los usuarios de la ontología, será necesario proveer el mapeo entre los lenguajes.

Una de las formas de determinar el alcance de la ontología es bosquejando una lista de preguntas que la base de conocimientos basada en la ontología debería ser capaz de responder; preguntas de competencia [Gruninger and Fox, 1995]. Esas preguntas servirán después como prueba de control de calidad: ¿La ontología contiene suficiente información para responder esos tipos de preguntas? ¿Las respuestas requieren un nivel particular de detalle o representación de un área particular? Las preguntas de competencia son solamente un bosquejo y no necesitan ser exhaustivas. Así por ejemplo para este caso en particular:

- ¿Qué tecnologías tiene la arquitectura empresarial de Java?
- ¿Son las JSP parte de JEE?
- ¿Qué tecnología Java es la mejor para la capa de presentación?
- ¿Necesito conocer los Servlets para programar EJB's?

Paso 2: Considerar la reutilización de ontologías existentes

Según la recomendación seguida para el desarrollo de esta ontología, casi siempre vale la pena considerar lo que otra persona ha hecho y verificar si podemos refinar y extender recursos existentes para nuestro dominio y tarea particular. Rehusar ontologías existentes puede ser un requerimiento si nuestro sistema necesita interactuar con otras aplicaciones que ya se han dedicado a ontologías particulares o vocabularios controlados.

Uno de los principales beneficios de las ontologías es su formato electrónico, lo que hace que puedan ser importadas dentro de otros entornos. Así por ejemplo en este caso particular, sería posible la incorporación de otra ontología sobre conceptos de la arquitectura JSE de Java o la ya citada OntoGLOSE sobre conceptos de ingeniería del Software [Hilera et al., 2005].

Paso 3: Enumerar términos importantes para la ontología

Es útil escribir una lista con todos los términos con los que se quisiera hacer enunciados o dar explicación a un usuario. ¿Cuáles son los términos de los cuales se quisiera hablar? ¿Qué propiedades tienen esos términos? Por ejemplo, términos importantes relativos a la arquitectura JEE podrían ser: Servlets, jsp, ejb, servicios Web, etc. Y dentro de cada uno de ellos por ejemplo ciclo de vida de los Servlets, jstl, entity ejb o SOAP respectivamente; todos ellos conceptos de JEE. También será posible obtener términos relativos a las necesidades de cada una de las tecnologías, como por ejemplo, servidor de aplicaciones, base de datos relacional, o dentro ellos, glassfish o MySQL respectivamente.

Inicialmente es importante obtener una lista integral de términos sin preocuparse del recubrimiento entre los conceptos que representan, relaciones entre los términos, o cualquier propiedad que los conceptos puedan tener, o si los conceptos son clases o slots. Los siguientes dos pasos (desarrollando la jerarquía de clases y definiendo las propiedades de los conceptos (slots)) están estrechamente relacionadas. Es difícil hacer primero uno de ellos y luego hacer el otro. Típicamente se crean unas cuantas definiciones de los conceptos en la jerarquía y luego continuamos describiendo las propiedades de esos conceptos y así sucesivamente. Esos dos pasos son también los más importantes en el proceso de diseño de la ontología.

Paso 4: Definir las clases y la jerarquía de clases

Hay varios posibles enfoques para desarrollar una jerarquía de clases [Uschold and Gruninger, 1996]:

- Un proceso de desarrollo *top-down* comienza con la definición de los conceptos más generales en el dominio, para posteriormente ir categorizando cada uno de esos conceptos. Así por ejemplo es posible comenzar creando clases para los conceptos generales (cada una de las tecnologías de JEE), para posteriormente especializarnos en cada una de ellas, y así sucesivamente. Por ejemplo: JSP ➔ Acciones ➔ jsp:include
- Un proceso de desarrollo *bottom-up* comienza con la definición de las clases más especificas, las hojas de la jerarquía, con el subsecuente agrupamiento de esas clases en conceptos más generales. Por ejemplo, comenzamos definiendo todas las acciones estándar que tienen las páginas JSP, para

posteriormente agruparlas en el concepto acciones, que evidentemente estará dentro de la tecnología de las JSP.

- Un proceso de desarrollo *combinado* es el resultado de una combinación de los enfoques *top-down* y *bottom-up*: primero se definen los conceptos más sobresalientes y luego se generalizan y especializan apropiadamente. Por ejemplo se podría comenzar con unos cuantos conceptos de nivel superior como JSP o Servlets, y unos conceptos específicos, como jsp:include o el objeto HttpSession. Así será factible posteriormente relacionarlos en conceptos de nivel medio, tal como acciones o gestión de la sesión respectivamente.

A continuación se muestra en la Figura 5.33 sería una parte de la jerarquía mostrada por esta ontología, en la que se puede ver los tres niveles comentados anteriormente.

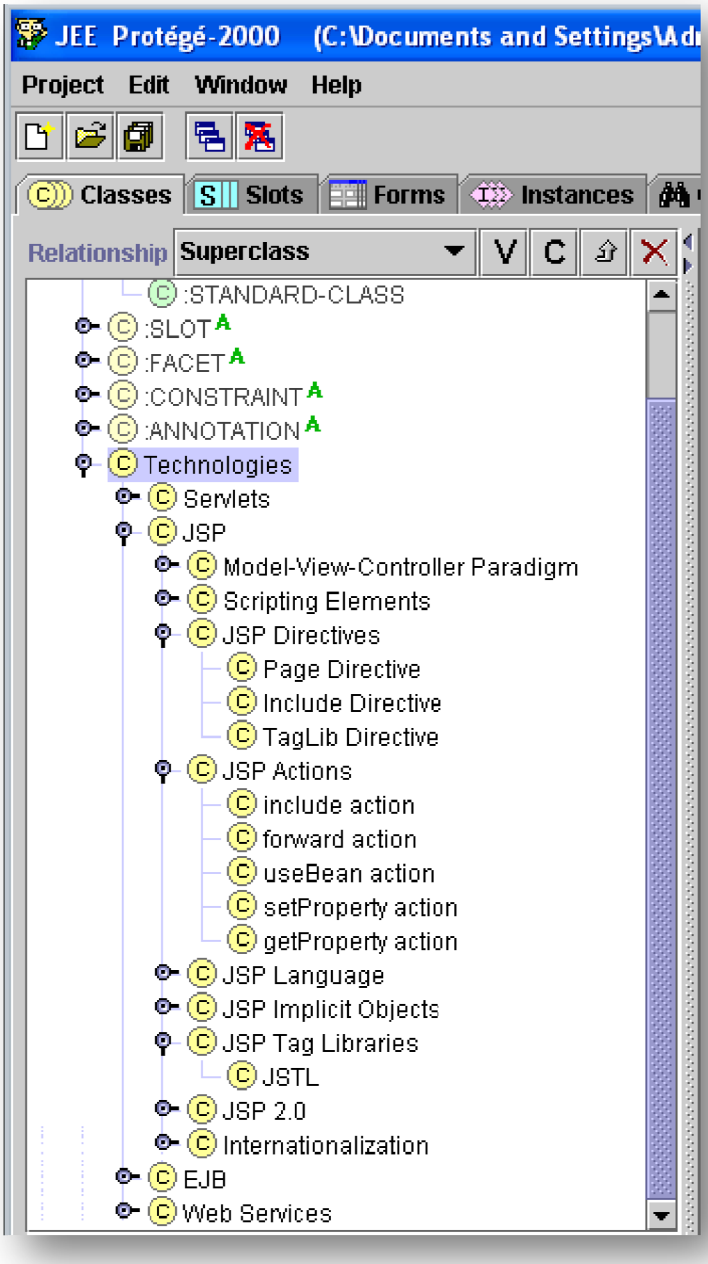


Figura 5.33 Los diferentes niveles de la ontología sobre JEE

Paso 5: Definir las propiedades de las clases (Slots)

Las clases aisladas no proveerán suficiente información para responder a las preguntas de competencia del paso 1. Una vez que se han definido algunas de las clases, es necesario ahora describir la estructura interna de los conceptos.

Ya se han seleccionado clases de la lista de términos creada en el paso 3. La mayoría de los términos restantes son muy probablemente propiedades de esas clases. Para cada propiedad en la lista, es necesario determinar qué clase es descrita por la propiedad. Esas propiedades se convierten en slots adosados a las clases. De esta forma, la clase JSTL por ejemplo tendrá los siguientes slots: dificultad, conocimientos previos, aptitudes necesarias, etc.

De esta forma, además de las propiedades que se han identificado previamente, es necesario añadir los siguientes slots a la clase JSTL, como muestra la Figura 5.34. Un slot deberá estar adosado a la clase más general que pueda tener esa propiedad; en nuestro caso a la clase del nivel superior, puesto que toda tecnología estará determinada por unos conocimientos previos, tendrá una dificultad, serán necesarias unas aptitudes previas, etc.

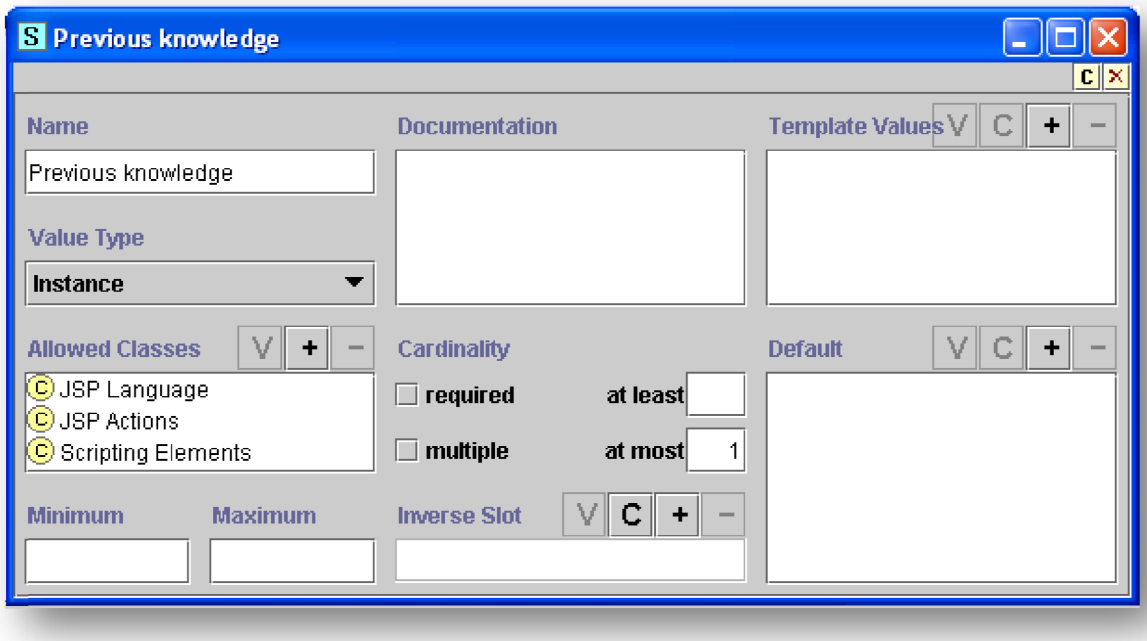


Figura 5.34 Slot de conocimiento previo necesario para el módulo de JSTL

Paso 6: Definir las facetas de los slots

Los slots puedes tener diferentes facetas que describen el tipo de valor, valores admitidos, el número de los valores (cardinalidad), y otras características de los valores

que los slots pueden tomar. Por ejemplo, el valor del slot nombre (como en “el nombre de una tecnología”) es una cadena de caracteres. Es decir, nombre es un slot con String como tipo de valor. El slot requisitos previos (como “las tecnologías que sería necesario conocer para poder comprender y manejar el término en sí”) puede tener valores múltiples y los valores son instancias de la clase Tecnologías. Es decir, requisitos previos es un slot con Instance como tipo de valor y Tecnologías como clase admitida.

La cardinalidad de un slot define cuantos valores un slot puede tener. Algunos sistemas solamente distinguen entre cardinalidad simple (admitiendo a lo sumo un valor) y cardinalidad múltiple (admitiendo cualquier cantidad de valores).

Una faceta tipo de valor describe qué tipos de valores pueden llenar el slot. A continuación se muestra una lista de los tipos de valores más comunes:

- String: es el tipo de valor más simple el cual es usado por slots tales como nombre; el valor es una simple cadena de caracteres
- Number (algunas veces los tipos de valores Float e Integer son usados por ser más específicos) describe slots con valores numéricos.
- Los slots del tipo Boolean son simples banderas si/no.
- Los slots del tipo Enumerated especifican una lista específica de valores admitidos para el slot. Por ejemplo, podemos especificar que el slot dificultad puede tomar uno de los siguientes valores posibles: muy baja, baja, intermedia, alta o muy alta. En Protégé [2009] los slots enumerados son del tipo Symbol.
- Los slots del tipo Instance admiten la definición de relaciones entre individuos. Los slots con tipo de valor Instance deben también definir una lista de clases admitidas de las cuales las instancias pueden provenir. Por ejemplo, el slot requisitos previos puede tener instancias de cualquier elemento de la ontología.

La Figura 5.35 se muestra la definición del slot dificultad en la clase JSTL, en la que se pueden ver todos los valores que podrá tomar.

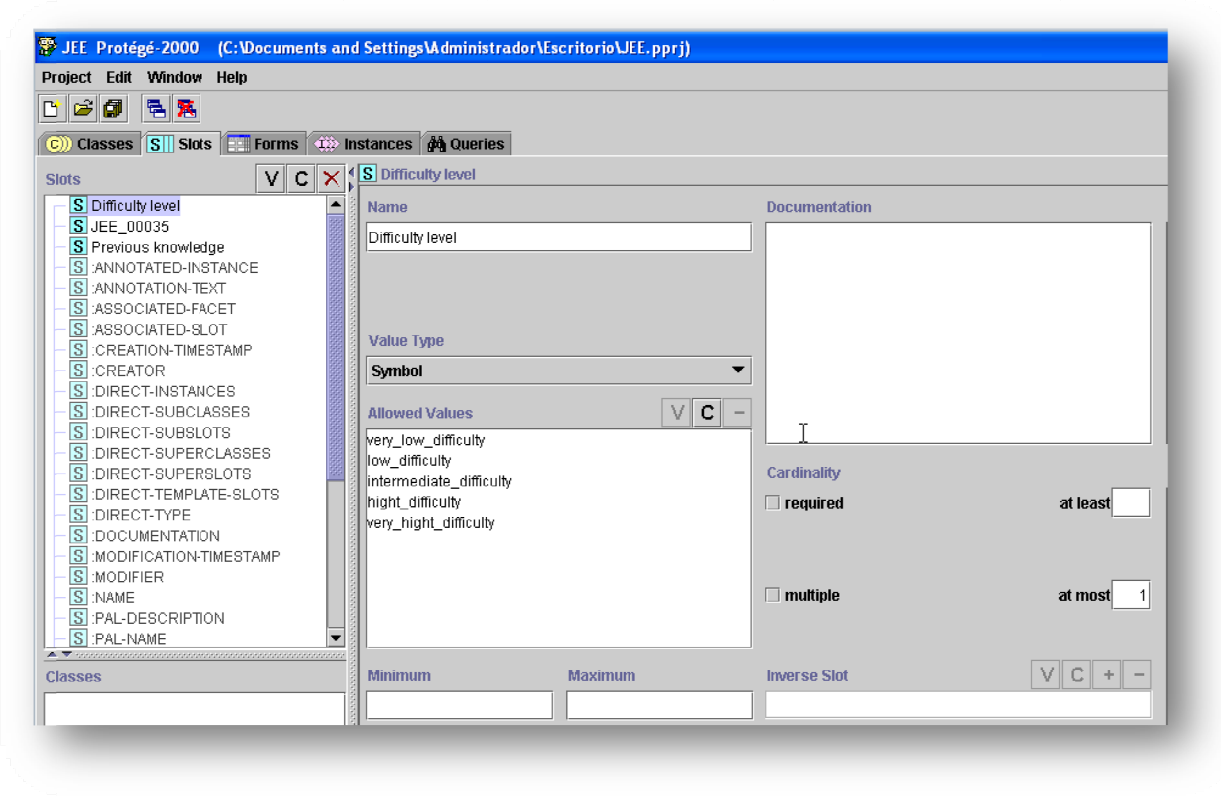


Figura 5.35 Slot de nivel de dificultad necesario para todos los módulos

Paso 7: Crear instancias

El último paso consiste en crear instancias individuales de clases en la jerarquía. La definición de una instancia individual de una clase requiere (1) elegir una clase, (2) crear una instancia individual de la clase y (3) rellenar los valores del slot. Por ejemplo, podemos crear una instancia individual como JSTL para representar un área de conocimiento dentro de la tecnología de los JSP. Esta instancia tiene definidos los siguientes valores de slot:

- Dificultada: Intermedia
- Requisitos Previos: JSE, Elementos del lenguaje JSP, Directivas, Acciones, XML.
- Software necesario: JDK, servidor de aplicaciones ligero.

