Universidad de las Ciencias Informáticas

Facultad 4

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Título: Implementación de estándares de interoperabilidad en el repositorio de recursos educativos RHODA.

Autores: *Analiet Puentes Daniel*

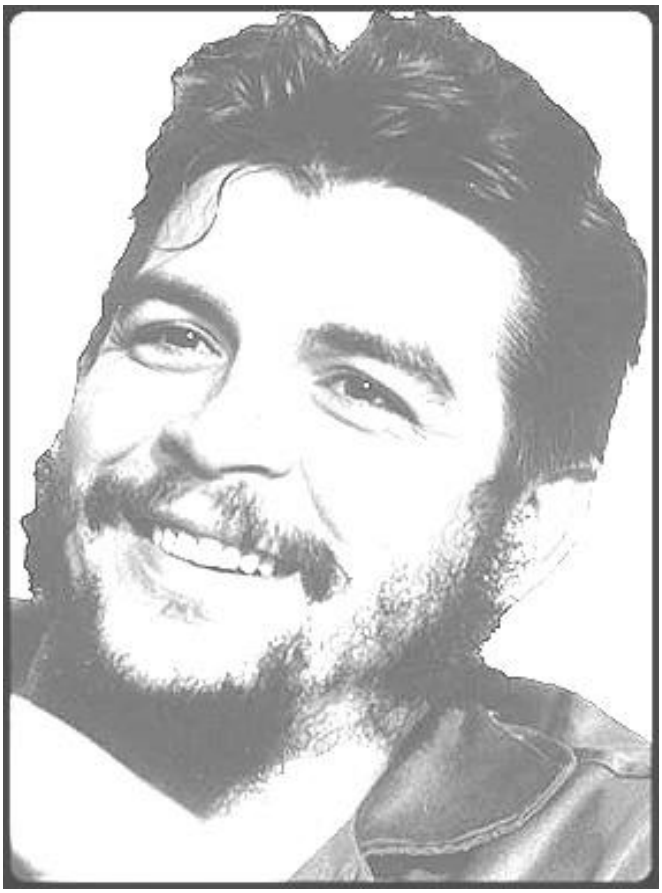
Anyel Jiménez Betancourt

Tutor: *Ing. Yailen San Juan Santana*

Ing. Yandris Mata Cabrera

La Habana, 2014

"Año 56 de la Revolución"



"El secreto del éxito radica en saber que debemos mantener cerca de nosotros y de que debemos alejarnos".

Ernesto Ché Guevara

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas; para que haga el uso que estime pertinente.

Para que así conste firmamos la presente a los __ días del mes de junio del año 2014.

AUTORES

Analiet Puentes Daniel

Anyel Jimenez Betancourt

TUTORES

Ing. Yailen San Juan Santana

Ing. Yandris Mata Cabrera

Anyel

A mis padres Mercedes de la Caridad Betancourt Ruiz y Angel Luis Jiménez Meneses, por ser mi fuente de inspiración y confianza.

A mi padrastro Aldo Reyes que llego a mi vida con tan solo 3 años de edad y ha sido capaz de apoderarse de una parte de mi corazón para siempre, por apoyarme en los momentos más tensos de mi vida y carrera, gracias.

En especial a mis abuelos que aunque no estén conmigo sé que están muy orgullosos de mí.

A mi preciosa hermana que aunque siempre la estoy molestando la quiero con todo mi corazón. Te quiero mucho Baby.

A mi novia Idailis Gutierrez Reyes por su cariño y amor incondicional, por estar en los momentos difíciles, gracias por estar en mi vida.

A mi compañera de tesis, muchas gracias por haberme escogido para este trabajo tan importante.

A mi tía Bertha, a mis primos Alexander y Aleysbert por darme siempre su apoyo incondicional.

A Lourdes, Mayor y su hija Gleysis por apoyarme siempre.

A todas aquellas personas que de alguna manera han transitado por mi vida y han dejado en ella una huella indestructible que no se borrará con el paso del tiempo.

Analiet

A mi mamá por apoyarme y creer siempre en mí, porque nunca dejo de hacerlo y siempre estuvo ahí cuando la necesite para darme su apoyo incondicional, además por no cortarme las alas y apoyarme en todas mis decisiones. Por otra parte a mi papá por su preocupación y ayudarme en todo lo que pudo.