Por último se muestra un pequeño esquema del código OWL (XML) generado tras la realización de estos siete pasos.

```
xml version="1.0"?>
<!DOCTYPE Ontology [
   <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
   <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
   <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
   <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
   <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
   <!ENTITY JEE "http://www.semanticweb.org/ontologies/2009/8/JEE.owl#" >
   <!ENTITY jsp "http://www.semanticweb.org/ontologies/2009/8/JEE.owl#jsp:" >
]>
<Ontology xmlns="http://www.w3.org/2006/12/owl2-xml#"
    xml:base="http://www.w3.org/2006/12/owl2-xml#"
    xmlns:JEE="http://www.semanticweb.org/ontologies/2009/8/JEE.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
    xmlns:jsp="&JEE;jsp:"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    URI="http://www.semanticweb.org/ontologies/2009/8/JEE.owl">
   <SubClassOf>
      <Class URI="&JEE;EJB"/>
      <Class URI="&JEE;Technologies"/>
   </SubClassOf>
   <Declaration>
      <Class URI="&JEE;EJB"/>
   </Declaration>
<SubClassOf>
      <Class URI="&JEE;JSP"/>
      <Class URI="&JEE;Technologies"/>
   </SubClassOf>
   <Declaration>
      <Class URI="&JEE;JSP"/>
   </Declaration>
   <SubClassOf>
      <Class URI="&JEE;JSP_2.0"/>
      <Class URI="&JEE;JSP"/>
   </SubClassOf>
```

5.2.5. Interfaces del sistema

Para validar la implementación de la arquitectura presentada en el capítulo anterior, se ha modificado la aplicación LORS-SQI, añadiendo y modificando los diversos servicios comentados anteriormente, obteniendo así el sistema denominado LORS-SC. Se trata, al igual que antes, de una aplicación Web que en su página principal ofrece al usuario la posibilidad de realizar una búsqueda federada en distintos repositorios distribuidos, la de registrar un nuevo repositorio en el sistema, y como novedad, la posibilidad de realizar la búsqueda de un curso completo de aprendizaje. A través de la Figura 5.36 se puede ver la nueva interfaz de entrada al sistema:

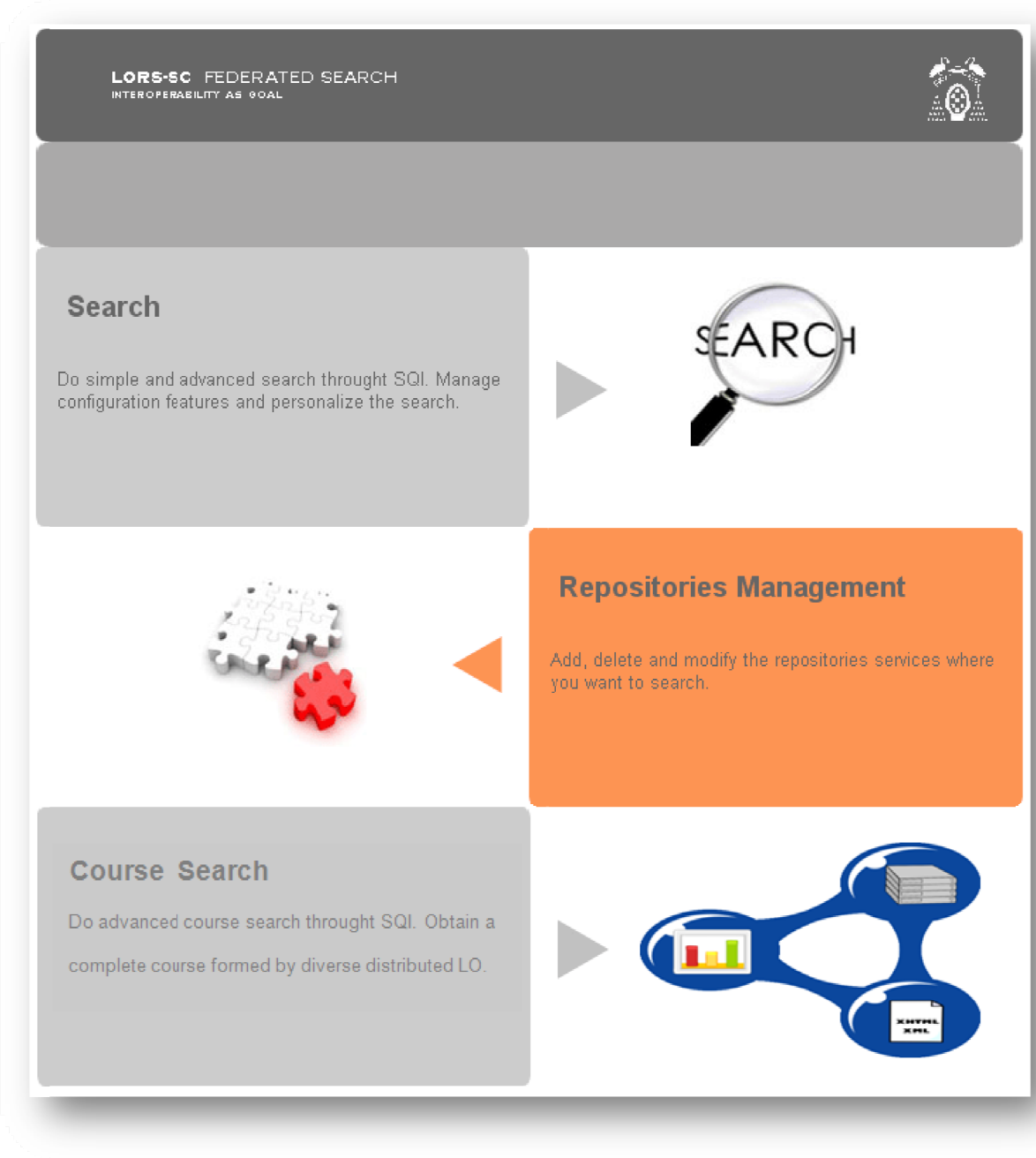


Figura 5.36 Interfaz de entrada en LORS-SC

Las otras dos funcionalidades permanecen inalteradas en lo que a las interfaces del sistema se refiere, por lo que en este apartado solamente se comentará la parte referente a la búsqueda de cursos completos de aprendizaje. Se pude ver en la Figura 5.37, cómo sería la interfaz de entrada de datos resultante. En ella, el usuario podrá, en primer lugar, configurar varios parámetros SQI como el número máximo de resultados, el tamaño del conjunto de resultados o el resultado inicial (estos datos serán utilizados por el sistema para la obtención de los objetos de aprendizaje individuales, no para la devolución del curso completo al usuario).

La búsqueda del curso se trata como si fuera una búsqueda exacta simple. Los campos que se utilizarán para la búsqueda, al igual que antes, están definidos en el fichero XSD. Todos estos campos se pueden utilizar para delimitar las búsquedas de objetos individuales, para que por ejemplo todos los módulos sean en español o no tengan copyright. Una vez que se rellenen estos datos el usuario podrá pasar a determinar los módulos que desea que se incluyan en su curso a través de la opción correspondiente mostrada también en la Figura 5.37.

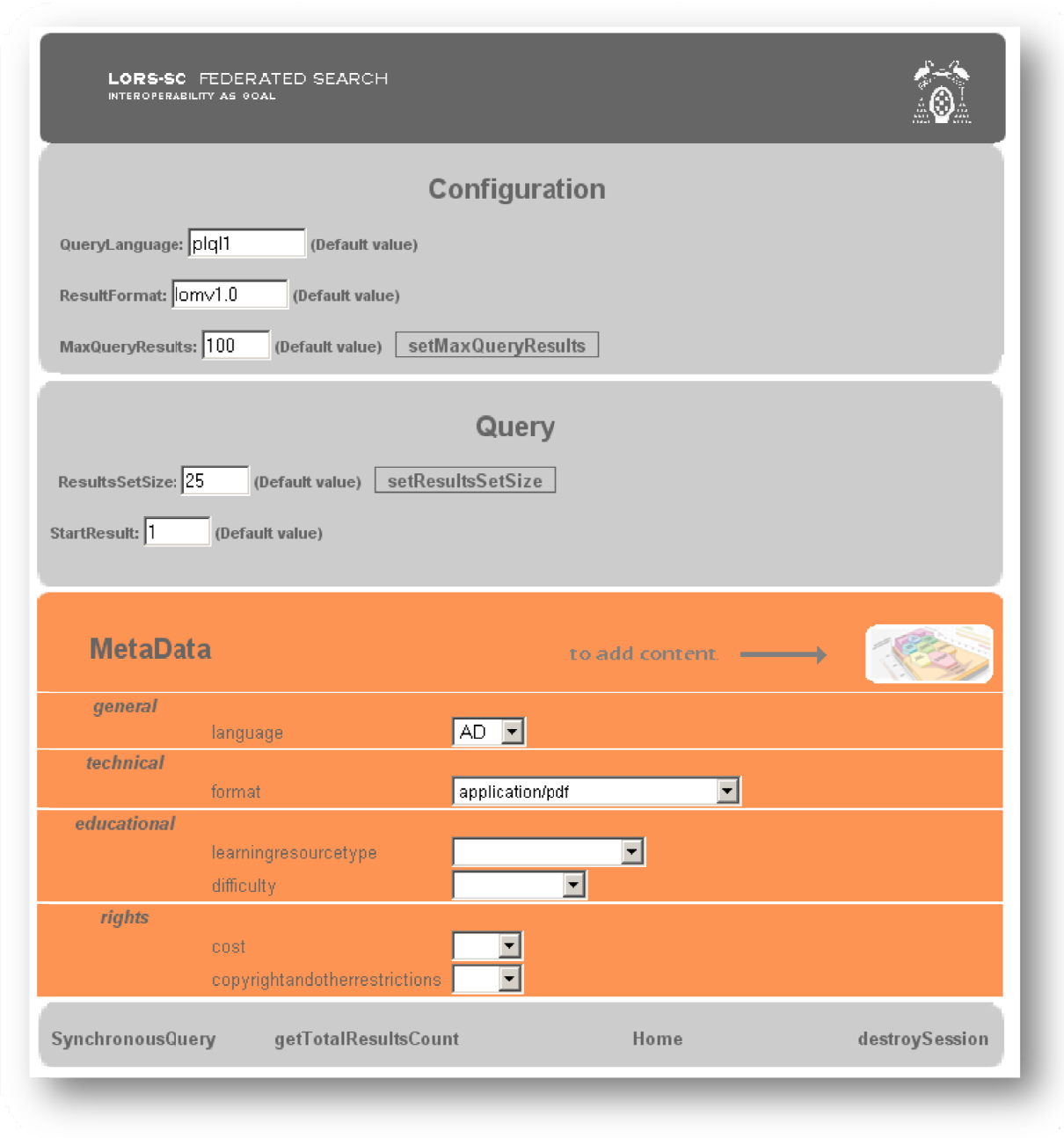


Figura 5.37 Interfaz de consulta de cursos completos de LORS-SC

Una vez que el usuario accede a la funcionalidad de añadir contenidos, le aparecerá una ventana como la que muestra la Figura 5.38. En esta ventana el usuario podrá ir seleccionando de las diferentes ontologías a las que acceda (a través de su URL) los contenidos que quiera. Se puede ver cómo irán apareciendo en forma de árbol estructurado y de ellos irá marcando todos los que desee. Una vez seleccionados todos los contenidos individuales, guardará los cambios y podrá realizar la consulta, comenzando el proceso de búsqueda federada individual que se explicó en el modelo de orquestación de servicios comentado en este apartado.

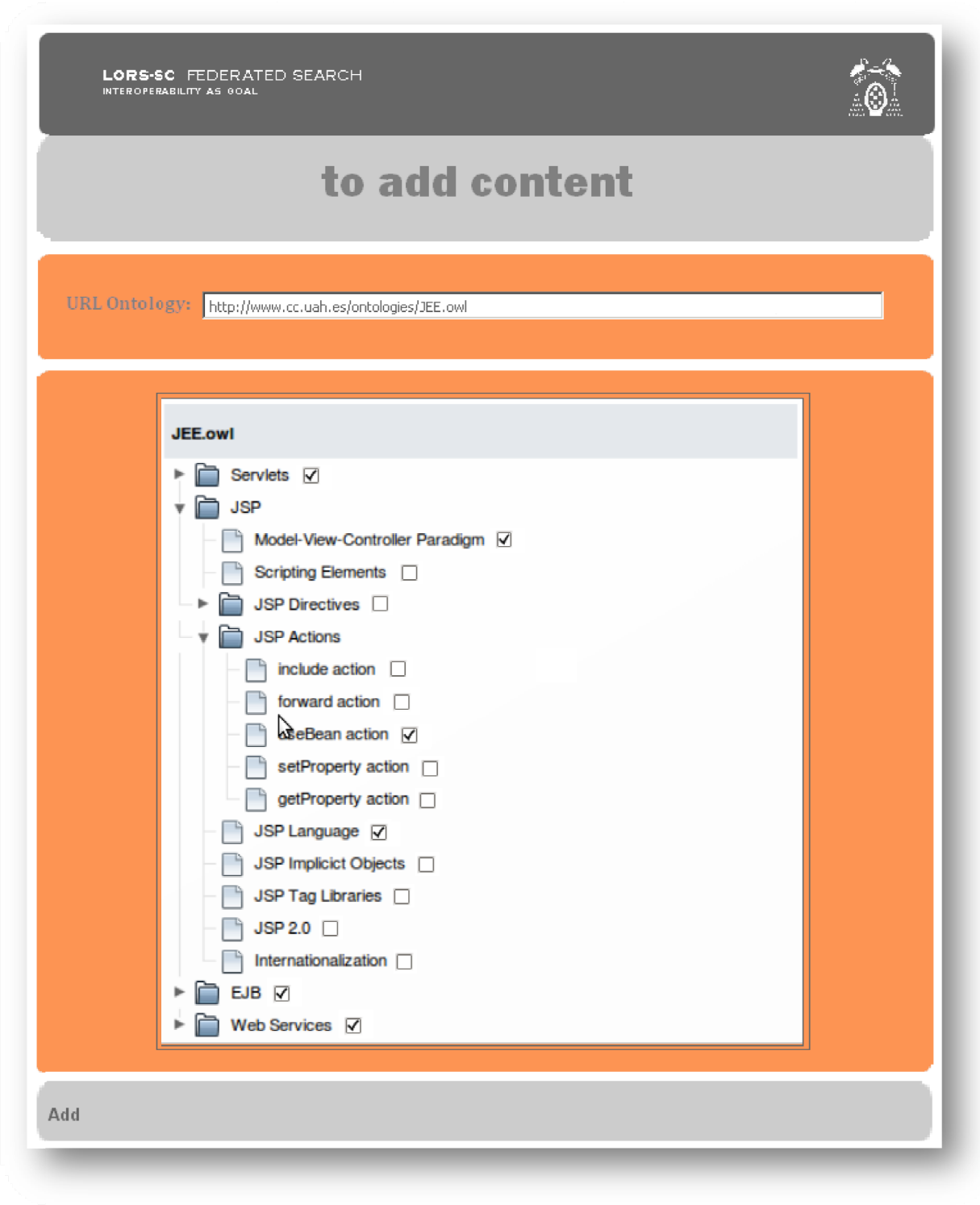


Figura 5.38 Interfaz de selección de contenidos de LORS-SC

Cuando el usuario haya seleccionado los contenidos de la ontología en cuestión, podrá seleccionar otra (también a través de su URL). Para guardar los contenidos anteriores deberá pulsar sobre el botón “Add”. Cuando finalice este proceso podrá cerrar

esta ventana y a través de la interfaz mostrada en la Figura 5.37 pulsará sobre la opción de búsqueda para comenzar este proceso.

Tras la realización de las búsquedas federadas individuales, el sistema construye una serie de cursos completos (realizando combinaciones de todos los recursos individuales que ha obtenido). A continuación le mostrará una relación de todos ellos, indicando en todo momento de qué repositorio se ha obtenido cada recurso individual, el grado de coincidencia de cada uno de ellos con el marcado por el usuario en la interfaz de búsqueda y el grado de coincidencia de todo el curso (realizando una media de todos los recursos). Una vez generado, el usuario podrá descargarlo haciendo uso del método de descarga anteriormente comentado y que a partir de ahora tendrá la interfaz de consulta propuesta denominada SQI 2.0. Se puede ver en la Figura 5.39 cómo sería la interfaz de descarga de este sistema de obtención de cursos completos de aprendizaje.

El usuario podrá seleccionar cada uno de los cursos que el sistema ha confeccionado (ordenados por índice de coincidencia con los requisitos marcados por el usuario), y verificará los contenidos incluidos en el (origen del recurso, grado de coincidencia, etc.). Para procesar la descarga del curso completo, tan solo tendrá que hacer clic en el número de curso una vez que este esté abierto y comenzará un proceso de descarga del curso completo, junto con su nueva metainformación tal y como se explicó en el apartado anterior (el proceso de descarga sería similar al de descarga de un contenido individual).

En esa interfaz de descarga se pueden ver, como ya se ha comentado, los módulos incluidos en el curso. Sirva de ejemplo esta captura de pantalla para verificar cómo se han incluido cinco módulos con un nivel de coincidencia del 100% y uno con un nivel del 90%, eso quiere decir que no se ha incluido un modulo exacto al que buscaba el usuario. En la Figura 5.38 se marcó el módulo “useBean action” de JSP y se consiguió el objeto de aprendizaje “JSP Actions” (mostrado en la Figura 5.39) que al ser más general e incluir más cosas, se calificó con un 90% de aproximación, de ahí la diferencia.

Figura 5.39 Interfaz de resultados obtenidos de LORS-SC

5.3. PRUEBA DEL PROTOTIPO LORS-SC

En este apartado se van a detallar los distintos usos que se le han dado hasta la fecha al prototipo desarrollado (LORS-SC) con el fin de demostrar la eficiencia y funcionalidad tanto del sistema, como de la arquitectura que lo sustenta, ambas cometido principal de esta Tesis.

Durante el curso escolar 2009/2010, la Universidad de Alcalá oferta entre su gran variedad formativa, dos másteres propios en Informática, dos títulos propios en especialización en Informática y un máster oficial también en Informática; presentándose todos ellos en modalidad de teleformación (virtual con tutorizaciones). Dependiendo del tipo de estudio seleccionado, tendrá una duración y unas asignaturas específicas, para lo cual, se presenta la Tabla 5.5.

ID[4]	TIPO	NOMBRE	DURACIÓN
1	Máster oficial	Máster Universitario en Ingeniería del Software para la Web	1 año (60 ECTS)
2	Máster propio	Máster en Lenguajes e Ingeniería del Software Avanzada	2 años (104 ECTS)
3	Máster propio	Máster en Lenguajes e Ingeniería Web Avanzada	2 años (104 ECTS)
4	Especialización	Especialización en Ingeniería del Software Avanzada	1 año (60 ECTS)
5	Especialización	Especialización en Desarrollo de Aplicaciones Web Avanzadas	1 año (60 ECTS)

Tabla 5.5 Resumen de algunos de los másteres que imparte la Universidad de Alcalá durante el curso 2009 / 2010

En todos estos estudios, como se ha comentado, se imparten diferentes asignaturas, algunas de ellas con algunos contenidos similares. A continuación se muestra en la Tabla

[4] La columna ID se utiliza para relacionar los másteres con sus nombres en la Tabla 5.5Tabla 5.5 con los nombres de asignaturas de la Tabla 5.6

5.6 las asignaturas que se imparten en cada máster (se mostrarán solamente las que interesan):

ID	NOMBRE
1	Herramientas de programación Web (12 créditos)
2	Herramientas avanzadas de programación I (12 créditos)
2	Herramientas avanzadas de programación II (12 créditos)
3	Herramientas avanzadas de programación (12 créditos)
3	Herramientas de programación Web (12 créditos)
4	Java SE (6 créditos)
4	Java JDBC (6 créditos)
5	Java SE (6 créditos)
5	Java JDBC (6 créditos)
5	Java EE (6 créditos)
5	Servicios Web en Java (6 créditos)

Tabla 5.6 Resumen de las asignaturas de los másteres

Como se ha comentado anteriormente, se trata de estudios en modalidad de teleformación, por lo que una plataforma educativa (un LMS por ejemplo), se encargará de mostrar los diferentes contenidos educativos a los alumnos. El problema residiría en que, como se puede apreciar en la Tabla 5.6, muchas de las asignaturas comparten contenidos, por lo que será necesario generar para cada asignatura el objeto de aprendizaje que incluya todos los contenidos, de forma que el LMS de turno sea capaz de mostrarlo al usuario. Esto va completamente en contradicción con el principio de reutilización de los objetos de aprendizaje. En este sentido es donde entra en juego el prototipo LORS-SC, ¿por qué no generar objetos de aprendizaje individuales y conformar cursos en base a estos objetos individuales? De esta manera, cada curso, en función de sus necesidades contará con unos u otros contenidos, de forma que así se reutilicen los mismos, consiguiendo con ello una reducción notable en los tiempos de elaboración y mantenimiento del material educativo.

Así se hizo; el departamento de Ciencias de la Computación de la Universidad de Alcalá, que es el encargado de la impartición y tutorización de estas asignaturas, desplegó cuatro repositorios que se conocen con los nombres de UAH/CC1 Repository, UAH/CC2 Repository, UAH/CC3 Repository y UAH/CC4 Repository. En estos repositorios se ubicaron diferentes objetos de aprendizaje individuales a los que atacaba el sistema LORS-SC. A través del uso de la ontología creada en el apartado 5.2.4, y de otra ontología similar pero para la edición estándar de la arquitectura de programación Java (JSE), ha sido posible irle indicando al sistema LORS-SC cuáles eran los contenidos que se deseaba para cada asignatura de los máster, de forma que se ha conseguido por una parte la reutilización de los mismos, y por otra, probar completamente la validez de los objetivos marcados al comienzo de esta Tesis, ya que el sistema LORS-SC, y la arquitectura que lo sustenta, son sistemas completamente interoperables que posibilitan la generación de cursos completos de aprendizaje a partir de contenidos individuales, utilizando para ello sistemas de búsqueda semánticos.