A mi papito Jose por estar ahí y hacerme sentir bien cuando más lo necesité, porque siempre en los momentos más difíciles estuvo para apoyarme. Además le agradezco a su familia.

A mis abuelos los que están y los que no, por darme los padres que me dieron y porque de una forma u otra han estado apoyándome en todas las etapas de mi vida.

A todos mis tíos y en esto incluyo a los que no son de sangre, pero que para mí sí lo son, por creer en mí aun cuando yo misma no lo hice y por apoyarme en todo. En especial a mi tío Octavio que aunque ya no está conmigo desde muy pequeña me encaminó en la vida para que hoy en día fuera lo que soy hoy.

A mis primos por siempre ayudarme en todo lo que han podido. En especial a mi prima Eve por haber compartido conmigo tantas experiencias en la vida desde muy pequeñas siempre fuimos una.

A mi madrastra y a toda su familia por ayudarme de una forma u otra.

A mi compañero de tesis Anyel porque juntos hicimos hasta lo imposible por lograr este sueño.

A mis tutores Yailen y Yandris por ayudarme durante este año en todo lo que pudieron.

A todos mis amigos de la vida que me han acompañado desde la infancia y durante la etapa de cada una de las enseñanzas que he pasado, que aunque ya ninguno estudie conmigo de una forma u otra han estado dando ánimos desde lejos. En especial a Ceci, Lilo, Yadira, Yelenis, Dayrelis, Mayi, Marinee, y muchos más.

A todos mis amigos que me han acompañado durante estos 5 años, juntos pasamos muchos momentos buenos y malos, enfrentándonos a todos los obstáculos que se nos fueron poniendo durante estos años. En especial a Elena, Yanet, Rosalía, Yuliexa, Nadia, Pablo, Ruby, Yudy, Jorgito, Carlos por fastidiar tanto, Liena, Yoan, Ramón, Eduardo,

Angel, y disculpen los que se me queden, pero recuerden siempre que aunque no los mencione los recordaré.

A todos muchas gracias.

Anyel

Dedico este trabajo a mis padres Mercedes de la Caridad Betancourt Ruiz y Angel Luis Jiménez Meneses, por ser mi fuente de inspiración y confianza.

A mi padrastro Aldo Reyes que llego a mi vida con tan solo 3 años de edad y ha sido capaz de apoderarse de una parte de mi corazón para siempre, por apoyarme en los momentos más tensos de mi vida y carrera, gracias.

En especial a mis abuelos que aunque no estén conmigo sé que están muy orgullosos de mí.

A mi preciosa hermana que aunque siempre la estoy molestando la quiero con todo mi corazón. Te quiero mucho Baby.

A mi novia Idailis Gutierrez Reyes por su cariño y amor incondicional, por estar en los momentos difíciles, gracias por estar en mi vida.

A mi compañera de tesis, muchas gracias por haberme escogido para este trabajo tan importante.

A mi tía Bertha, a mis primos Alexander y Aleysbert por darme siempre su apoyo incondicional.

A Lourdes, Mayor y su hija Gleysis por apoyarme siempre.

A todas aquellas personas que de alguna manera han transitado por mi vida y han dejado en ella una huella indestructible que no se borrará con el paso del tiempo.

Analiet

A mi mamá por apoyarme y creer siempre en mí, porque nunca dejo de hacerlo y siempre estuvo ahí cuando la necesite para darme su apoyo incondicional, además por no cortarme las alas y apoyarme en todas mis decisiones. Por otra parte a mi papá por su preocupación y ayudarme en todo lo que pudo.

A mi papito Jose por estar ahí y hacerme sentir bien cuando lo más lo necesite, porque siempre en los momentos más difíciles estuvo para apoyarme. Además le agradezco a su familia.

A mis abuelos los que están y los que no, por darme los padres que me dieron y porque de una forma u otra han estado apoyándome en todas las etapas de mi vida.

A todos mis tíos y en esto incluyo a los que no son de sangre, pero que para mí sí lo son, por creer en mí aun cuando yo misma no lo hice y por apoyarme en todo. En especial a mi tío Octavio que aunque ya no está conmigo desde muy pequeña me encaminó en la vida para que hoy en día fuera lo que soy hoy.