Sirva como ejemplo las siguientes asignaturas:

- Herramientas avanzadas de programación del Máster propio en Lenguajes e Ingeniería Web Avanzada
- Java JDBC del Título de Especialización en Ingeniería del Software Avanzada

Ambas asignaturas contienen una parte común (JDBC, Java EE y una parte de los Servlets), mientras que la otra es específica de cada una de ellas. Se puede ver en la primera de ellas los contenidos de JDBC, JDBC Avanzado, Java EE, JSP/Servlets y EJBs a través de la Figura 5.40, mientras que los contenidos de la segunda son los de JDBC, Java EE, ciclo de vida de los Servlets, JDBC 2.0 e Hibernate, mostrados en la Figura 5.41.

Para su elaboración tan solo ha sido necesario ir marcando los contenidos individuales que se necesitaban para cada asignatura, y el sistema LORS-SC se encargó de componer los cursos para subirlos al gestor de contenidos (Blackboard en este caso), consiguiendo así una reutilización de los mismos y comprobar de manera empírica el funcionamiento del sistema por un lado, y la validez de la arquitectura presentada por el otro.

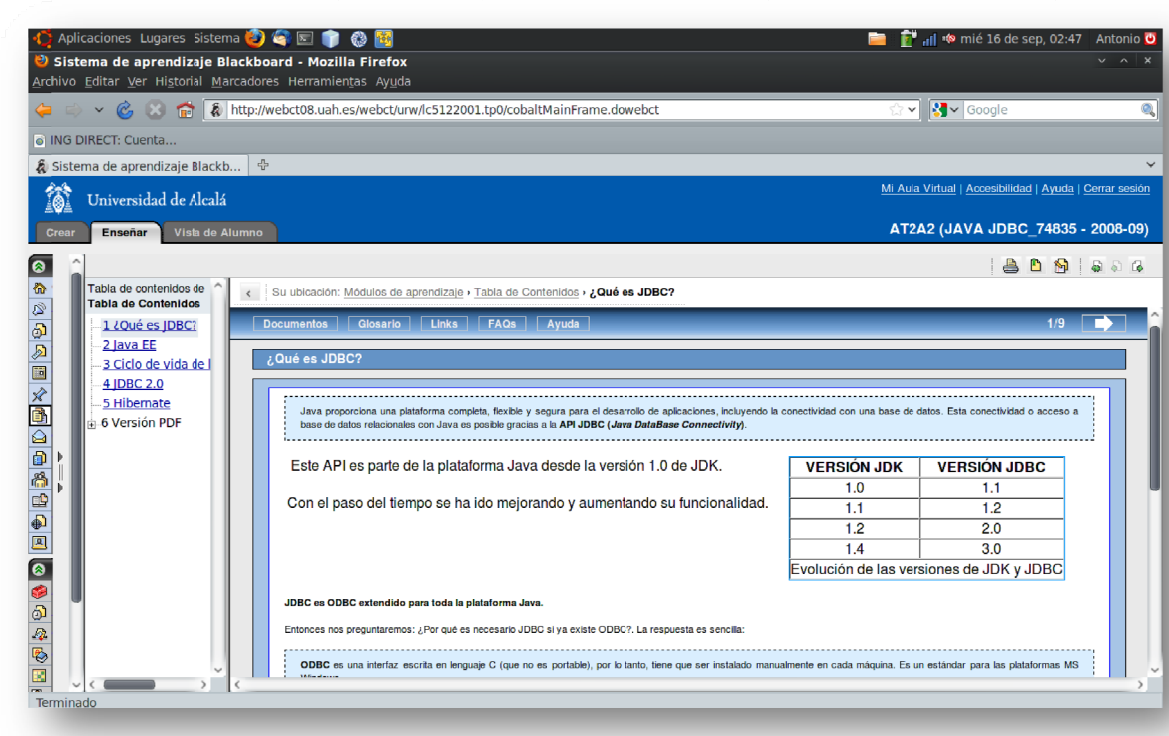


Figura 5.40 Contenidos de la asignatura del máster propio

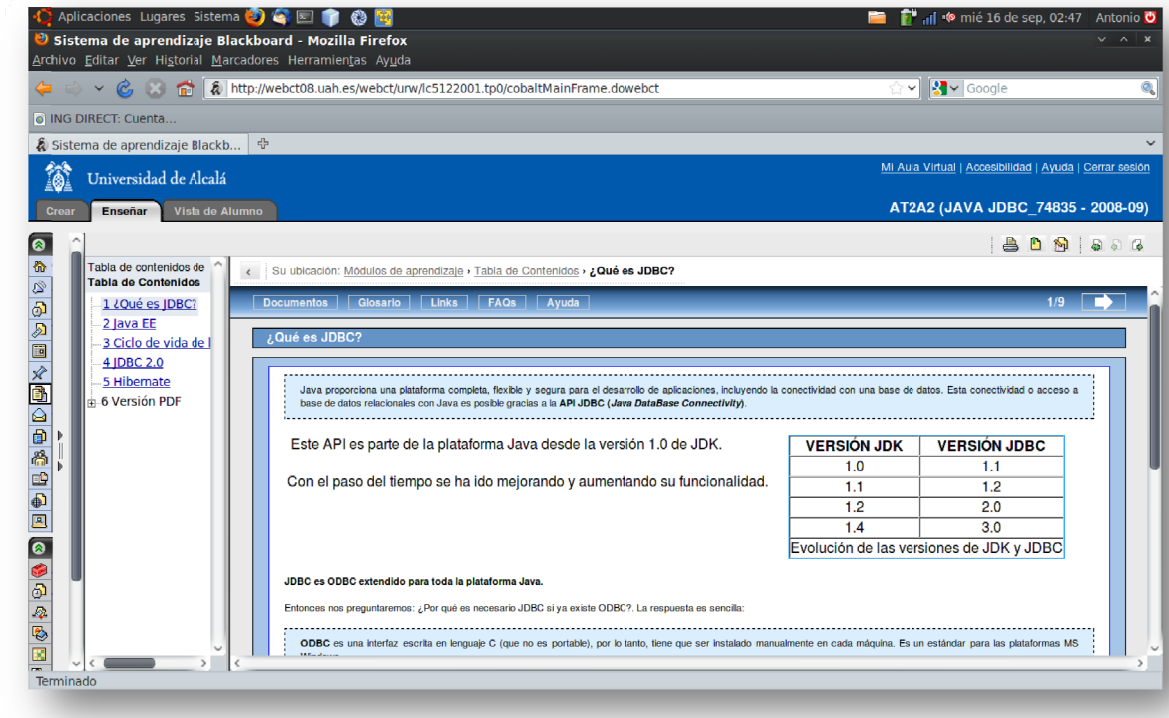


Figura 5.41 Contenidos de la asignatura del título de especialización

6. CONCLUSIONES Y FUTURAS LÍNEAS DE INVESTIGACIÓN

En este capítulo se recogen las conclusiones de la investigación realizada; muchas de ellas, aunque no todas, ya explicadas en los diversos apartados de conclusiones parciales de los capítulos anteriores. Se plantearán algunas futuras líneas de investigación que han surgido del trabajo y que supone, todavía en mayor medida la evolución de la arquitectura y el prototipo construido, y que, desde el año 2005 está evolucionando. Antes de acometer estos apartados se enumerarán los objetivos que se perseguían al comienzo del trabajo para determinar dónde se han logrado, así como las aportaciones principales derivadas del esfuerzo investigador.

6.1. OBJETIVOS Y APORTACIONES

Conviene recordar el objetivo inicial que se propuso al iniciar este trabajo de investigación, el cual era enunciado de la siguiente forma:

Adaptación y actualización de una arquitectura software capaz de cubrir la necesidad de reutilizar objetos de aprendizaje almacenados en diferentes repositorios, que permitirá localizarlos independientemente de su ubicación física y de su tecnología de almacenamiento a través de Internet de manera que haga que los sistemas de teleformación sean interoperables. Aportando además sistemas que optimicen la búsqueda de estos contenidos mediantes técnicas semánticas, así como que permita la creación de cursos completos como resultado de la agregación de diversos objetos de aprendizaje provenientes de diferentes repositorios distribuidos.

Para alcanzar este objetivo se plantearon otros objetivos intermedios. En la Tabla 6.1 se muestran estos objetivos y en qué capítulo de la Tesis se ha conseguido satisfacer dicho objetivo.

OBJETIVO	CAPÍTULO
1. Definir una arquitectura orientada a servicios para la localización universal de objetos de aprendizaje, incluyendo la composición de cursos formados por la agregación de diferentes objetos de aprendizaje de diversos repositorios distribuidos	Capítulo 4
2. Describir cada uno de los servicios presentes en la arquitectura y su organización en la misma.	Capítulo 4
3. Estudiar las características de los repositorios existentes en la actualidad y su adaptabilidad a las especificaciones y estándares.	Capítulos 2 y 3
4. Facilitar la interoperabilidad de sistemas de teleformación aunque utilicen distintos estándares de metadatos de los objetos docentes que intercambian.	Capítulo 4
5. Ofrecer una capa de abstracción superior con servicios comunes disponibles para todos los sistemas que cubran sus necesidades de comunicación.	Capítulo 4
6. Basar la propuesta en estándares reconocidos y sólidos.	Capítulos 2 y 4
7. Analizar las diferentes deficiencias de un sistema de reutilización de objetos de aprendizaje ya diseñado anteriormente y verificar sus posibilidades de adaptación.	Capítulo 4

8. Crear una nueva interfaz para la consulta que permita la obtención sencilla del contenido buscado, así como que posibilite la descarga del material. Esta interfaz estará basada en la especificación SQI y se denomina SQI 2.0	Capítulo 4
9. Definir el uso de técnicas semánticas para el acceso eficiente a la metainformación con la que se describen a los objetos de aprendizaje	Capítulo 4
10. Definir una arquitectura que posibilite la búsqueda y descarga de cursos completos como conjuntos de objetos de aprendizaje individuales	Capítulo 4
11. Validar la arquitectura propuesta.	Capítulo 5

Tabla 6.1 Objetivos de la Tesis y capítulo donde se justifica su cumplimiento

Las aportaciones principales que se han llevado a cabo a lo largo de la investigación se pueden resumir en los siguientes artículos enviados a diversos congresos, tanto nacionales como internacionales:

5. Ortiz, A., Otón, S, Barchino, R., "Arquitectura para publicación y localización universal de Objetos de Aprendizaje mediante Servicios Web". I Congreso IberoAmericano sobre Computación Ubicua (CICU 05). 2005.

En este trabajo se presentaron las primeras ideas y características que formarían parte de la arquitectura de la que parte el estudio de esta Tesis. Se plantearon las ideas de la utilización de los servicios Web asociados a cada repositorio para dar acceso a sus contenidos, y una forma de realizar conversiones de metadatos utilizando Dublin Core como formato intermedio. Las principales aportaciones en este congreso se publicaron además en el CEUR Workshop Proceedings número 132, así también se destaca la siguiente publicación:

6. Otón, S., Hilera, J.R., Gutiérrez, I., Ortiz, A. "Learning Objects universal publishing and location Architecture using Web Services". CEUR Workshop Proceedings Vol. 132 (CEUR-WS.org). 2005.

7. Otón, S., Hilera, J.R., Gutiérrez, I., Ortiz, A., "Arquitectura orientada a servicios Web para la implementación de repositorios distribuidos de objetos y unidades de aprendizaje". I Simposio Nacional de Tecnologías de la Información y de las Comunicaciones en la Educación (SINTICE 2005).

En este trabajo se presentó la primera versión de la arquitectura que sirve como base para el desarrollo elaborado en la Tesis. En él se planteaba la utilización de los servicios Web y SOA para la construcción de la arquitectura.

8. Otón, S., Ortiz, A., Barchino, R., "Arquitectura orientada a servicios para la reutilización de objetos de aprendizaje distribuidos". IADIS Conferencia Ibero Americana WWW/Internet 2006.

Este es el último trabajo relacionado con la primera arquitectura presentada en un congreso, en él se explica la última versión de la arquitectura así como el sistema LORS implementado con toda su funcionalidad. También se editó este trabajo en la revista RIICU (Revista Internacional Iberoamericana sobre Computación Ubicua:

9. Otón, S., Ortiz, A. "Búsqueda de objetos de aprendizaje para dispositivos móviles en repositorios distribuidos". RIICU (Revista Internacional Iberoamericana sobre Computación Ubicua). Magazine nº 1.

10. Otón, S., Ortiz, A., "Búsqueda de objetos de aprendizaje para dispositivos móviles en repositorios distribuidos". II Congreso IberoAmericano sobre Computación Ubicua (CICU 06). 2006.

En este trabajo ya se propone la utilización del prototipo implementado en la Tesis para la integración de dispositivos móviles en el proceso de búsqueda de contenidos docentes distribuidos. Se realizará un estudio de esta idea en el apartado de futuras líneas de investigación.

11. Ortiz, A., Otón, S., Hilera J.R., "Hacia la utilización de Servicios Web Semánticos en Repositorios de Objetos de Aprendizaje". III Simposio Pluridisciplinar sobre Objetos y Diseños de Aprendizaje Apoyados en la Tecnología (od@06). 2006.

En este trabajo ya se estudiaron diferentes tecnologías que podrían mejorar las propiedades de descubrimiento, accesibilidad y reutilización universal de objetos de

aprendizaje almacenados en repositorios. Todas estas adaptaciones son las que conforman una de las partes fundamentales del cometido esta Tesis.

12. Ortiz, A., de Blas, J.M., Otón, S., Barchino, R., Gutiérrez, J.A. "Interoperabilidad de Servicios Web entre Plataformas". IADIS Ibero-Americano WWW/Internet 2006.
13. de Blas, J.M., Ortiz, A., Hilera, J.R., Gutiérrez, J.M., Martínez, J.J. "Resolución de Problemas de Interoperabilidad entre dos aplicaciones de e-learning basadas en Servicios Web". IADIS Ibero-Americano WWW/Internet 2006.

En estos dos artículos se empezaron a analizar los principales problemas que planteaba la primera versión del sistema (denominado LORS). Sobre todo se hacía hincapié en las problemáticas surgidas a la hora de interoperar diferentes sistemas SOA, o diferentes arquitecturas de aprendizaje respectivamente.

14. Otón, S., Ortiz, A., Hilera, J.R., "SROA: Sistema de Reutilización de Objetos de Aprendizaje". VIII Congreso Iberoamericano de Informática Educativa. 2006.

En este artículo se presenta el sistema LORS ya completamente diseñado y operativo en diversas actividades formativas desarrolladas dentro del Departamento de Ciencias de la Computación de la Universidad de Alcalá, como se verá más adelante. Como resultado de este artículo, se le propuso la elaboración de un documento más extenso para su publicación en la Revista Iberoamericana IE Comunicaciones, así pues surge esta contribución:

15. Otón, S., Ortiz, A., Hilera, J.R., "SROA: Sistema de Reutilización de Objetos de Aprendizaje". Revista Iberoamericana de Informática Educativa IE Comunicaciones. Enero – Junio 2007.

16. Ortiz, A., Otón, S., Hilera, J.R., Gutiérrez, J.A., de Blas, J.M., "Web Services Coordination Model for Federated Search in Learning Objects Repositories". IX Simposio Internacional de Informática Educativa. SIIE 2007.
17. Otón, S., Ortiz, A., Barchino, R., Gutiérrez, J.M., "Adaptabilidad de Repositorios de Objetos de Aprendizaje para su integración en Sistemas de

Búsquedas Federadas". IX Simposio Internacional de Informática Educativa. SIIE 2007.

18. Ortiz, A., de Blas, J.M., Gutiérrez, J.M., "Adjustment needs in SOA based e-Learning Applications". International Conference on Web Information Systems and Technologies WEBIST 2007.

Estos tres artículos tratan de identificar las diferentes problemáticas detectadas en el primero de los sistemas, así como plantear soluciones de cara a garantizar la completa integración entre repositorios y sistemas de aprendizaje por un lado y entre repositorios y sistemas de búsqueda por el otro.

19. Otón, S., Ortiz, A., Hilera, J.R., Martínez, J.J., Barchino, R., Gutiérrez, J.M., Gutiérrez, J.A., De Marcos, L., "The Integration of SQI in a Reusable Learning Objects System: Advantages and Disadvantages". X International Conference on Information Integration and Web-Based Application & Services. IIWAS 08.

En este artículo se presenta por primera vez la arquitectura LORA-SQI. Esta arquitectura tiene entre sus novedades la incorporación de la especificación de consulta estándar SQI, la cual dotará a la arquitectura de la interoperabilidad y accesibilidad necesaria en una arquitectura de estas características. También se realiza en el mismo, un estudio de la presencia o no de esta especificación en los principales repositorios y sistemas de búsqueda del mercado, verificando así la necesidad de adaptación al convertirse SQI en un imperamento si se desea hacer completamente accesible un sistema de búsqueda.

20. Ortiz, A., Hilera, J.R., Otón, S., Barchino, R., Martínez, J.J., Gutiérrez, J.M., "Estandarización de los Sistemas de Búsqueda Federada: SQI como interfaz de búsqueda". X Simposio Internacional de Informática Educativa. SIIE 2008.

En este segundo artículo se presenta el sistema LORS-SQI, junto con la arquitectura que lo sustenta (LORA-SQI) en el que se ha adaptado a la especificación SQI. De esta forma se consigue una completa interoperabilidad entre sistemas. También se plantean como conclusiones la posibilidad de incorporar sistemas semánticos de búsqueda y la

composición de cursos como agregaciones de objetos de aprendizaje; objetivo de esta Tesis y que constituirá el sistema LORS-SC.

21. Hilera, J.R., Otón, S., Ortiz, A., Martínez, J.J., Gutiérrez, J.A., De Marcos, L., "Assessment of the standard query interface of public learning objects repositories". IX IEEE International Conference on Advanced Learning Technologies. ICALT 09.

En este artículo se realiza un análisis del grado de cumplimiento de la especificación SQI por parte de los principales repositorios del mercado. En él se muestran una serie de gráficas gracias a las cuales se puede observar el grado de interoperabilidad que tendrá cada repositorio con el sistema de búsqueda LORS-SQI en función del grado de cumplimiento de la especificación.

22. Otón, S., Ortiz, A., Hilera, J.R., Barchino, R., Gutiérrez, J.M., De Marcos, L., Martínez, J.J., Gutiérrez, J.A., "Requirements to ensure interoperability between learning object repositories". The 2009 International Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government. EEE 09.

En este artículo se realiza un estudio de los principales estándares y recomendaciones existentes hasta la fecha, con el fin de garantizar la interoperabilidad de un sistema de búsqueda federada como el presentado en esta Tesis. Además de este estudio, en el que se detallan las mejoras y nuevas características que tendrá el sistema tras el seguimiento del estándar o recomendación en cada caso, también se realiza un análisis en el que se muestran las partes de la arquitectura en las que se aplican, así como el seguimiento de los mismos por parte del resto de sistemas de búsqueda del mercado.