A mis primos por siempre ayudarme en todo lo que han podido. En especial a mi prima Eve por haber compartido conmigo tantas experiencias en la vida desde muy pequeñas siempre fuimos una.

A mi madrastra y a toda su familia por ayudarme de una forma u otra.

Resumen

La incorporación de internet y las nuevas tecnologías en el sector educacional, han posibilitado la creación de aplicaciones que promueven el aprendizaje electrónico. En esta modalidad, resulta imprescindible que dichas aplicaciones sean capaces de interoperar entre sí, a fin de reutilizar y compartir los recursos educativos que se crean. Un ejemplo de estas herramientas, es el repositorio creado en la Universidad de las Ciencias Informáticas, dedicado al almacenamiento y gestión de objetos de aprendizaje. RHODA, es capaz de comunicarse con otros repositorios y plataformas e-learning, haciendo uso para ello de estándares de interoperabilidad. Sin embargo, debido a las necesidades de ampliar su alcance hasta la gestión de cualquier tipo de recurso educativo y de utilizar el marco de trabajo Xalix, es preciso implementar una nueva versión, manteniendo su interoperabilidad. La presente investigación tiene como objetivo: implementar estándares de interoperabilidad en RHODA, que permitan su comunicación con herramientas externas. El desarrollo de la solución estuvo guiado por la metodología RUP, utilizando el Visual Paradigm 8.0 y UML para el modelado de las funcionalidades. En la implementación, se emplearon los framework Symfony 2.3.7 y Bootstrap 3.0, así como las herramientas, NetBeans 7.3.1, PostgreSQL 9.1 y el servidor web Apache 2.2. Además, se usaron los lenguajes PHP5, CSS3, HTML5, JavaScript 1.3. Como resultado, se obtuvo un módulo que incorpora los estándares SQI, SPI y OAI-PMH para la búsqueda federada, publicación de recursos educativos y recolección de metadatos en aplicaciones e-learning.

Palabras Clave

e-learning, estándar, interoperabilidad, servicios web

Índice de Contenido

Índice de Figuras

Índice de Tablas

Introducción

En los últimos años, con el avance de las Tecnologías de la Información y las Comunicaciones (TIC), han surgido métodos que permiten transmitir el conocimiento por medios electrónicos, ignorando de este modo la posición geográfica del estudiante. Estos métodos adquieren un verdadero protagonismo con la llegada de internet, dando lugar a lo que hoy se conoce como aprendizaje electrónico (e-learning), que se define como *"conjunto de tecnologías, aplicaciones y servicios orientados a facilitar la enseñanza y el aprendizaje a través de Internet/Intranet, que facilitan el acceso a la información y la comunicación con otros participantes".*(1)

El e-learning, como se indica en el concepto anterior, no pretende sustituir los viejos modelos de enseñanza y mucho menos la figura del profesor, sino otorgar a los estudiantes un mayor protagonismo, haciendo que los mismos sean capaces de gestionar su propio conocimiento mediante la interacción con diferentes aplicaciones, tales como los Sistemas de Gestión del Aprendizaje del inglés Learning Management System (LMS), las Bibliotecas Digitales (BD) y los Repositorios Digitales (RD). Cada una de estas herramientas cumple un marcado objetivo dentro de un entorno e-learning, como puede ser la creación de los contenidos educativos, su almacenamiento y gestión, así como la conformación de cursos a partir de los mismos. Sin embargo, su funcionamiento no puede realizarse del todo aislado, es necesario que estas aplicaciones sean capaces de comunicarse entre sí para de esta forma, intercambiar y reutilizar los contenidos creados. Esta capacidad de comunicación entre sistemas para intercambiar la información y utilizarla, es definida por el Instituto de Ingenieros Eléctricos y Electrónicos, del inglés Institute of Electrical and Electronics Engineers (IEEE), como interoperabilidad.(2)

Lograr la interoperabilidad entre las diferentes aplicaciones con fines educativos, no resulta una tarea fácil, más aún cuando la heterogeneidad de las tecnologías actuales utilizadas para la creación de las mismas, genera problemas de incompatibilidad entre sí, provocando la pérdida de los recursos y materiales educativos.

En la actualidad, son muchas las personas agrupadas en organizaciones y consorcios que dedican esfuerzos a resolver esta problemática, dentro de los más reconocidos figuran: el Consorcio de Aprendizaje Global del Sistema de Gestión de la Información, del inglés Information Management System-Global Learning Consortium (MS GLC), el grupo Aprendizaje Avanzado Repartido (ADL, por sus siglas en inglés Advanced Distributed Learning), el Comité de la Industria de Aviación Basada en Computadoras,

del inglés Aviation Industry Computer-Based Training Comitee (AICC) y el IEEE. Los resultados de las investigaciones realizadas por estas organizaciones, consisten fundamentalmente, en especificaciones y estándares que han llegado a tener una amplia aceptación por las soluciones que engloban. Algunos de los principales estándares propuestos son los siguientes: la Iniciativa de Archivos Abiertos-Protocolo para la Recolección de Metadatos en inglés Open Archives Initiative–Protocol for Metadata Harvesting (OAI-PMH), la Interfaz de Publicación Simple, del inglés Simple Query Interface (SPI) y la Interfaz de Consulta Simple o Simple Query Interface (SQI).(3) De manera general, estos estándares definen cómo debe realizarse la comunicación entre diferentes sistemas con el objetivo de facilitar la búsqueda, recuperación y publicación de los contenidos educativos, especificando para ello, un conjunto de métodos. Estos métodos suelen ser implementados como servicios web, de manera tal que cualquier sistema que incorpore los estándares, pueda consumir dichos servicios y establecer la interoperabilidad.

Actualmente existe un gran número de plataformas centradas en la educación a distancia, en Cuba, particularmente, son varias las instituciones que se han interesado en apoyar el desarrollo de las mismas. Una de ellas es la Universidad de las Ciencias Informáticas (UCI), la cual constituye un ejemplo de constante transformación e innovación. En ella, se han creado un conjunto de herramientas dedicadas a promover un proceso de enseñanza y aprendizaje más ligado a las tecnologías, entre las que figuran: la Herramienta de Autor CRODA, en la cual se crean Objetos de Aprendizaje (OA), un Entorno Virtual de Aprendizaje (EVA) basado en la plataforma Moodle, una BD en la que se realizan búsquedas por catálogos de diferentes temas y el Repositorio de Objetos de Aprendizaje RHODA, donde se gestionan y almacenan los OA.

RHODA en su versión 2.2, cuenta con la implementación de los métodos propuestos por los estándares SQI, SPI y OAI-PMH, los que permiten la realización de búsquedas federadas, la publicación de OA en el repositorio por parte de otras herramientas y la recolección de metadatos en sistemas externos que implementen este estándar.

Con vista a integrar las plataformas con fines educativos que pertenecen a FORTES, se definió como política establecida por él mismo, un nuevo marco de trabajo, el cual ha generado problemas de compatibilidad con el marco utilizado en el desarrollo de las versiones anteriores de RHODA. Esta incompatibilidad viene dada por las diferencias existentes entre ambos marcos, fundamentalmente, a la hora de acceder a la información que se encuentra contenida en la base de datos de los sistemas. Teniendo en cuenta que RODHA fue desarrollado haciendo uso de una versión inferior del marco de

trabajo Symfony a la que se propone emplear, y que las versiones del mismo varían significativamente en cuanto a la sustitución de los plugins[1] por un nuevo concepto (bundles[2]), además de sustituir el objeto Propel por Doctrine[3], se hace necesario implementar una nueva versión del repositorio, ya que estos cambios impiden migrar las funcionalidades de la versión anterior.

A partir de la problemática anteriormente descrita, se identificó como **problema de investigación**: ¿Cómo contribuir a la interoperabilidad del repositorio de recursos educativos RHODA?

Se define como **objeto de estudio**: la interoperabilidad en herramientas e-learning.

Este a su vez se enmarca en el **campo de acción**: los estándares de interoperabilidad SQI, SPI y OAI-PMH para la comunicación de las herramientas dentro de un entorno e-learning.

Para dar solución al problema planteado se define como **objetivo general**: Implementar estándares de interoperabilidad en RHODA, que permitan su comunicación con herramientas externas.

Dicho objetivo general se desglosa en los siguientes **objetivos específicos**:

- ✓ Elaborar el marco teórico-conceptual relacionado con la comunicación entre las herramientas e-learning mediante el uso de estándares de interoperabilidad.
- ✓ Desarrollar las funcionalidades que contribuyan a la interoperabilidad de RHODA con herramientas externas.
- ✓ Aplicar pruebas al módulo desarrollado para comprobar su funcionamiento.