23. Otón, S., Ortiz, A., Hilera, J.R., Barchino, R., Gutiérrez, J.M., De Marcos, L., "importancia de los servicios Web como medio para garantizar la interoperabilidad entre repositorios de objetos de aprendizaje". IADIS Ibero-Americano WWW/Internet 2009.

En este artículo se realiza un estudio centrado también en mecanismos, estándares y recomendaciones para garantizar la interoperabilidad de los sistemas de búsqueda federada. En este caso, en lo que al desarrollo y uso de los servicios Web como mecanismo de comunicación e intercambio de información entre sistemas se refiere.

Sin lugar a dudas la mayor aportación de la Tesis reside en la futura publicación en la revista "International Journal of Innovative Computing, Information and Control (IJICIC)" con un índice de impacto JCR en 2008 de 2,791 (primer tercio) con el título de "Service Oriented Architecture for the Implementation of Distributed Repositories of Learning Objects" [Otón et al., 2010]. En esta publicación se resume el cometido principal de esta Tesis, presentando la arquitectura LORA-SC junto con sus principales novedades, como son la presencia de mecanismos semánticos para la realización de las búsquedas de información o la capacidad de realizar búsquedas de cursos completos de aprendizaje.

Otra aportación importante de de esta Tesis la tenemos en dos proyectos de investigación PROFIT: "FIT-350101-2004-7" denominados "Plataforma para la Gestión y Explotación de Recursos Educativos Virtuales", y "FIT-350101-2005-4" denominado "Sistema para la Publicación y Localización Universal de Objetos de Aprendizaje" del que el autor de esta Tesis forma parte del equipo investigador, coordinado por el Dr. D. José Antonio Gutiérrez de Mesa.

El primer PROFIT tenía como objetivo el desarrollo de un sistema informático que permitiera a los usuarios la utilización de un entorno virtual de aprendizaje. Éste proyecto se subdividía, a su vez, en dos subsistemas:

- Gestor del Aprendizaje: Para el control del acceso a los contenidos y gestión de la plataforma.
- Gestor de Recursos: Repositorio de contenidos docentes virtuales, además de permitir la realización de búsquedas de esos contenidos.

El segundo PROFIT tiene como objetivo fundamental el análisis y diseño de un prototipo de sistema para la construcción de cursos a través de la reutilización de contenidos docentes distribuidos en diversos repositorios. Además se ha estudiado una nueva forma de especificar los contenidos docentes independientemente de la propia

visualización y la incorporación de herramientas de comunicación a los sistemas de teleformación.

Además de estos dos proyectos de investigación el autor ha participado en otros dos en los que se pone de manifiesto la utilidad de la primera versión de la plataforma, así como su completa viabilidad. Estos proyectos de investigación son:

- "Integración de SROA (Sistema de Reutilización de Objetos de Aprendizaje) en la plataforma e-learning de una empresa", suscrito con la empresa Grupo Conforsa, Análisis, Desarrollo y Formación. Referencia 102/2007.
- "Estudio e implantación de técnicas avanzadas para la evaluación de alumnos en sistemas de enseñanza y aprendizaje electrónico", suscrito con la empresa Panel Sistemas Informáticos. Referencia 114/2007.

6.2. CONCLUSIONES

Partiendo del objetivo principal del trabajo de investigación se puede decir que lo pretendido inicialmente se ha logrado de una manera efectiva: se ha realizado una propuesta de arquitectura multicapa, demostrando la validez de la propuesta con la evolución de un prototipo que implementa un sistema concreto basado en dicha arquitectura. La arquitectura diseñada es útil y valiosa, ya que demuestra que se puede llevar a la práctica la arquitectura (proviene de un prototipo diseñado) y de esta forma permitir la completa interoperabilidad entre distintos sistemas de formación y repositorios distribuidos.

La arquitectura propuesta puede resolver los problemas de reutilización de los objetos de aprendizaje mediante su publicación y localización universal. Esto permite distribuir contenidos educativos entre distintas plataformas de e-learning y hacer interoperables los repositorios de las mismas, de una manera efectiva y eficiente.

Además de todo ello, a lo largo del tiempo en el que se ha realizado la investigación, se ha obtenido una serie de conclusiones como consecuencia directa de la misma y de los problemas que se han tenido que solventar en la construcción de la arquitectura. A continuación se muestran estas conclusiones:

- Como ha quedado reflejado en el capítulo dos, se ha llevado a cabo un análisis exhaustivo de la situación actual relacionada con los estándares y recomendaciones de teleformación. Sobre la base de estas experiencias, se mantiene el criterio de que la utilización de estándares, a ser posible “de facto” y no solamente “de jure”, ha sido siempre beneficiosa para el desarrollo y consolidación de tecnologías emergentes. Además de ayudar a la interoperabilidad entre sistemas, establecen marcos comunes de actuación que permiten aprovechar los avances de los diferentes equipos de investigación, así como la reutilización de contenidos en otros sistemas de parecidas características.

  En la medida en que seamos capaces de difundir la utilización de los estándares ya existentes y de desarrollar nuevas y útiles recomendaciones de uso y utilización de los mismos, se crearán mejores condiciones para la implantación de tecnologías que ayuden a la mejora de la enseñaza y del aprendizaje.

Como se ha comentado a lo largo del trabajo, para que un objeto de aprendizaje sea reutilizable es necesario que su construcción se haya realizado conforme a estándares o especificaciones como los establecidos por IMS, ADL (SCORM) o IEEE (LOM). De esta forma aseguramos que, a través de su empaquetamiento y descripción mediante metadatos, pueda ser integrado en cualquier plataforma de e-learning compatible con estos estándares. Sin embargo, al existir varios estándares sería recomendable el desarrollo de uno único que simplifique la interoperabilidad entre distintos sistemas de teleformación o repositorios. Como actualmente esto no ocurre y debido a la variedad de estándares y especificaciones existentes, nace pues, de manera casi natural, la necesidad de tener utilidades y herramientas que permitan la adaptación tanto de los contenidos docentes como de la metainformación que se gestiona en el proceso de aprendizaje.

Esta es una las características que se ha integrado en la arquitectura y que permite desarrollar sistemas capaces de realizar las conversiones necesarias de formato de metainformación de forma transparente al usuario.

Sería deseable que las herramientas de preparación y creación de contenidos, cuyo número es muy elevado, incorporaran utilidades o herramientas que, centrándose en aquellos estándares que se perfilen como realmente utilizados, es decir, que lo sean “de facto”, permitan, incluso también de una manera guiada, la adaptación de los contenidos a dichos estándares. Y siguiendo con esta idea, permitir que los repositorios trabajen con diversos estándares “de facto” para conseguir la integración de contenidos educativos heterogéneos.

- Siguiendo con la idea de la reutilización de los objetos de aprendizaje, además de estar construido siguiendo los estándares, debe ser localizable universalmente. De poco sirve un objeto de aprendizaje con un alto nivel de calidad si sólo es accesible por unos cuantos usuarios de una determinada plataforma o repositorio. Las instituciones educativas requieren mecanismos de interoperabilidad, ya que sería muy costoso quedar con contenido aislado en un mundo cada vez más interconectado y que clama por la colaboración institucional como mecanismo para garantizar una educación de calidad.

  Si se pretende que los objetos de aprendizaje sean reutilizables por un número potencialmente alto de usuarios, estos deben estar integrados en un sistema capaz de localizarlos y exponerlos a estos usuarios y sistemas. De

esta forma también los proveedores de contenido podrán fácilmente reutilizarlos y distribuirlos entre diferentes sistemas.

Pero para que esto ocurra no se debería obligar a que los actuales sistemas de teleformación o los repositorios existentes modifiquen su estructura y funcionamiento. Se deben proporcionar los mecanismos necesarios para que la incorporación de esta funcionalidad sea lo menos traumática posible. De ahí la utilización de los servicios Web y las arquitecturas orientadas a servicios, tecnologías que han demostrado su principal valía en la integración de aplicaciones, de forma que se añade cierta funcionalidad sin modificar lo ya desarrollado.

- En cuanto a la definición de arquitecturas de sistemas de e-learning se ha podido comprobar que la implantación de arquitecturas orientadas a servicios (como la presentada en la Tesis) será una realidad inmediata. La propia definición del Abstract Framewok de IMS lo demuestra, y ellos mismos examinan un conjunto de arquitecturas actuales para desarrollar la suya. De esta forma se consigue que en un futuro los sistemas de teleformación desarrollados siguiendo este tipo de arquitectura sean capaces de integrarse con otros sistemas de una manera muy sencilla y por lo tanto realizar sistemas interoperables. Esto se conseguirá con la integración de servicios de integración capaces de comunicarse o compartir información con otros sistemas.
- También destacar las conclusiones ofrecidas por la implementación de SQI como interfaz de consulta estandarizada. Esta implementación hace que sea posible un acceso global a la información, estandarizando el sistema de consulta. Evidentemente este es un factor fundamental en el propósito de esta Tesis, pero que sin embargo como se observó en el capítulo cuatro, dificulta mucho la obtención del contenido docente; de ahí que se proponga la utilización de un SQI2 que posibilite y estandarice lo que se cree fundamental en un sistema de búsqueda y por lo tanto de obtención de contenidos.
- Por último identificar los beneficios obtenidos por la implementación de la arquitectura planteada en esta Tesis. Este sistema permite generar y obtener un curso completo formado por diversos objetos de aprendizaje provenientes de diversos repositorios distribuidos por Internet. Para que su generación sea lo más efectiva posible, el sistema se basará en la búsqueda de los mismos en base a ontologías propias de cada temática. De esta forma se consigue un sistema completo de reutilización de contenidos docentes,

pionero en la obtención de cursos completos, y con ello, favorecer aún más la utilización de sistemas formativos e-learning.

Como se explicó en las conclusiones del capítulo tres, se quería desarrollar una arquitectura que tuviera una serie de características, a continuación se presentan estas características y las razones por las que se han conseguido integrar.

- **Abierta**. La arquitectura planteada permite la creación de sistemas e-learning interoperables y conectables entre sí de forma sencilla; es decir, que sistemas y herramientas comerciales de fabricantes distintos puedan ensamblarse en un único sistema global. Esto se ha conseguido utilizando SOA, servicios Web, y la especificación SQI para realizar las consultas.
- **Escalable**. La arquitectura está definida de tal forma que permite su crecimiento. Este crecimiento se puede ver desde dos perspectivas. Por un lado el crecimiento de datos, representado por los nuevos objetos de aprendizaje que se incorporarían al sistema al incluir un nuevo repositorio. Y por otro lado el crecimiento en funcionalidad, que se daría al incluir nuevos servicios a la arquitectura, tarea relativamente sencilla dada la naturaleza del modelo SOA aplicado en su desarrollo. Este último punto se puede ver desde una perspectiva práctica ya que la arquitectura presentada es una evolución de otra anterior, lo que aporta una visión real de la escalabilidad del sistema.
- **Global**. Permite la diversidad lingüística y cultural. El sistema resultante de la implementación de la arquitectura puede presentarse en diferentes lenguas en función del usuario a quien esté destinado. También es capaz de localizar contenidos docentes independientemente del lenguaje en el que estén realizados.
- **Integrada**. Un sistema que se desarrolle siguiendo la arquitectura propuesta será capaz de integrarse con una gran cantidad de sistemas de e-learning existentes hoy en día (y en un futuro), consiguiendo la interoperabilidad entre todos ellos. Para conseguirlo, tan solo se tiene que desarrollar un servicio Web que de acceso a sus contenidos y registrarlo en nuestro sistema.
- **Flexible**. La arquitectura presentada tiene la capacidad de implementar nuevas soluciones sin tener que efectuar grandes cambios en la misma. Como se comentó anteriormente, esto se consigue al haber realizado una arquitectura orientada a servicios en la que la inclusión de nuevos servicios no resulta una tarea traumática, sino todo lo contrario.

A modo de resumen se pueden presentar como características más destacadas de la arquitectura las siguientes:

- Presenta una arquitectura abierta y utiliza protocolos y formatos estándar para permitir la interoperabilidad e integración de plataformas de aprendizaje y repositorios.
- La posibilidad de publicar y descubrir cualquier objeto de aprendizaje independientemente de su localización y sus metadatos.
- Presenta una forma uniforme y bien definida de acceso a los objetos de aprendizaje.
- Hace que los objetos de aprendizaje sean reutilizables.
- Presenta sistemas eficientes de búsqueda basados en conceptos semánticos y que permiten la obtención no solo de objetos de aprendizaje individuales, sino de cursos completos de una temática particular.

Por último, destacar que todas estas conclusiones, conducen directamente al resultado de que la arquitectura propuesta es necesaria y deseable en el desarrollo actual de sistemas de e-learning, y plantea soluciones interesantes y viables para la interoperabilidad de sistemas heterogéneos y la reutilización de contenidos docentes.

Además, el desarrollo de la arquitectura se ha basado en estándares y especificaciones actuales que la convierten en una opción sólida y que ayuda a la consolidación de normas.

En la evolución de los sistemas de e-learning se está tendiendo hacia la idea de interoperabilidad e interconexión planteada en esta Tesis, por lo tanto es factible pensar que la arquitectura presentada puede suponer un gran avance en este sentido.

6.3. FUTURAS LÍNEAS DE INVESTIGACIÓN

Como continuación del trabajo de investigación objeto de esta Tesis, en la actualidad se han abierto varias líneas de investigación en las que se está ya trabajando. La primera y más avanzada es la integración de dispositivos móviles en la arquitectura propuesta en esta Tesis. La segunda es la integración de agentes inteligentes para la búsqueda personalizada y adaptada a las necesidades de los usuarios. La última es la incorporación de un sistema de registro basado en el DNIe, y que le dote al sistema de la capacidad de presentar contenidos adecuados al perfil, gustos y capacidades del usuario del sistema.

Los objetivos que nos se han planteado en este sentido, y una descripción de estas líneas de investigación se detallan a continuación.

6.3.1. Integración de dispositivos móviles en la arquitectura

La educación apoyada por la tecnología ha pasado por diferentes fases en los últimos años como se explicó en el capítulo dos: EAO (Enseñanza Apoyada por el Ordenador), multimedia educativo, tele-educación, enseñanza basada en Web (web-based teaching), aprendizaje electrónico (e-learning), etc. Desde el apoyo simple del ordenador y los soportes multimedia, se ha ido incorporando el uso de la red en general, y de Internet y las tecnologías Web en particular, para apoyar el proceso de enseñanza-aprendizaje en sus diferentes modalidades y aspectos. Recientemente, se incorporan a este panorama las tecnologías móviles, dando lugar a lo que se ha dado en llamar "mobile learning" o "m-learning".

El uso en educación de pequeños dispositivos móviles, tales como teléfonos móviles, agendas electrónicas, pero también tablet PCs, es un tema que está levantando gran expectación en la actualidad y sobre el que se están realizando interesantes iniciativas empresariales y proyectos de investigación. M-learning emerge para satisfacer las necesidades individuales de diferentes usuarios, permitiéndoles el acceso a información específica desde cualquier lugar en cualquier momento y en cualquier dispositivo [Sharma y Kitchens, 2004].

Por lo tanto, surge la necesidad de que el proceso educativo se pueda adaptar a diversos tipos de dispositivos más o menos flexibles y potentes. Un nuevo reto que surge es la adaptación de las interfaces Web que se utilizan en las aplicaciones de teleformación a estas condiciones, cambiantes, flexibles y de máxima disponibilidad.

En el desarrollo de esta línea de investigación se está también teniendo en consideración el fuerte proceso de estandarización por parte de organismos internacionales que los sistemas de teleformación están sufriendo. Estos procesos están dando como resultado una identificación de necesidades y características deseables y una aportación de soluciones de interacción entre sistemas LMS y entre estos y terceros sistemas.

Una de las principales características que tiene la arquitectura presentada, es que está dotada de una interfaz Web y, por lo tanto, es completamente accesible desde cualquier ubicación. En el capítulo cuatro se mostraba el resultado de la interfaz de

entrada al sistema de los dos sistemas previos al desarrollado en esta Tesis, así como el resultado de la búsqueda sobre un equipo convencional. Este sistema es perfectamente accesible no sólo desde un equipo de estas características, sino desde cualquier sistema dotado de un navegador Web, como se puede apreciar en la Figura 6.1, en la que se muestra un ejemplo de acceso al sistema desde una PDA dotada con conectividad WIFI y con navegador Web. De esta manera, la reutilización de objetos de aprendizaje se ve claramente reforzada, ya que permitiremos acceder a información docente desde este tipo de terminales, cada vez más utilizados, mejorando con ello la ubicuidad y accesibilidad del sistema.

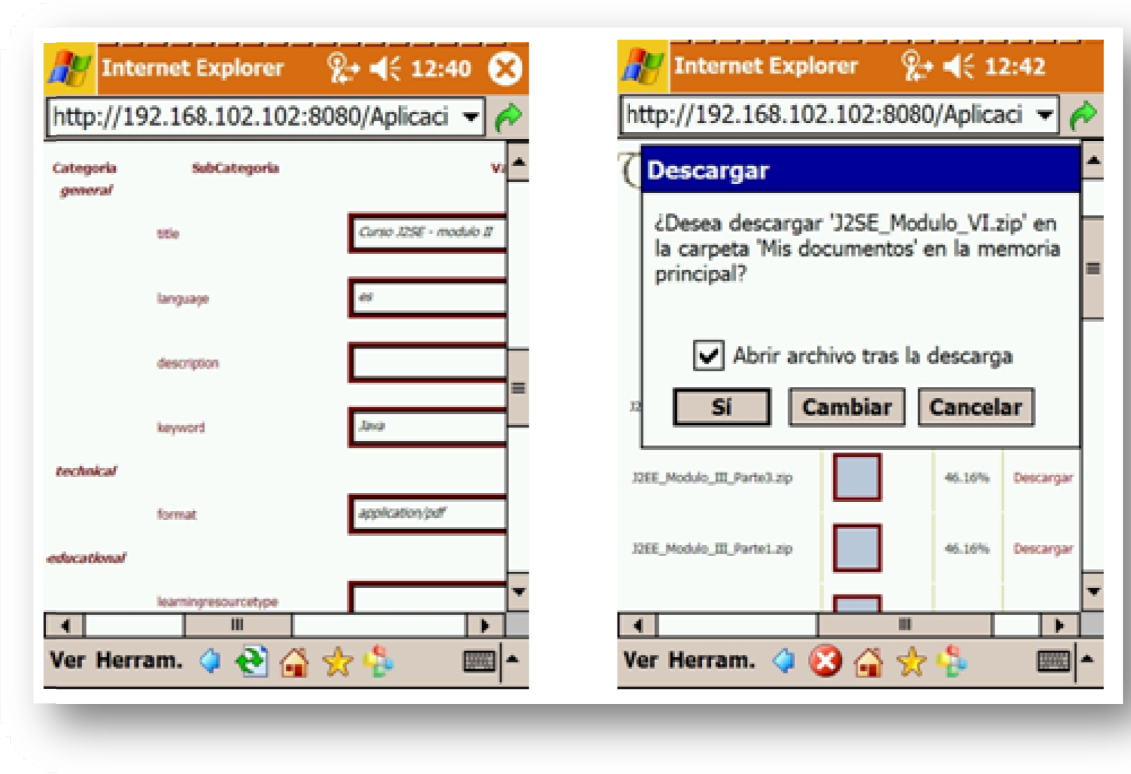


Figura 6.1 Interfaz de entrada al sistema y resultado de una búsqueda sobre un terminal móvil.

Otro aspecto destacable, es que uno de los metadatos que se utilizan en el sistema por defecto, es el del formato del contenido educativo del objeto de aprendizaje, como es *format*, dentro de la categoría *technical*. A través de este campo, el usuario podrá buscar solamente aquellos objetos educativos con un formato determinado, para que así estén adaptados a las posibilidades computacionales del terminal sobre el que se vaya a visionar el contenido educativo, mejorando sensiblemente el filtrado sobre este tipo de material. Pero no solamente será necesario centrarse en este campo educativo de LOM previamente seleccionado para el sistema, ya que, según las características de

adaptabilidad de la arquitectura presentada, será posible incorporar campos como el tamaño del archivo, requisitos técnicos, tipo de interactividad o coste, también presente en la especificación LOM, para limitar así, determinados objetos de aprendizaje demasiado grandes para el terminal, o para la tasa de conexión en un momento determinado. También serán interesantes para indicar el tipo de recursos con los que contamos en un momento determinado y así evitar buscar objetos de los que no vayamos a poder disfrutar con el terminal.

La arquitectura presentada permite distribuir contenidos educativos entre distintas plataformas de e-learning y distintos tipos de clientes. Queda patente que una descripción de la metainformación de un objeto de aprendizaje es muy importante a la hora de permitir su búsqueda y por lo tanto su reutilización. También es importante señalar que mediante esta metainformación es posible determinar si un objeto de aprendizaje es válido para ser accedido desde distintos tipos de dispositivos.

Se sigue trabajando para que la arquitectura y los sistemas desarrollados permitan una mejor adaptación a este tipo de dispositivos incorporando mecanismos de presentación y filtrado de contenidos para su adaptación a sus limitaciones propias.

6.3.2. Integración de agentes software en la arquitectura

La tecnología de agentes es un área de desarrollo relativamente joven dentro de las Tecnologías de la Información que está sufriendo un rápido crecimiento. No es una tecnología totalmente nueva, sino más bien la aplicación integrada de múltiples tecnologías, tales como la Inteligencia Artificial, los Sistemas Distribuidos Orientados a Objetos, y la interacción persona-ordenador.

Para poder realizar un análisis adecuado del estado de la tecnología de agentes, es necesario partir de conceptos como agente o sistema multiagente, lo que hace conveniente definirlos adecuadamente [Brenner et al., 1998]. Como suele suceder en las disciplinas aún en estado emergente, no existe una definición universalmente aceptada del concepto de agente software. El diccionario nos proporciona como definición de agente "cualquier entidad que actúa" [RAE, 2009]. Evidentemente, esa definición es demasiado genérica para apoyar en ella el desarrollo de sistemas software.

Algunas de las diversas definiciones del concepto de agente que se encuentran en la bibliografía especializada se ven influidas por la aplicación específica sobre la que se realiza el desarrollo, si bien existen ciertos grupos de investigación que son partidarios de definiciones más generales y universales [FIPA, 2009]. Desde el punto de vista de las implicaciones tecnológicas y de diseño, se podría definir un agente software como un programa autocontenido capaz de controlar su propia toma de decisiones y de actuar, basándose en la percepción de su entorno, para la consecución de uno o más objetivos [Franklin y Graesser, 1996]. Atendiendo más a la perspectiva funcional del usuario, puede definirse como una entidad software en la que se pueden delegar tareas. Esta última definición, aunque más simple, ilustra con claridad el propósito último de los agentes software: la automatización de determinadas tareas comúnmente llevadas a cabo por los usuarios.

Características de los agentes

A continuación se detallan algunas de las características básicas que pueden presentar los diferentes agentes de un sistema [Milojicic et al., 1998] [Lange y Oshima, 1998]:

- **Autonomía**: La autonomía es la capacidad de actuar sin la intervención directa de una persona o de otro agente. Hasta cierto punto, los agentes

pueden funcionar sin una intervención externa directa. Un agente debe poder controlar sus propias acciones y estado interno. Una vez que el usuario activa el agente indicando algún objetivo de alto nivel, éste actúa independientemente, seleccionando estrategias y monitorizando el progreso en busca de la meta. Si falla con una estrategia, usará otra, pero sin intervención humana o con la mínima indispensable.

- **Reactividad:** Se refiere al hecho de que un agente debe poder interpretar el estado del ambiente dentro del cual se encuentra inmerso y, en función de esto, actuar, respondiendo de manera adecuada a cambios producidos en el mismo. Los efectos producidos pueden modificar el estado de su entorno.
- **Interactividad:** La interactividad se define como la capacidad de comunicarse tanto con el entorno como con otras entidades o agentes. La forma más básica de comunicación en sistemas software es el paso de mensajes entre agentes (también llamada invocación de métodos). Otra forma de interacción más compleja es la percepción de cambios en el entorno del agente. Resulta evidente que para que un agente resulte útil debe ser capaz de interaccionar de algún modo con su entorno o con otras entidades, por lo que la interactividad es una propiedad esencial de un agente software.
- **Adaptabilidad:** La adaptabilidad es la capacidad de un agente de responder a otros agentes o a su entorno en cierta medida. Un nivel mínimo de esta propiedad se encuentra en agentes que son capaces de reaccionar ante estímulos simples, proporcionando una respuesta determinista ante los mismos. Un nivel más alto de adaptabilidad supone que el agente sea capaz de razonar, es decir, de realizar inferencias a partir de los datos recibidos. Las formas más avanzadas de adaptabilidad implican agentes que sean capaces de modificar su comportamiento en base a la experiencia adquirida, y de este modo aprender y evolucionar.

  Al igual que sucedía con la interactividad, un agente que no sea capaz de responder ante su entorno o ante otras entidades carece de utilidad, por lo que la adaptabilidad es esencial para el concepto de agente. Un ejemplo de esta característica sería un agente de reconocimiento del habla que reconociera los patrones que posee en diferentes entornos, ruidos y personas.

- **Movilidad:** La movilidad es la capacidad de un agente para transportarse a sí mismo desde un entorno de ejecución a otro. La ventaja principal de la movilidad para ciertas aplicaciones es que permite al agente trasladarse directamente a la máquina donde se encuentran los servicios que necesita, con lo que se incrementa el rendimiento del intercambio de datos entre el

agente y las entidades que le proporcionan los servicios. Las principales desventajas de la movilidad de agentes son las consideraciones de seguridad que genera y la posibilidad de cargar excesivamente la máquina que ofrece un determinado servicio con la ejecución del código de los agentes cliente, con los problemas de escalabilidad que eso puede ocasionar.

La movilidad no es una propiedad indispensable para un agente, sino que modifica la forma por la cual el agente cumple con sus objetivos, en este caso recurriendo a los recursos que puede ofrecer una red de computadores. Aporta una nueva forma de computación distribuida.

- **Inteligencia:** No existe un acuerdo claro en lo que se refiere a la definición del término inteligencia aplicada a los agentes software. Una aproximación prudente es entender como inteligencia de los agentes software cierto grado de proactividad, es decir, que los agentes muestren cierto nivel de conocimiento y sean capaces de dar unas respuestas correctas en base a ese conocimiento [Plekhanova, 2002].

  La dimensión de inteligencia representa el grado en el cual el agente utiliza el razonamiento, aprendizaje y otras técnicas para interpretar la información o conocimiento al cual tiene acceso.

  Esta dimensión tiene diferentes aproximaciones desde una inteligencia marginal a otra muy avanzada. Las formas más modestas de inteligencia permiten al usuario expresar sus preferencias. En un nivel superior se tiene una formalización de un conjunto de reglas de razonamiento que combinadas con conocimiento a corto y largo plazo, en un proceso de inferencia, puede conducir a la forma de alguna acción. El tercer y último paso es la capacidad del agente de modificar su capacidad de razonamiento en la base de un nuevo conocimiento derivado de un amplio rango de fuentes, esto es, aprender.
- **Proactividad:** Un agente busca satisfacer cierto estado interno o lograr un objetivo determinado, con mínima intervención humana. Por ejemplo, un agente recuperador de datos tiene especificada una tarea. El agente deberá intentar permanentemente satisfacer la tarea que le fue delegada en base a las estrategias de búsqueda y recuperación con las cuales fue construido, hasta cumplir con el objetivo. Mientras tanto, el agente debe mostrar un comportamiento activo, que lo lleve a tomar la iniciativa de la acción a tomar cuando lo considere apropiado.

- **Sociabilidad:** Un agente debe ser comunicativo y "sociable". Debe tener habilidad para interactuar con otros agentes o incluso con alguna persona o usuario, para solicitar información o bien para exponer los resultados obtenidos de la ejecución de las tareas aprendidas. La naturaleza de la comunicación dependerá del tipo de agente con quién se comunique, estableciéndose un protocolo común de intercambio de información entre ambas partes. Los agentes deben poseer algún tipo de interfaz para comunicarse con sus usuarios.
- **Continuidad temporal:** Un agente es un proceso temporalmente continuo. A diferencia de un programa convencional del cual se conoce su inicio y fin, un agente debe ejecutarse hasta que se haya alcanzado el conjunto de objetivos solicitados, o bien, mientras su ciclo perdure y su usuario no desee detenerlo. La continuidad temporal es la propiedad que da "vida" al agente, posibilitando que se mantenga alerta a una solicitud o a algún cambio en el medio. El ciclo de vida de un agente depende de sus características, de las tareas que realice y de los deseos de su usuario en cuanto al tiempo durante el cual el agente debe ejecutarse.

Sistemas multiagente

Existen aplicaciones basadas en agentes en las que no se produce prácticamente ninguna interacción entre diferentes agentes (agentes software aislados). Por ejemplo, los servicios de búsqueda a través de Internet se basan en el lanzamiento de múltiples agentes autónomos que recorren las páginas Web y seleccionan las secciones más relevantes de la información que encuentran, la clasifican y la almacenan en una base de datos que posteriormente sirve como índice para devolver al usuario la información cuando realiza una consulta. Sin embargo, se pueden crear sociedades de agentes que se coordinen para llevar a cabo diferentes tareas. Estas sociedades son lo que se conoce como sistemas multiagente [Wooldridge, 2002].

Un sistema multiagente es, pues, un sistema compuesto por agentes que se coordinan a través de las relaciones entre ellos. Estas relaciones pueden ser de cooperación, de competencia, o ser una mezcla de ambas. Algunas de las ventajas de los sistemas multiagente son las siguientes:

- Permiten dividir la funcionalidad necesaria para desempeñar tareas complejas entre distintos tipos de agentes, lo que proporciona una mayor

modularidad, flexibilidad y extensibilidad, además de presentar un mejor rendimiento que la opción de un único agente que desempeñe por sí solo la tarea completa.

- Permiten repartir también la información manteniéndola accesible, eliminando la necesidad de repositorios de datos centralizados. El agente que necesite hacer uso de la información que posee otro agente sólo necesita pedírsela.
- Facilitan el desarrollo de aplicaciones de computación distribuida.

Los sistemas multiagente requieren la existencia de un entorno adecuado para el funcionamiento de las entidades que lo forman. En general, dicho entorno constituye lo que se denomina una plataforma multiagente, como por ejemplo JADE [2009], plataforma multiagente desarrollada en Java, que proporciona la infraestructura necesaria para posibilitar la ejecución simultánea controlada de varios agentes y la comunicación fiable y organizada entre los mismos.

Integración de agentes en la arquitectura

Después de ver una introducción de los conceptos más relevantes sobre la tecnología de agentes, a continuación se indica cómo se pueden integrar en la arquitectura propuesta en la Tesis, así como en los sistemas basados en ella, para añadir ciertas mejoras.

La búsqueda federada es la funcionalidad más destacable de la arquitectura, pero como se ha comentado esa búsqueda se produce en un momento dado, con unos parámetros de búsqueda y sobre una serie de repositorios. Sin embargo, puede existir la necesidad en el tiempo de conseguir contenidos docentes de unas determinadas características. De esta forma, el usuario puede indicarle al sistema que conserve sus parámetros de búsqueda y un histórico con los resultados obtenidos, de forma que cuando se encuentren contenidos nuevos que coincidan con los parámetros de búsqueda los recupere y avise al usuario; por ejemplo, mediante un correo electrónico. De esta forma, si se añaden nuevos repositorios al sistema o se añaden nuevos contenidos que interesen al usuario, éste recibirá un aviso indicándoselo.

Para realizar esta tarea es muy adecuado el uso de agentes software, ya que como se ha visto en su descripción tendrían un objetivo a cumplir, que sería el de cubrir las necesidades de localización de contenidos docentes de unas determinadas características. Se puede indicar que el agente utilizaría, de forma proactiva, los datos de búsqueda para automatizar y optimizar la localización de contenidos docentes requeridos por el usuario, de un modo más rápido y eficiente que si el usuario realizara esta labor de forma continuada.

Habría dos formas de desarrollar esta nueva funcionalidad:

- Realizar agentes aislados en el sistema, encargados de tratar los distintos tipos de búsqueda de manera que consultara de forma periódica cada uno de los repositorios y mantuviera un histórico de los resultados.
- Realizar un sistema multiagente, de forma que en cada uno de los repositorios hubiera una serie de agentes encargados de comunicar a los agentes del sistema posibles cambios en los contenidos.

Sobre esta funcionalidad se está trabajando actualmente para dar un nuevo valor añadido a la arquitectura y a los prototipos desarrollados.

6.3.3. Integración del DNIe como sistema de validación en la arquitectura

Como declara el Art. 6 Declaración Universal de Derechos Humanos, "Todo ser humano tiene derecho, en todas partes, al reconocimiento de su personalidad jurídica". Por tanto la identidad personal es un derecho de todo ciudadano y los Estados tienen la obligación de establecer los mecanismos adecuados para facilitársela.

Según la definición de la Real Academia Española de la Lengua [RAE, 2009], "La identidad es el conjunto de rasgos propios de un individuo o de una colectividad que los caracterizan frente a los demás", y también "Conciencia que una persona tiene de ser ella misma y distinta a las demás".

Como se puede apreciar, la identidad personal es un concepto importante que toma aún más valor en la actual Sociedad de la Información. De esta forma se entiende la necesidad de establecer los medios y mecanismos más adecuados para que el Estado otorgue esta identidad personal a sus ciudadanos.

Otorgar identidad personal a los ciudadanos adquiere una nueva dimensión cuando se trata de establecerla para un uso no presencial en medios telemáticos. Y aunque la identidad siempre es física, es necesario establecer mecanismos y procedimientos electrónicos para verificarla en estos nuevos ámbitos.

El nacimiento del Documento Nacional de Identidad electrónico (DNIe) viene a cubrir la necesidad de otorgar identidad personal a los ciudadanos para su uso en la nueva Sociedad de la Información, además de servir de dinamizador de la misma.

Concepto de Autentificación

Antes de iniciar el proceso de exposición de esta futura vía de investigación del sistema de autentificación de usuarios seleccionado, será necesario que se identifiquen claramente algunos conceptos que se tratarán durante el desarrollo de este apartado.

En primer lugar es necesario que se comprenda cuál es el significado del término *autentificación*. Se llamará autentificación a la comprobación de la identidad de una persona o de un objeto. Por lo tanto, se puede decir que un sistema de autentificación es aquel sistema que permite comprobar la identidad del usuario, verificando en todo momento que el usuario es quien dice ser. A grandes rasgos, se pueden destacar tres sistemas de identificación de usuarios: mediante contraseña, mediante dispositivo y mediante dispositivo biométrico.

- La autentificación mediante contraseña es el sistema más común. Este sistema permite comprobar la identidad del usuario a través de la introducción de una serie de caracteres secretos que sólo dicho usuario debería conocer. Este sistema es empleado en casi cualquier entorno. Algunos ejemplos pueden ser los sistemas operativos, servidores Web o de correo.
- La autentificación mediante dispositivo es un sistema de autentificación que reconocerá al usuario a través de una llave (normalmente una tarjeta con chip). Es decir, este tipo de sistemas identifican al usuario a través de un elemento externo que introducen para que el sistema lo lea e identifique al usuario.
- Los dispositivos biométricos son un caso especial del anterior, en los que la llave es una parte del cuerpo del usuario, huella dactilar, voz, pupila o iris. Existen ya en el mercado, a precios relativamente económicos, bastantes dispositivos que llevan incorporado un lector de huellas dactilares, como por ejemplo ratones, teclados, o terminales móviles.

También cabe destacar que es bastante común el uso de varios de estos sistemas combinados, como por ejemplo sistemas que además de introducir cualquier tipo de llave, es necesaria la introducción de una contraseña, de forma que el sistema de autentificación se vuelve más seguro.

El DNI Electrónico (DNIe)

Después de varios años de preparación, el Consejo de Ministros aprobó en Febrero de 2004 la creación y distribución de un nuevo Documento Nacional de Identidad Electrónico (DNIe) entre los ciudadanos españoles.

La Dirección General de la Policía adjudicó la definición y el desarrollo de este documento a la Unión Temporal de Empresas (UTE) formada por Telefónica, Indra y Software AG, las cuales se presentaron con dos tecnologías de infraestructuras de clave pública: la española Safelayer y la americana Entrust (distribuida por la multinacional española Grupo SIA).

Se prevé que los costes de la implementación para un período de cuatro años sean de unos 100 millones de euros, por lo que se trata, sin duda, de uno de los proyectos más importantes de la e-Administración en España. El Comité de Coordinación y Comisión Técnica será el responsable de la campaña de comunicación, el soporte técnico para los ciudadanos y la selección del primer rango de servicios basados en el DNIe. La fase inicial de nueve meses incluye el desarrollo de un sistema para crear y gestionar los DNIe. El consorcio tendrá que buscar y aportar todo el material y la tecnología necesaria para la implementación exitosa, incluyendo entre otras cosas una infraestructura de clave pública (PKI), la provisión de captura biométrica, las soluciones de impresión de tarjetas, etc.

Para Febrero de 2006 el consorcio tenía que seleccionar un centro piloto para la emisión y personalización de los DNIe que permitiera la validación del proceso de cara al desarrollo del sistema a nivel nacional en 2007-2008. Cada uno de los 350 centros tendrá que emitir y entregar los DNIe inmediatamente a petición de los ciudadanos. Además, cada centro estará conectado a la base de datos central que almacenará todos los datos capturados y autorizará la emisión de cada DNIe.

En la actualidad hay más de 29 millones de documentos nacionales de identidad en España y aproximadamente se realizan unos 6 millones de renovaciones anualmente. Las estimaciones oficiales indican que la mayor parte de los ciudadanos españoles tendrán un DNIe a finales del año 2010.

La implantación del DNI electrónico en España tiene como objetivo proporcionar nuevas utilidades a los ciudadanos y supondrá un salto indiscutible en el desarrollo de las nuevas tecnologías y la sociedad de la información. El reto del sector público y de las empresas privadas es poner las tecnologías de la información al servicio del ciudadano, acelerar y dinamizar las comunicaciones e incrementar su eficacia.

El DNI electrónico sirve al igual que el DNI tradicional para identificar a las personas físicas, pero con el añadido que además permite acreditarlas electrónicamente y firmar en su nombre, permitiendo así realizar numerosos trámites telemáticos con mayor comodidad y a través de Internet.