Idea a defender: con la incorporación de estándares para la interoperabilidad en RHODA, se contribuirá a la comunicación del repositorio con herramientas externas permitiendo el intercambio y la reutilización de los recursos educativos.

Para dar cumplimiento a los objetivos se plantean las siguientes **tareas de investigación**:

- ✓ Estudio de la implementación de métodos que proponen los estándares SQI, OAI-PMH y SPI en la versión 2.2 de RHODA.

[1] Los **plugins** permiten agrupar todo el código diseminado por diferentes archivos y reutilizar este código en otros proyectos.

[2] Los **bundles** permiten utilizar funcionalidades construidas por terceros o empaquetar las propias funcionalidades del usuario para distribuirlas y reutilizarlas en otros proyectos.

[3] **ORM de Doctrine** permite asociar objetos a una base de datos relacional (tal como MySQL, PostgreSQL o Microsoft SQL).

- ✓ Selección de las clases y métodos que deben ser implementados para garantizar la interoperabilidad de RHODA en su nueva versión.
- ✓ Definición de los requerimientos funcionales y no funcionales.
- ✓ Estudio de las herramientas y tecnologías a utilizar.
- ✓ Análisis y diseño de las funcionalidades definidas.
- ✓ Implementación de las funcionalidades diseñadas previamente.
- ✓ Selección del método y técnica de pruebas a utilizar para comprobar el funcionamiento de los requisitos implementados.
- ✓ Ejecución de las pruebas definidas con anterioridad.
- ✓ Corrección de los errores detectados durante la ejecución de las pruebas.

Los métodos que se utilizan para la presente investigación son:

El método teórico **histórico-lógico** permite investigar el origen y evolución de la interoperabilidad, así como el surgimiento de estándares que facilitan la difusión de contenidos educativos mediante la comunicación de las aplicaciones que intervienen en un entorno e-learning.

El método teórico **analítico-sintético** permite realizar un estudio de los estándares de interoperabilidad SQI, OAI-PMH y SPI, implementados en versiones anteriores de RHODA y sintetizar los métodos que deben ser implementados en la presente versión de manera que facilite la comunicación de RHODA con herramientas externas.

El método teórico de **modelación** permite representar la información generada durante los flujos de trabajo de análisis y diseño, implementación y prueba.

El presente trabajo se estructura en tres capítulos:

Capítulo 1: Fundamentación Teórica. Se precisan elementos teóricos que sustentan la investigación y el desarrollo del tema propuesto, a través del estudio y análisis de soluciones existentes. Se hace referencia a los principales aspectos relacionados con los estándares para la interoperabilidad de RHODA con las demás herramientas externas. Se describen las herramientas, tecnologías y la metodología a utilizar en el desarrollo de la solución propuesta.

Capítulo 2: Propuesta de solución. Se definen los requisitos que debe tener el sistema y se modelan los casos de usos que responden a los mismos. Se describe la solución que se propone a partir de los diagramas de clases del análisis y clases del diseño a las funcionalidades del sistema, así como los diagramas de colaboración.

Capítulo 3: Implementación y prueba. Comprende el diagrama de componentes, obteniéndose una descripción para la implantación del sistema. Se describe el método y técnica de pruebas a utilizar para comprobar el correcto funcionamiento de los requerimientos desarrollados, así como los resultados obtenidos.

Capítulo 1: Fundamentación Teórica

Introducción del capítulo

En el desarrollo de sistemas y adaptación de materiales para la educación, basados en las TICs, las instituciones han utilizado, principalmente, distintas tecnologías y arquitecturas incompatibles entre sí, lo que ha generado problemas a la hora de intercambiar y reutilizar los recursos. Con el fin de resolver este problema, algunas organizaciones han invertido esfuerzos en dictar estándares que especifican cómo lograr una comunicación entre los sistemas, así como empaquetar los contenidos de manera que puedan ser manejados por las mismas.