La validez jurídica de esta firma electrónica será equivalente a la de la firma manuscrita. En palabras de Vincent Bernard, Director General de Oberthur Card Systems Ibérica: "La nueva tarjeta o DNIe nace con vocación de herramienta segura de identificación en el entorno tanto físico como digital, y supondrá el acceso a las nuevas tecnologías por todos y cada uno de los ciudadanos españoles".

Este nuevo documento electrónico de identidad es similar en su apariencia física al DNI actual, puesto que tiene el mismo tamaño y color, aunque incorpora un chip en el que se recoge la información de su usuario. La tecnología utilizada para el nuevo DNIe se basa en la utilización de las denominadas "tarjetas inteligentes", ya que contiene claves privadas y sus correspondientes certificados de claves públicas como se verá en los apartados posteriores.

Una tarjeta inteligente o tarjeta con chip es una tarjeta plástica, normalmente fabricada con PVC, que cumple con estándares ISO y que contiene un chip de silicio incrustado en el cuerpo de plástico.

A continuación se destacan las principales características que presenta este novedoso DNIe:

- El DNI electrónico integra, por una parte, el soporte físico (la tarjeta) que es de policarbonato, un material de alta fiabilidad que tiene una duración estimada superior a los diez años, y que no puede dividirse en láminas sin provocar su destrucción. La fabricación de la tarjeta se realiza con una tinta ópticamente variable, un hilo de seguridad embebido en el papel, relieves en el plástico y una fotografía impresa que está protegida con fondos de seguridad.
- El DNI también consta del soporte electrónico que incluye los datos de identificación del titular, como la huella digital y la imagen facial, e incorpora la firma electrónica. Se han utilizado además métodos criptográficos y

códigos de barras bidimensionales que incrementan la seguridad del nuevo DNIe.

- Una de las partes de este DNIe será privada e incluye el certificado que garantiza la identidad del ciudadano. Le dará capacidad para firmar con una clave criptográfica personal (un PIN) que sólo él conocerá. A esta parte privada sólo podrá acceder el ciudadano cuando desee realizar por ejemplo operaciones telemáticas. En este caso, el portador sólo necesitará disponer de un lector de tarjetas, un software específico que podrá descargar de Internet e introducir el código PIN o su huella dactilar. La parte pública del DNIe es en la que aparecen los datos que estarán impresos en la tarjeta (por ejemplo: fecha de nacimiento, lugar de residencia, etc.).
- Para identificar al usuario, el chip del DNIe integra un criptoprocesador, la firma, la huella y la foto digital. Como explica Vincent Bernard, la seguridad y la protección de los datos es absoluta: “Mediante este sistema los datos nunca podrán salir de la tarjeta, lo que representa una medida de seguridad para proteger los datos de los ciudadanos”.

La tecnología de las tarjetas inteligentes aplicadas a los documentos de identificación nacional e internacional no es una novedad. Esta iniciativa también se ha producido en otros países europeos como por ejemplo Bélgica, Dinamarca, Finlandia, Irlanda, Italia, Luxemburgo, Noruega o Suecia. Por ejemplo, en Bélgica, la francesa Oberthur Card Systems ha sido elegida para aportar la tecnología necesaria y su conocimiento en tecnología de tarjetas de alta seguridad a la fabricación de pasaportes electrónicos. Otros países europeos ya llevan tiempo funcionando con este nuevo sistema. En concreto, Finlandia fue el primer estado en incorporar el documento electrónico: fue en 1999.

En la medida que el DNI electrónico vaya sustituyendo al DNI tradicional y se implanten las nuevas aplicaciones, se podrá utilizar entre otras cosas para:

- Realizar compras firmadas a través de Internet.
- Hacer trámites completos con las Administraciones Públicas a cualquier hora y sin tener que desplazarse ni hacer colas.
- Realizar transacciones seguras con entidades bancarias.
- Acceder a edificios.
- Utilizar de forma segura cualquier equipo que cuente con un lector de tarjetas.

- Participar en una conversación por Internet con la certeza de que nuestro interlocutor es quien dice ser.

El DNI electrónico es una oportunidad para acelerar la implantación de la Sociedad de la Información en España y situarnos entre los países más avanzados del mundo en la utilización de las tecnologías de la información y de las comunicaciones, lo que, sin duda, redundará en beneficio de todos los ciudadanos.

Integración del DNIe en la arquitectura

El propósito de integración es que el DNIe sirva como mecanismo de validación en el sistema. A priori podría parecer que esta medida carece de importancia para un sistema de búsqueda, ya que no se necesita saber qué usuario realiza qué búsqueda. Sin embargo si el sistema de búsqueda federada conociera la identidad del usuario, y con ello sus gustos, especialidades, conocimientos, etc. podría mostrarles contenidos interesantes para su perfil, como ya se comentó en la línea de investigación futura anteriormente presentada. Por lo tanto en esta nueva vía de investigación se plantean dos cosas: permitir el registro de usuarios por un lado (con su correspondiente validación a través del DNIe), y la posibilidad de mostrarle contenidos adaptados a las necesidades del usuario tanto en el momento de realizar la búsqueda, como en momentos posteriores, como también se comentó en el anterior apartado.

Evidentemente, para que el sistema de búsqueda pueda notificarle al usuario la aparición de nuevo material docente que se ajuste a la solicitud de una búsqueda pasada, es necesario que el sistema sepa, por un lado, qué usuario está realizando cada búsqueda, y por otro, cómo avisarle. Para que esta tarea sea posible, es imprescindible contar con un sistema de registro.

Existen en el mercado bastantes sistemas de registro e identificación de usuarios, como ya se ha visto anteriormente, sin embargo, uno lleva registrando a la población durante más de 50 años, factor que garantiza su fiabilidad y aceptación popular; se trata del DNI. Desde hace relativamente poco tiempo, ha sufrido una importante remodelación. Ha pasado de ser un simple sistema de identificación tradicional, a ser un sistema de identificación también vía telemática; pero no sólo esto, sino que además se

puede utilizar para cifrar información, autenticarse frente a un sistema telemático, establecer canales seguros de comunicación, firmar electrónicamente documentos con la misma validez legal que la firma manuscrita tradicional, garantizar el no repudio de la información, etc. Todas estas funcionalidades han hecho que sea elegido como sistema de identificación y autenticación de la actual vía de investigación futura.

De esta forma, el sistema contará con un sistema de verificación de usuarios basado en el DNIe. Cada usuario que quiera conectarse al sistema deberá contar un lector de tarjetas inteligentes, e introducir su DNIe. El sistema leerá de este, el número de DNI (único como ya se sabe para cada usuario), y de esta manera identificarlo de forma inequívoca. Dadas las capacidades de este DNIe, sería posible crear un sistema seguro de comunicación entre usuario y sistema, de forma que toda la información que envíe el usuario, vaya cifrada con su clave privada. Así, el sistema asegura que, es dicho usuario y nada más que dicho usuario el que está interactuando con el sistema. Para que esto sea posible, es imprescindible que el usuario introdujera además de su DNIe, la clave PIN asociada al mismo. Los motivos por los que se piensa que esta característica no es necesaria para el sistema, es porque la información manejada por el mismo no es tan relevante como para contar con estas medidas de seguridad. Sin embargo, es posible dejar la "puerta abierta" a nuevos tipos de información gestionada, de forma que en un futuro sería posible contar con un sistema tan seguro como el comentado sin realizar un esfuerzo considerable. Esta es otra de las ventajas de contar con un sistema tan eficiente como el DNIe, desarrollado basándose en los principales estándares del sector, y por personal y empresas altamente cualificadas.

El proceso que seguiría un usuario para acceder al sistema de búsqueda federada quedaría reflejado por la Figura 6.2.

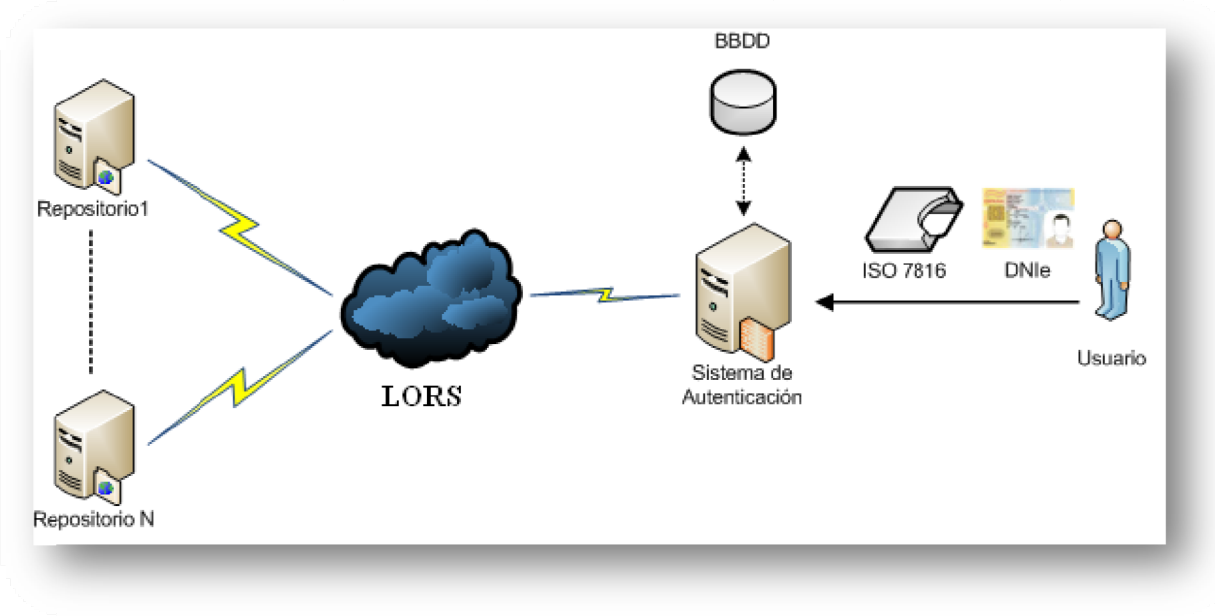


Figura 6.2 Sistema de autenticación de LORS

Como se puede ver en la Figura 6.2, cualquier usuario provisto del DNIe, lo introducirá en un lector de tarjetas compatible con el estándar ISO 7816. Tras este proceso, accederá a la interfaz Web del sistema LORS, que verificará a través de su sistema de autenticación que se trata de un DNIe correcto. Tras esta verificación, leerá del mismo, el número de identificación, y comprobará en su base de datos que se trata de un usuario registrado en el sistema.

En el caso que esta comprobación sea correcta, el sistema le dará acceso al sistema de búsqueda federada, permitiéndole buscar y acceder a los distintos objetos de aprendizaje que se encuentren en los diferentes repositorios distribuidos por Internet. En caso contrario, se instará al usuario a registrarse en el mismo, para que así pueda acceder al contenido docente.

Cuando el registro se haya completado satisfactoriamente, quedará almacenada en la base de datos una nueva ficha de usuario sobre la que informarse cuando aparezcan nuevos objetos docentes, siempre y cuando este usuario así lo haya notificado durante su registro, o posteriormente. Evidentemente, este sistema no podía recurrir al DNIe como único mecanismo de registro, ya que actualmente la mayor parte de la población todavía no cuenta con uno, o bien, es posible que en el momento de querer realizar una búsqueda, ese usuario no disponga de su DNIe, o de un lector de tarjetas inteligentes. Posibilitando estas dos opciones de registro e identificación, aumentan

considerablemente las posibilidades para los usuarios, de forma que en todo momento se podrá acceder al sistema y, usando su mecanismo de búsqueda federada, acceder a todo el contenido docente que se encuentra distribuido por los diferentes repositorios de Internet.

Una vez que el usuario se identifique frente al sistema, este creará un registro en el que se almacenarán todas las peticiones que el usuario indique. Siempre que se añada un nuevo repositorio al sistema, o dicho repositorio informe al sistema de la incorporación o modificación de los objetos docentes con los que cuenta, se verificarán los listados de registros de búsquedas de los diferentes usuarios. Como esta tarea podría resultar muy costosa (en cuanto a lo que tiempo de procesado se refiere), se deberían crear tipologías de búsquedas latentes, similares a las comentadas anteriormente para los objetos docentes. De esta forma, se irá accediendo, en forma de árbol de búsqueda, a las diferentes tipologías de búsquedas realizadas, y se irán despreciando el resto, disminuyendo con ello el tiempo requerido para esta tarea. A continuación se muestra en la Figura 6.3, un ejemplo clarificador del sistema de búsqueda, en el que evidentemente se contaría con elementos ontológicos que mejoraran su eficiencia.

En esta imagen se simularía el funcionamiento del proceso seguido por el sistema para incorporar un nuevo usuario a un listado de búsquedas pasadas. En este ejemplo se presupone que un usuario (número cuatro), ha realizado búsquedas de objetos de aprendizaje del lenguaje de programación Java. Para registrar este tipo de solicitud, el sistema irá buscando por las diferentes categorías que incluyen a Java, hasta dar con el último nodo. Según el ejemplo, pasaría por Informática ➔ Programación ➔ Java. Una vez localizado el último nodo, simplemente se añadiría un nuevo enlace dinámico al nuevo usuario.

Dependiendo de la categoría que se esté tratando en cada momento, se podrán contar con diferentes tipos de nodos, que se crearán y se destruirán de forma dinámica.

Tanto la figura como el sistema de acceso a los usuarios serían similares para el proceso de búsqueda de usuarios una vez que se detecten cambios en los repositorios. Si un repositorio informara de la modificación de ciertos contenidos de Java, el sistema iría buscando por la tipología que contiene a Java. Al igual que se ha comentado anteriormente, comenzaría por la categoría de Informática, de ahí pasaría a la de

Programación, hasta llegar a Java. De este enlace se obtendrían todos los usuarios que se dieron de alta para recibir modificaciones o apariciones de objetos docentes de Java, ya que en el pasado realizaron algún tipo de búsqueda sobre estos objetos. A partir de este momento, el sistema simplemente obtendría de su registro, el medio por el que desea el usuario que sea avisado, vía SMS o e-mail, datos que serán rellenados por el usuario cuando se registre por primera vez en el sistema.

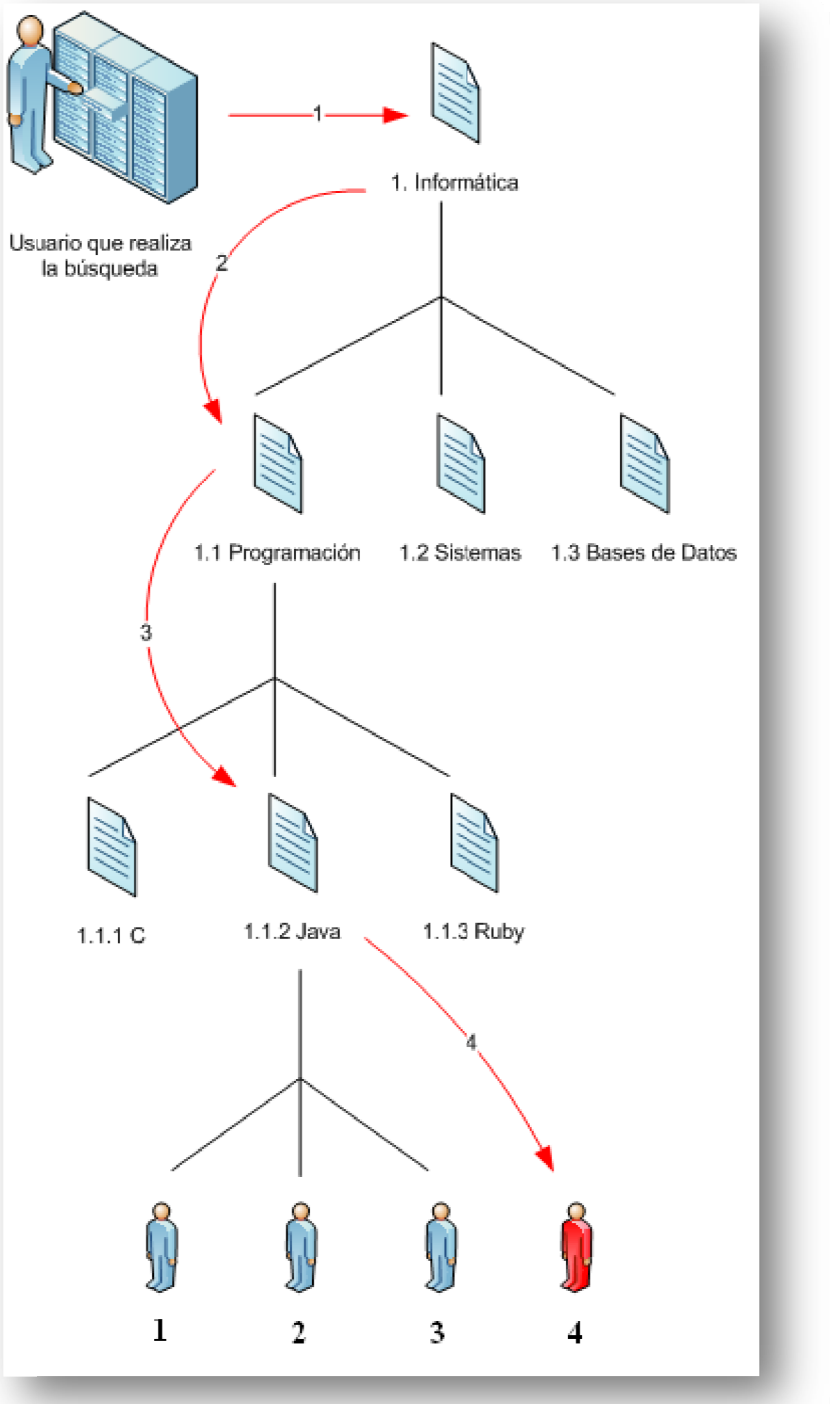


Figura 6.3 Sistema de registro de usuarios que han solicitado búsquedas latentes

Según el proceso que se acaba de comentar, se muestra en la Figura 6.4 cómo es todo el proceso completo de notificación por parte del sistema LORS.

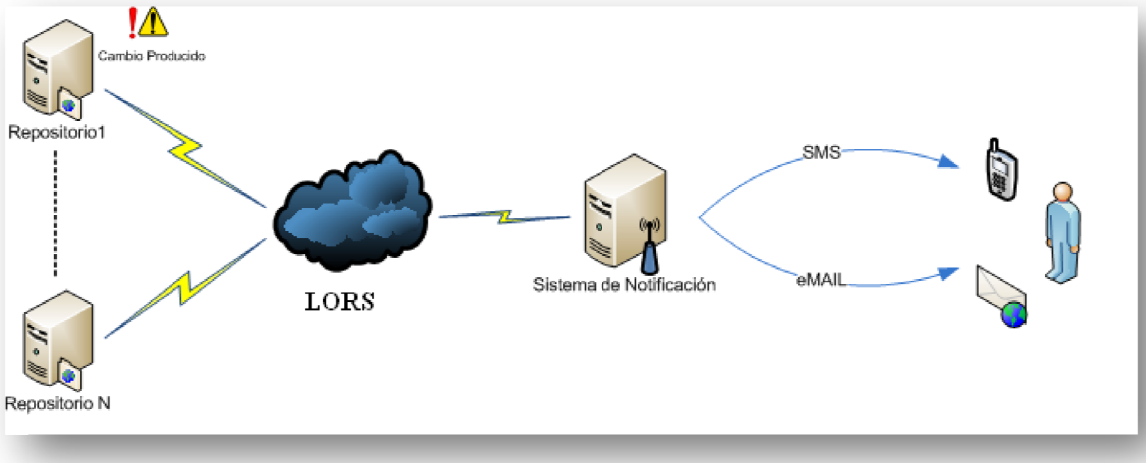


Figura 6.4 Sistema de notificación de búsquedas pasadas

Como muestra la Figura 6.4, cuando uno de los repositorios informe que se ha producido un cambio, el sistema LORS contactará con su servicio de notificación, el cual tras obtener del registro del usuario el medio de notificación que especificó en su ficha, le enviará, o bien un e-mail o un SMS en el que se le indicará la aparición de nuevos objetos docentes que se ajustan a los parámetros establecidos durante una búsqueda pasada. En cada uno de estos formatos, se le indicará al usuario que si desea dejar de recibir esta información, solamente tiene que modificar su ficha, a la que tendrá acceso una vez que se identifique frente al sistema.

Una vez comentado todo el proceso de registro y notificación de usuarios, se desea hacer hincapié en una de las posibles características con las que puede contar este sistema, y que fue comentada anteriormente. Si un usuario se autentica frente a LORS utilizando su DNIe, no solamente sabrá el sistema que se trata inequívocamente de dicho usuario, sino que puede utilizar este documento para establecer un canal de comunicación seguro entre sistema y usuario, y así garantizar la integridad de toda la información que se maneje.

En el siguiente supuesto, cuando un usuario se identifique frente al sistema, además de ser necesaria la introducción de su DNIe, será necesaria la introducción de su clave PIN. Cuando el sistema compruebe que toda esta información es correcta, podrá entonces obtener la clave pública que se encuentra dentro del DNIe.

El sistema enviará toda la información cifrada con la clave pública del usuario, siendo este el único capaz de descifrar esa información, ya que es el único que cuenta con la clave privada (clave que nunca sale del DNIe). De esta forma se garantizaría que la comunicación no es legible por el resto de usuarios.

Por otro lado, la comunicación inversa (entre cliente y sistema), se enviará cifrada con la clave privada del usuario, de forma que el sistema podrá descifrarla con su clave pública, garantizándose la autenticidad del usuario, ya que es el único que ha podido cifrar esa información.

De este modo, se ha construido de forma sencilla y eficiente, un sistema de búsqueda federada completamente seguro. Para garantizar aún más la seguridad del sistema, se podrían utilizar mecanismos similares a los que utiliza el protocolo SSH en la autenticación de usuarios (también basados en sistemas criptográficos de clave asimétrica junto con los de cifrado simétrico). Un ejemplo de cómo sería el proceso completo, se puede ver en la Figura 6.5.

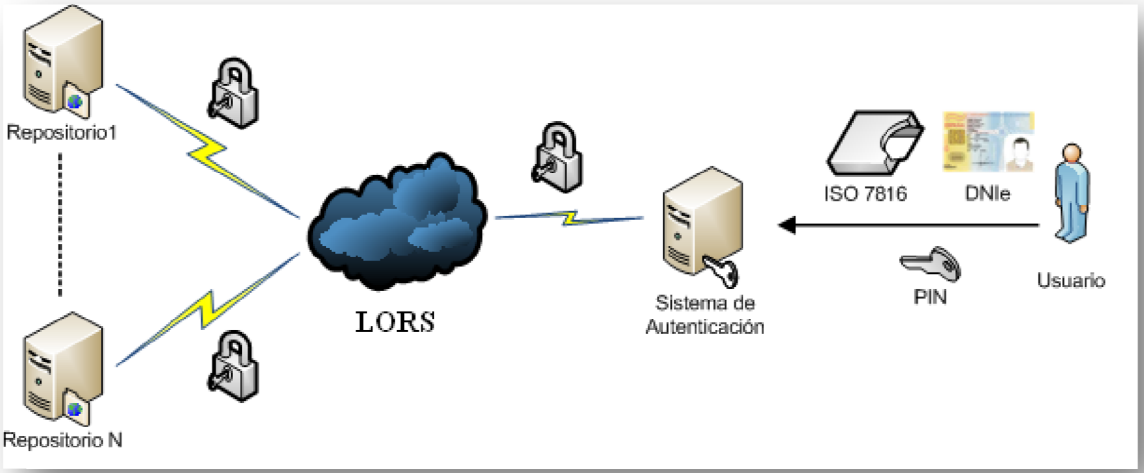


Figura 6.5 Sistema de autenticación del sistema LORS

Se puede ver en la Figura 6.5, cómo a partir del DNIe, y del PIN del mismo, el servicio de autenticación puede cifrar y descifrar toda la información, convirtiendo en canales seguros todos los enlaces a partir del mismo, representados con un candado en la figura. El proceso de cifrado y descifrado se muestra más claramente en la Figura 6.6.

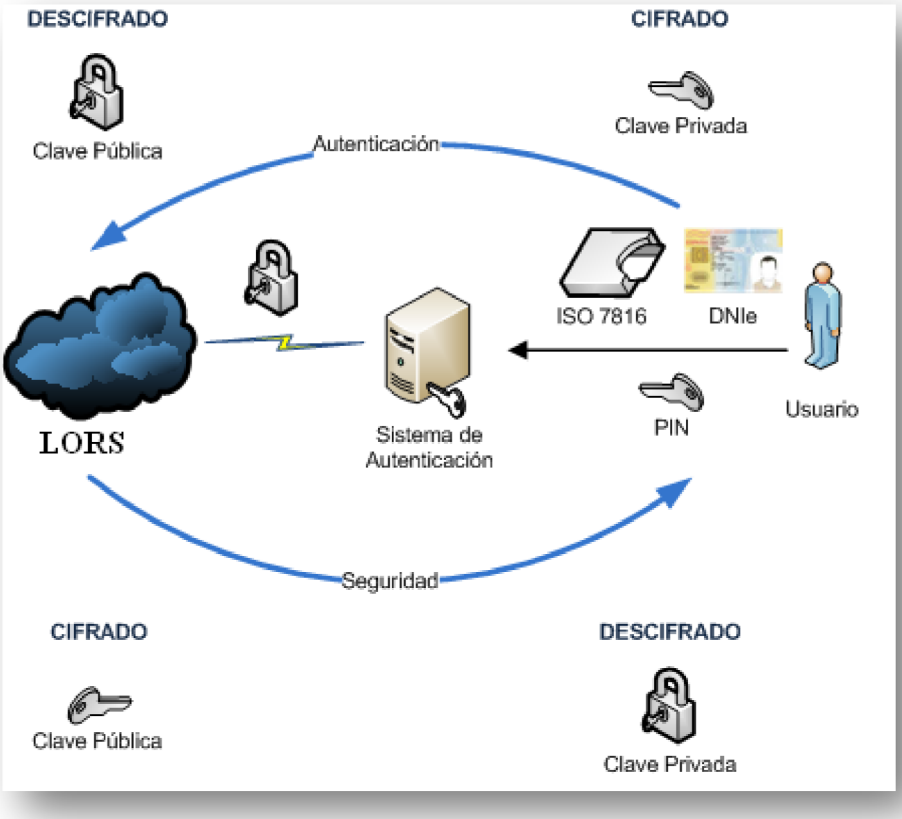


Figura 6.6 Sistema de autenticación detallado. Claves utilizadas y características conseguidas

Como refleja la Figura 6.6, por un lado el sistema consigue aportar autenticación de usuarios cuando el usuario envía la información cifrada con su clave privada y se descifrará con la pública. Por otro lado, consigue seguridad cuando LORS le envía información cifrada al usuario con su clave pública, y éste último, la descifra con su clave privada.

Como ya se ha comentado, en esta primera aproximación no se considera necesario contar con todo este sistema de canales seguros de comunicación, pero, sin embargo queda patente que sería completamente viable y sencillo poder adoptar un sistema de estas características.

Como conclusión, se muestra en la Figura 6.7, una representación completa de cómo quedaría el sistema con todas las características comentadas.

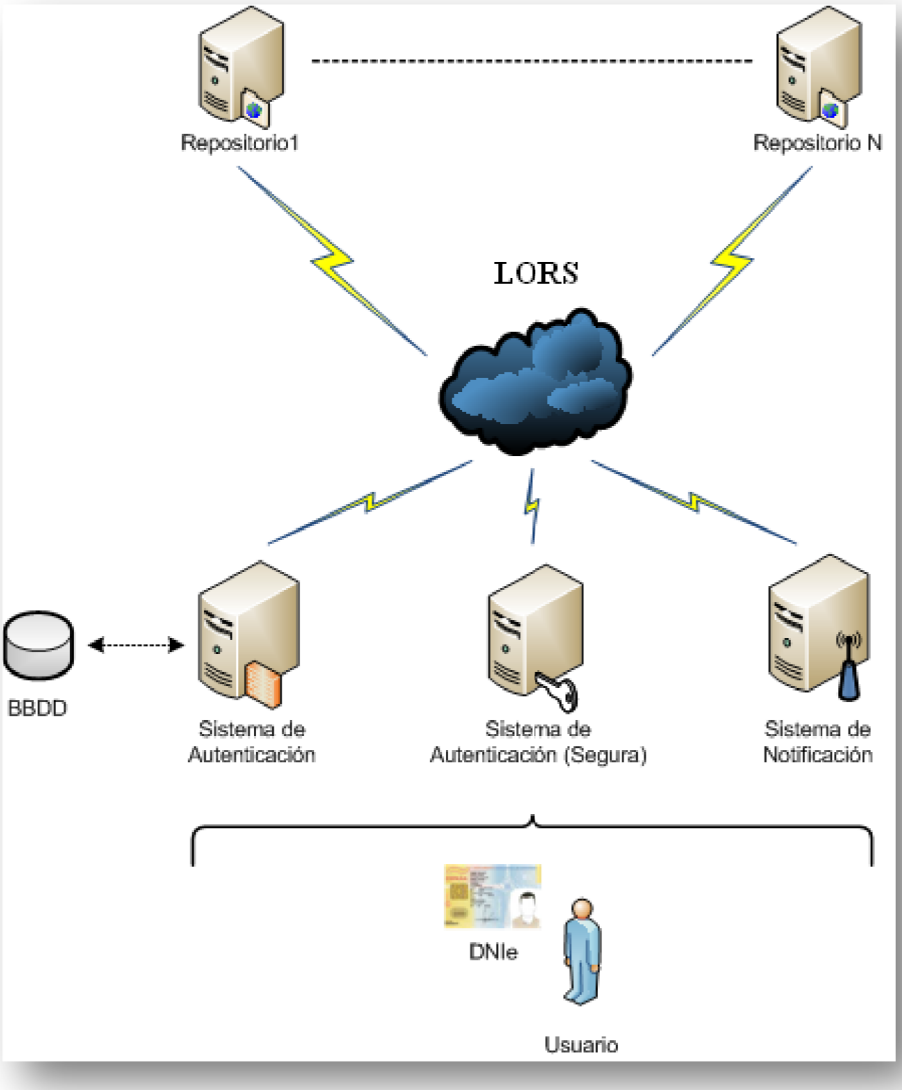


Figura 6.7 Sistema LORS completo

Se puede ver en la Figura 6.7 como gracias al sistema de autenticación, al de autenticación segura y al de notificación se compondría un sistema completo de búsqueda y reutilización de objetos de aprendizaje completamente integrado con la utilización del DNIe.

7. BIBLIOGRAFÍA Y REFERENCIAS

[ActiveBPEL, 2009]

ActiveBPEL: The Open Source BPEL Engine. 2009. http://www.activebpel.org

[ADL, 2004]

ADL: Sharable Content Object Reference Model (SCORM) overview. 2004. http://www.adlnet.gov/scorm/index.cfm

[ADL, 2009]

Advanced Distributed Learning Network. 2009. http://www.adlnet.gov/index.cfm

[ADL-R, 2009]

ADL Registry. 2009. http://adlregistry.adlnet.gov/

[AICC, 2009]

AICC - Aviation Industry CBT Committee. 2009. http://www.aicc.org/

[Alexander et al., 1977]

Alexander, C., Ishikawa, S., Silverstein, M., "A Pattern Language: Towns, Buildings, Construction". Oxford University Press. 1977.

[Allamaraju, 1998]

Allamaraju, S., "Architecture Paradox". 1998. http://www.subrahmanyam.com/articles/architecture/Paradox.html.

[Anido et al., 2002]

Anido, L. E., Fernández, M. J., Caeiro, M., Santos, J. M., Rodríguez, J. S., Llamas, M., "Educational metadata and brokerage for learning resources" Computers & Education 38: 351-374. 2002.

[Apache, 2009]

Apache Tomcat home page. 2009. http://tomcat.apache.org/

[ARIADNE, 2009]

ARIADNE (Alliance of Remote Instructional Authoring and Distribution Network for Europe). 2009. http://www.ariadne-eu.org/

[AYS, 2007]

Revista AYS – Auditoría y Seguridad. Revista Nº14 Octubre 2007. http://www.revista-ays.com/DocsNum14/PersEmpresarial/Miro.pdf

[Barbacci et al., 1997]

Barbacci, M. R., Carriere, S. J., Feiler, P. H., Kazman, R., Klein, M. H., Lipson, H. F., Longstaff, T. A., Weinstock, C. B., "Steps in an Architecture Tradeoff Analysis Method: Quality Attribute Models and Analysis". Technical Report. CMU/SEI-97-TR-029. Carnegie Mellon University, 1997. http://www.sei.cmu.edu/publications/documents/97.reports/97tr029/97tr029abstract.html

[Bass et al., 1997]

Bass, L., Clements, P., Kazman, R., "Software Architecture in Practice". Boston: Addison-Wesley. 1997.

[Batman, 1999]

Batman, J., "Characteristics of an Organization with Mature Architecture Practices". Essays on Software Architecture of Software Engineering Institute. Carnegie Mellon University. 1999. http//www.sei.cmu.edu/architecture/essays.html.

[Boehm y Port, 1998]

Boehm, B., Port, D., "Conceptual Modeling Challenges for Model-Based Architecting and Software Engineering". 1998.

[Booch et al., 1999]

Booch, G., Rumbaugh J., Jacobson, I., "The UML Modeling Language User Guide". Addison-Wesley. 1999.

[Brenner et al., 1998]

Brenner, W., Zarnekow, R., Wittig, H., "Intelligent Software Agents: Foundations and Applications". Springer-Verlag. 1998.

[Brown et al., 1989]

Brown, J. S., Collins, A., Duguid, P., "Situated Cognition and the Culture of Learning. Educational Researcher". 1989.

[BSI, 2009]

BSI (British Standard Institute). 2009. http://www.bsi-spain.com

[Buschmann et al., 1996]

Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M., "Pattern-Oriented Software Architecture". 1996.

[CAREO, 2009]

CAREO (Campus Alberta Repository of Educational Objects). 2009. http://www.ucalgary.ca/commons/careo/

[CBDI, 2006a]

CBDI Service Oriented Architecture Practice Portal. "Web Services Protocols Summary". 2006. http://roadmap.cbdiforum.com/reports/protocols/summary.php

[CBDI, 2006b]

CBDI Forum. 2006. http://www.cbdiforum.com

[CCE, 2000]

Comisión de las Comunidades Europeas. "e-Learning. Concebir la educación del futuro". Comunicación de la Comisión. Bruselas, 25 mayo 2000. Referencia: Bruselas 25.5.2000 COM (2000) 318 Final. 200. http://europa.eu.int/eur-lex/es/com/cnc/2001/com2001_ 0172es01.pdf.

[CEN, 2005]

CEN CWA Simple Query Interface (SQI). 2005. ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/WS-LT/CWA15454-00-2005-Nov.pdf

[CEN, 2007]

CEN CWA Simple Publishing Interface (SPI). 2007. http://ariadne.cs.kuleuven.be/lomi/index.php/SimplePublishingInterface

[Cisco, 2000]

Cisco Systems. "Network Architectures for E-learning Applications". Cisco Systems. 2000. http://www.digitalpipe.com/pdf/dp/white_papers/e_learning/cisco_network_arch.doc

[Cisco, 2001]

Cisco Systems. "Model of an E-Learning Solution Architecture for the Enterprise". Cisco Systems. 2001. http://www.cisco.com/warp/public/10/wwtraining/elearning/learn/whitepaper_docs/solution_architecture_wp.pdf.

[CORDRA, 2009]

CORDRA (Content Object Repository Discovery and Registration/Resolution Architecture). 2009. http://cordra.net/

[Dodero, 2002]

Dodero, J. M., "Una arquitectura multiagente para la producción distribuida de conocimiento y su aplicación al desarrollo compartido de objetos educativos.". Departamento de informática. Universidad Carlos III, Madrid. 2002.

[DOI, 2009]

DOI (Digital Object Identifier). 2009. http://www.doi.org/

[Downes, 2002]

Downes, S., "Design and Reusability of Learning Objects in an cademicContext: A New Economy of Education". National Research Council, Moncton, Canada. 2002. http://www.downes.ca/files/milan.doc

[DublinCore, 2009]

Dublin Core Metadata Initiative. 2009. http://www.dublincore.org

[EDUCASE, 2009]

EDUCASE. 2009. http://www.educause.edu

[Edutella, 2009]

Edutella. 2009. http://www.edutella.org

[EduTools, 2009]

EDUTOOLS. "Product Information" (análisis de productos online). 2009. http://www.edutools.info/static.jsp?pj=4&page=HOME

[Elena, 2009]

Elena project. 2009. http://www.elena-project.org

[FIPA, 2009]

Foundation for Intelligent Physical Agents. 2009. http://www.fipa.org

[Franklin y Graesser, 1996]

Franklin, S., Graesser, A. "Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents". Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages. Springer-Verlag. 1996.

[Friesen, 2004]

Friesen, N., "International LOM Survey: Report. Information Technology for Learning, Education, and Training". 2004. http://mdlet.jtc1sc36.org/doc/SC36_WG4_N0109

[Gamma et al., 2000]

Gamma, E., Helm, R., Johnson, R., Vlissides, J. "Design Patterns: Elements of Reusable Object-Oriented Software". Addison-Wesley. 2000.

[Garlan y Perry, 1995]

Garlan, D., Perry, D., "Special Issue on Software Architecture" (Editor's Introduction). IEEE Transactions on Software Engineering. vol.21, num.4. 1995.

[Glassfish, 2009]

Sun Microsystems Glassfish Server. 2009. https://glassfish.dev.java.net/

[GLOBE, 2009]

GLOBE project. 2009. http://www.globe-info.org/

[Gram et al., 1998]

Gram, M., Mark, T., McGreal, R., "A survey of new media development and delivey software for internet based learning". Industry Canada. Science Promotion and Academic Affairs Branch. Canada. 1998.

[Gutiérrez y Otón, 2004]

Gutiérrez, J. M., Otón, S., "Necesidad de estándares en una arquitectura distribuida para sistemas de teleformación" En: Actas 3ra. Conferencia Iberoamericana en Sistemas, Cibernética e Informática. 2004.

[Henderson, 2003]

Henderson, A. J., "The E-Learning question and answer book, a survival guide for trainers and business managers". 2003.

[Hilera et al., 2005]

Hilera, J.R., Sánchez-Alonso, S., García, E., Del Molino., C.J., "OntoloGLOSE: a Light-weight Software Engineering Ontology". First Workshop on Ontology, Conceptualizations and Epistemology for Software and Systems Engineering (ONTOSE 2005).

[Hilera et al., 2009]

Hilera, J.R., Otón, S., Ortiz, A., Martínez, J.J., Gutiérrez, J.A., De Marcos, L., "Assessment of the standard query interface of public learning objects repositories". IX IEEE International Conference on Advanced Learning Technologies. ICALT 09.

[Hiltz y Norwood, 1994]

Hiltz, S. R., Norwood, N. J., "The Virtual Classroom: Learning without Limits via Computer Networks". Ablex. 1994.

[Hodgins, 2000]

Hodgins, H. W., "Into the Future. A vision paper". Learnativity(ed). 2000. http:/www.learnativity.com/download/MP7.PDF.

[Hodgins, 2001]

Hodgins, H. W., "IEEE LTSC Learning Technology Standards Committee P1484". ADLNET, USA. 2001.

[Horton, 2001]

Horton, W., "Leading e_Learning". American Society for Training & Development". 2001.

[HP, 2009]

HP Web Services. 2009. http://www.hpmiddleware.com/webservices

[IBM, 2008]

IBM Web Services. 2008. http://www.ibm.com/developerworks/webservices

[IEEE, 1990]

IEEE Std 610.12-1990. "IEEE standard glossary of software engineering terminology". New York. 1

[IEEE, 2000]

IEEE Std 1471-2000. "IEEE Recommended Practice for Architectural Description of Software-Intensive Systems". New York. 2000.

[IEEE, 2001]

Learning Technology Standards Committee -LTSC-. IEEE Computer Society, "IEEE P1484.1/D9, 2001-11-30 Draft Standard for Learning Technology—Learning Technology Systems Architecture (LTSA)". 2001. http://edutool.com/ltsa/

[IEEE, 2002]

IEEE 1484.12.1-2002. "Draft Standard for Learning Object Metadata". 2002. http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf

[IEEE, 2009]

Institute of Electrical and Electronics Engineers. 2009. http://www.ieee.org/portal/site

IMS (2002b)

Reusable Definition of Competency or Educational Objective - Best Practice and Implementation Guide. IMS Global Learning Consortium.

[IMS, 2003a]

IMS Global Learning Consortium. "IMS Digital Repositories Interoperability". 2003. http://www.imsglobal.org/digitalrepositories/index.html

[IMS, 2003b]

IMS Global Learning Consortium. "IMS Abstract Framework". 2003. http://www.imsglobal.org/af/index.html

[IMS, 2004a]

IMS Global Learning Consortium. "IMS Enterprise Services". 2004. http://www.imsglobal.org/es/index.html

[IMS, 2004b]

IMS Global Learning Consortium. "IMS Resource List Interoperability". 2004. http://www.imsglobal.org/rli

[IMS, 2004c]

IMS Global Learning Consortium. "IMS Vocabulary Definition Exchange". 2004. http://www.imsglobal.org/vdex/

[IMS, 2005]

IMS Global Learning Consortium. "IMS General Web Services". 2005. http://www.imsglobal.org/gws/index.html

[IMS, 2006]

IMS Global Learning Consortium. "IMS Learning Tools Interoperability". 2006. http://www.imsglobal.org/ti/index.html

[IMS, 2007]

IMS Global Learning Consortium. "IMS Learning Information Services". 2007. http://www.imsglobal.org/enterprise.cfm

[IMS, 2008]

IMS Global Learning Consortium. “IMS Common Cartridge”. 2008. http://www.imsglobal.org/cc/index.html

[IMS, 2009]

IMS Global Learning Consortium. 2009. http://www.imsproject.org/

[ISO, 2004]

International LOM Survey: Report. Information Technology for Learning, Education, and Training. 2004.

[ISO, 2009]

ISO (Organización Internacional para la Estandarización). 2009. http://www.iso.org

[ISO-SC36, 2009]

ISO/IEC JTC1/SC36. Information Technology for Learning, Education, and Training. 2009. http://jtc1sc36.org/

[Jackson, 2001]

Jackson, R. H., “Web Based Learning Resources Library”. 2001.

[Jacobson et al., 1992]

Jacobson, I., Christerson, M., Jonsson P., Overgaard, G., “Object Oriented Software Engineering: A Use Case Driven Aproach”. Addison Wesley. 1992.

[Jacobson et al., 2000]

Jacobson, I., Booch, G., Rumbaugh, J., “El Proceso Unificado de Desarrollo de Software”. Addison Wesley. 2000.

[JADE, 2009]

JADE (Java Agent DEvelopment Framework). 2009. http://jade.tilab.com/

[Jazayeri et al., 2000]

Jazayeri, M., Ran, A., Linden., F., “Software Architecture for Product Families: Principles and Practice”. Addison Wesley. 2000.

[Jena, 2009]

Jena – A Semantic Web Framework for Java. 2009. http://jena.sourceforge.net/

[Kazman et al., 1998]

Kazman, R., Klein, M., Barbacci, M., Longstaff, T., Lipson, H., Carriere, J. , "The Architecture Tradeoff Analysis Method". 4th Int'l Conference on Engineering of Complex Computer Systems (ICECCS98). 1998.

[Koper, 2000]

Koper, R., "From change to renewal: Educational technology foundations of electronic learning environments." Open University of Netherlands. 2000.

[Lanfranco, 1993]

Lanfranco, S., Utsumi, T., "Objects, Agents and Events in a Global Learning Environment. Proceedings of Teleteaching'93". Trondheim (Noruega). 1993.

[Lange y Oshima, 1998]

Lange, D. B., Oshima, M. "Programming and Deploying java mobile agents with aglets". Addison-Wesley. 1998.

[Larman, 2001]

Larman, C., "Applying UML and Patterns - An Introduction to object Analysis and Design ". 2nd Edition, Prentice Hall. 2001.

[Levy, 1993]

Levy, P., "As tecnologias da inteligencia, o futuro do pensamento na era da informática". Editora 34. 1993.

[LUISA, 2009]

LUISA (Learning Content Management System Using Innovative Semantic Web Services Architecture). 2009. http://luisa.atosorigin.es/

[Martignago, 1998]

Martignago, E., "Decentriamo l'insegnamento". En: Sesto potere. Guida per giornalisti, comunicatori aziendali, formatori nell'era di Internet. 1998.

[Martínez et al., 2001]

Martínez, J. J., Sicilia, M. A., Garcia., E., "Extending IMS Course Structures for Conditional Learning Path Support". En: Actas del 3º Simposio Internacional de Informática Educativa. 2001.

[Masie, 2002]

Masie, E., "Making Sense of Learning Specifications & Standards: A Decision Maker's Guide to their Adoption". The Masie Center, Saratoga Springs. http://www.masie.com. 2002.

[McGovern et al., 2003]

McGovern, J., Tyagi, S., Stevens, M., Mathew, S. "Java Web Services Architecture". Morgan Kaufmann. 2003.

[MERLOT, 2009]

MERLOT (Multimedia Educational Resource for Learning and Online Teaching). 2009. http://www.merlot.org

[Microsoft, 2006]

Microsoft .NET. 2006. http://www.microsoft.com/net/default.mspx

[Microsoft, 2009]

Microsoft e-learning. 2009. https://www.microsoftelearning.com/

[Milojicic et al., 1998]

Milojicic, D., Musliner, D., Schrueder, W., "Agents: Mobility and communication". Proceedings of the Thirty-first Annual Hawaii International Conference on System Sciences. VIII. 2-3. 1998.

[Molnar, 1990]

Molnar, A. R., "Computers in Education: a Historical Perspective of the Unfinished Task". T.H.E. Journal 18(4): 80-83. 1990.

[Moreno y Bailly-Baillière, 2002]

Moreno, F., Bailly-Baillière, M., "Diseño instructivo de la formación on-line. Aproximación metodológica a la elaboración de contenidos". Ariel Educación, Barcelona. 2002.

[Morrison, 2004]

Morrison, D., "E-Learning Strategies How to get implementation and delivery right first time". Wiley Publishing. 2004.

[MS, 2009]

Microsoft Web Services. 2009. http://msdn.microsoft.com/webservices

[MTOM, 2005]

W3C Working Group. SOAP Message Transmission Optimization Mechanism (MTOM). 2005. http://www.w3.org/TR/soap12-mtom/

[MySQL, 2009]

MySQL home page. 2009. http://www.mysql.com/

[Nasseh, 1996]

Nasseh, B., "Artificial Intelligence and Internet". Ball State University. 1996. http://www.bsu.edu/classes/ne

[Newcomer, 2002]

Newcomer, E., "Understanding Web Services: XML, WSDL, SOAP, and UDDI". Addison-Wesley. 2002.

[Noy, 2001]

Noy, N., McGuinness, D., "Ontology Development 101: A Guide to Creating Your First Ontology". Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001.

[OASIS, 2009]

Organization for the Advancement of Structured Information Standards (OASIS). Web Services Business Process Execution Language (WSBPEL). 2009. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel

[OMG, 2001]

OMG Architecture Board Mad Drafting Team. "Model Driven Architecture: A Technical Perspective". Architecture Board ORMSC1 (ed.). Document number ormsc/2001-07. 2001.

[OpenURL, 2009]

OpenURL. 2009. http://en.wikipedia.org/wiki/OpenURL

[Ortiz, 2006]

Ortiz, A. "Arquitectura orientada a servicios Web para la implementación de repositorios y unidades de aprendizaje. Tomo II". Proyecto Fin de Carrera. Ingeniería en Informática. Universidad de Alcalá. 2006.

[Ortiz et al., 2006]

Ortiz, A., Otón, S., Hilera J.R., "Hacia la utilización de Servicios Web Semánticos en Repositorios de Objetos de Aprendizaje". III Simposio Pluridisciplinar sobre Objetos y Diseños de Aprendizaje Apoyados en la Tecnología (od@06). 2006.

[Otón, 2006]

Otón, S. "Propuesta de una arquitectura software basada en servicios para la implementación de repositorios de objetos de aprendizaje distribuidos". Tesis Doctoral. Departamento de Ciencias de la Computación. Universidad de Alcalá. 2006.

[Otón et al., 2005]

Otón, S., Hilera, J.R., Gutiérrez, I., Ortiz, A., "Arquitectura orientada a servicios Web para la implementación de repositorios distribuidos de objetos y unidades de aprendizaje". I Simposio Nacional de Tecnologías de la Información y de las Comunicaciones en la Educación (SINTICE 2005). 2005.

[Otón et al., 2006a]

Otón, S., Ortiz, A., Hilera, J.R., "SROA: Sistema de Reutilización de Objetos de Aprendizaje". VIII Congreso Iberoamericano de Informática Educativa. 2006.

[Otón et al., 2006b]

Otón, S., Ortiz, A., Barchino, R., "Arquitectura orientada a servicios para la reutilización de objetos de aprendizaje distribuidos". IADIS Conferencia Ibero Americana WWW/Internet 2006. 2006.

[Otón et al., 2007]

Otón, S., Ortiz, A., Hilera, J.R., "SROA: Sistema de Reutilización de Objetos de Aprendizaje". Revista Iberoamericana de Informática Educativa IE Comunica. 2007.

[Otón et al., 2008]

Otón, S., Ortiz, A., Hilera, J.R., Martínez, J.J., Barchino, R., Gutiérrez, J.M., Gutiérrez, J.A., De Marcos, L., "The Integration Of SQI in a Reusable Learning Objects System: Advantages and Disadvantages". X International Conference on Information Integration and Web-Based Application & Services. IIWAS 08

[Otón et al., 2010]

Otón, S., Ortiz, A., Hilera, J.R., Barchino, R., Gutiérrez, J.M., Martínez, J.J., Gutiérrez, J.A., de Marcos, L., Jiménez, L., "Service Oriented Architecture for the Implementation of Distributed Repositories of Learning Objects". International Journal of Innovative Computing, Information and Control (IJICIC). Índice de impacto JCR (2008) de 2,791.

[Pagés et al., 2003]

Pagés, C., Sicilia, M. A., García, E., Martínez, J. J., Gutiérrez, J. M., "On the Evaluation of Completeness of Learning Object Metadata in Open Repositories". Proceedings of the Second International Conference on Multimedia and Information & Communication Technologies in Education (m-ICTE 2003), 1760-17

[Patron, 2003]

Patron, L., "Online Learning Related Terminology". UNU Online Learning. 2003. http://www.onlinelearning.unu.edu/images/documents/UNU%20OL4-terms.pdf

[Pelechano, 2005]

Pelechado, V., "Servicios Web. Estándares, Extensiones y Perspectivas de Futuro". ISIS. 2005.
http://pangea.upv.es/N+ISIS05/documents/VicentePelechano/ServiciosWeb.pdf

[Piskurich, 2003]

Piskurich, G. M., "The AMA Handbook of E-Learning: Effective Design, Implementation, and Technology Solutions". Amacon. 2003.

[Plekhanova, 2002]

Plekhanova, V., "Intelligent Agent Software Engineering". Idea Group Publishing. 2002.

[POA, 2009]

Programación Orientada a Aspectos (POA). 2009. http://es.wikipedia.org/wiki/Programaci%C3%B3n_Orientada_a_Aspectos

[Pozo, 2002]

Pozo I., "Aprendices y maestros: la nueva cultura del aprendizaje.". Madrid: Alianza Editorial. 2002.

[Protégé, 2009]

The Protégé Ontology Editor and Knowledge Acquisition System. 2009. http://protege.stanford.edu/

[PURL, 2009]

PURL (Persistence Uniform Resource Locator). 2009. http://purl.org/

[RAE, 2009]

Real Academia Española. 2009. http://www.rae.es/

[Rehak,2003]

Rehak, D., "e-Learning Standards Questions, Decisions, Actions". Learning Systems Architecture Lab, Carnegie Mellon University, Pittsburg. 2003.

[Reload, 2009]

Programa Reload. 2009. http://www.reload.ac.uk/

[Rosenberg, 2001]

Rosenberg, M. J. "e-Learning. Strategies for delivering knowledge in the digital age". McGraw-Hill. 2001.

[SAAJ, 2009]

SAAJ (SOAP with Attachments API for Java). 2007. https://saaj.dev.java.net/

[Sancho, 2002]

Sancho, P., "Lenguajes de marcado y su aplicación en el dominio de las tecnologías de aprendizaje Web". Universidad Complutense de Madrid. 2002.

[Sarasa et al., 2008]

Sarasa, A., Canabal, J.M., Sacristán J.C., Jiménez, R. "Uso de IMS VDEX en AGREGA", SIIE (X Simposio Internación de Informática Educativa) Salamanca (Spain). 2008.

[Sharma y Kitchens, 2004]

Sharma, S., Kitchens, F., "Web Services Architecture for M-Learning, Electronic Journal on e-Learning", Vol.2, Issue 1. 2004.

[Shaw y Garlan, 1996]

Shaw, M., Garlan, D., "Software Architecture: Perspectives on an Emerging Discipline". Prentice Hall, 1996.

[Schoolnet, 2009]

Schoolnet Europa. 2009. http://www.eun.org

[SeCo, 2009]

SeCo - Semantic Computing Research Group. Universidad de Helsinki. 2009. http://www.seco.tkk.fi

[Semántica, 2009]

Wikipedia. "Definición de Semántica". 2009. http://es.wikipedia.org/wiki/Semántica

[Sicilia et al., 2005]

Sicilia, M., García, E., Sánchez-Alonso, S., Soto, J. "A semantic lifecycle approach to learning object repositories". Proceedings of ELETE 2005 - IEEE eLearning on Telecommunications, 17/07/2005, p.466- 471, (2005)

[Sigh, 2001]

Sigh, H., "Learning Content Management Systems". E-learning magazine. 2001.

[SkillSoft, 2009]

SkillSoft: Integrating the learning in the life of the enterprise. 2009. http://www.skillsoft.com/

[Soni et al., 1995]

Soni, D., Nord, R. L., Hofmeister, C., "Software architecture in industrial applications". En: Actas de 17th International Conference on Software Engineering. New York: ACM Press. p.196-207. 1995.

[Sosteric y Hesemeier, 2002]

Sosteric, M., Hesemeier, S., "When is a learning object not an object: a first step towards a theory of learning objects". En: International Review of Research in Open and Distance Learning vol. 3 num. 2. 2002. http://www.irrodl.org/content/v3.2/soc-hes.html

[Soto et al., 2007]

Soto, J., García, E., Sanchez-Alonso, S. "Semantic Learning Objects Repositories". International Journal of Continuing Engineering Education and Life-Long Learning 2007 - Vol. 17, No. 6 pp. 432 – 446

[TIOBE, 2009]

TIOBE Programming Community Index for March 2009. 2009. http://www.tiobe.com/tpci.htm

[UDDI, 2009]

UDDI Consortium. 2009. http://www.uddi.org

[UMA, 2009]

Grupo de Ingeniería del Software de la Universidad de Málaga. 2009. http://www.lcc.uma.es

[W3C, 1998]

W3C Working Group. eXtensible Markup Language (XML). 1998. http://www.w3.org/XML

[W3C, 1999]

W3C Working Group. Resource Description Framework (RDF). 1999. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

[W3C, 2001a]

W3C Working Group. Web Services Description Language (WSDL) 1.1, 15 March 2001, Erik Christensen, Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, http://www.w3.org/TR/wsdl

[W3C, 2001b]

W3C Working Group. W3C Semantic Web Activity. 2001. http://www.w3.org/2001/sw/

[W3C, 2001c]

W3C Working Group. DAML+OIL. 2001. http://www.w3.org/TR/daml+oil-reference

[W3C, 2004a]

W3C Working Group. Note, D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, D. Orchard. "Web Services Architecture". 11 February 2004. http://www.w3.org/TR/ws-arch/

[W3C, 2004b]

W3C Working Group. Note, D. Booth, H. Haas, D. Orchard. "Web Services Architecture Usage Scenarios". 11 February 2004. http://www.w3.org/TR/ws-arch-scenarios/

[W3C, 2004c]

W3C Working Group. Ontology Web Language (OWL). 2004. http://www.w3.org/TR/2004/REC-owl-features-20040210/

[W3C, 2004d]

W3C Working Group. OWL-S: Semantic Markup for Web Services (OWL-S). 2004. http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/

[W3C, 2004e]

W3C Working Group. XML Schema. 2004. http://www.w3.org/XML/Schema

[W3C, 2004f]

W3C Working Group. RDF Schema. 2004. http://www.w3.org/TR/rdf-schema/

[W3C, 2005a]

W3C Working Group. Web Service Modeling Ontology (WSMO). 2005. http://www.w3.org/Submission/WSMO/

[W3C, 2005b]

W3C Working Group. Web Service Execution Environment (WSMX). 2005. http://www.w3.org/Submission/WSMX/

[W3C, 2005c]

W3C Working Group. Web Service Modeling Language (WSML). 2005. http://www.w3.org/Submission/WSML/

[Winer, 1999]

Winer, D., "XML-RPC Specification". 1999. http://www.xmlrpc.com/

[Wooldridge, 2002]

Wooldridge, M., "Introduction to MultiAgent Systems". John Wiley & Sons. 2002.