1.1. Repositorio Digital

Los RD son *"herramientas eficaces para almacenar, organizar y hacer uso eficiente de la información y el conocimiento. Constituyen un sitio centralizado donde se almacena y mantiene información digital, habitualmente bases de datos o archivos informáticos. Los repositorios están preparados para distribuirse habitualmente sirviéndose de una red informática como Internet"*. (4)

Se puede considerar además, a los RD como *"estantes, armarios o archivos centralizados donde se almacena, gestiona y controla un gran volumen de información digital, como documentos, imágenes, videos, bases de datos, entre otros"*. Según la naturaleza de la información digital que almacenan, los RD se clasifican en dos grandes grupos, los repositorios temáticos que albergan contenidos de uno o varios temas específicos, y los repositorios institucionales que almacenan toda la información generada en una institución.(5)

Otros autores clasifican los RD según el tipo de contenido que almacenan, existiendo de este modo Repositorios de OA y Repositorios de Recursos Educativos (RRE).

1.1.1. Repositorio de Recursos Educativos

Un RRE es *"un catálogo electrónico/digital que facilita las búsquedas en internet de objetos digitales para el aprendizaje"*.(6)

Cañizares define este tipo de repositorio como un *"sistema especializado en el almacenamiento de recursos educativos, accesibles y operables por usuarios a través de Internet/Intranet, con funciones encaminadas a la: clasificación, localización, reutilización, recuperación y mantenimiento de los recursos, que permita compartir estos con otras herramientas de un entorno e-learning"*.(7)

Tomando como punto de partida la experiencia de los usuarios en interacción con los diferentes repositorios existentes, surgen factores técnicos, pedagógicos y ergonómicos que determinan las características fundamentales que deben tener los recursos educativos digitales. Estas características son:(8)

- **Interactividad**: el diseño de recursos interactivos proporciona la base para el desarrollo de experiencias de aprendizaje más ricas. Se asegura una motivación intrínseca al contemplar la posibilidad de tomar decisiones, realizar acciones y recibir una respuesta más inmediata a las mismas. La manipulación directa de variables o parámetros en situaciones de simulación o experimentación permite estrategias de aprendizaje por ensayo-error. El desarrollo de itinerarios de aprendizaje individuales a partir de los resultados obtenidos en cada paso favorecen una individualización de la enseñanza. La interactividad también tiene una dimensión social que puede facilitar que el alumno/a participe en procesos de comunicación y relación social.
- **Multimedia**: los recursos deben aprovechar las prestaciones multimedia disponibles para superar los formatos analógicos. Además del texto y la imagen, el audio, el vídeo y la animación son elementos clave que añaden una dimensión multisensorial a la información aportada pero que también permiten exponerla con una mayor riqueza de matices: descripción gráfica de procesos mediante animaciones, simulación de situaciones experimentales manipulando parámetros, entre otros.
- **Accesibilidad**: los contenidos educativos digitales deben ser accesibles. Esta accesibilidad debe garantizarse en sus tres niveles: **Genérico**: que resulte accesible al alumnado con necesidades educativas especiales; **Funcional**: que la información se presente de forma comprensible y usable por todo el alumnado a que va dirigido; y **Tecnológico**: que no sea necesario disponer de condiciones tecnológicas extraordinarias de software, equipos, dispositivos y periféricos, además que sea accesible desde cualquier sistema operativo: Windows, Mac y GNU/Linux.
- **Flexibilidad**: se refiere a la posibilidad de utilizar los contenidos en múltiples situaciones de aprendizaje: clases ordinarias, apoyos a alumnos con necesidades educativas, en horario lectivo, no lectivo, en un ordenador del aula de informática, de la biblioteca, del aula, de casa, tanto individualmente como por parejas, tríos y grupos. Esta flexibilidad también debe aludir a la posibilidad de usarlo con independencia del enfoque metodológico que ponga en práctica el docente.

- **Modularidad**: el diseño modular de un recurso multimedia debe facilitar la separación de sus objetos y su reutilización en distintos itinerarios de aprendizaje favoreciendo un mayor grado de explotación didáctica. A menudo se tiene experiencia de la existencia de recursos donde una animación concreta resulta interesante en un momento puntual mientras que el resto no tanto. El diseño modular garantizaría un acceso directo a un elemento concreto y ello aumenta sus posibilidades de uso.
- **Adaptabilidad y reusabilidad**: el diseño de recursos fácilmente personalizables por parte del profesorado permite la adaptación y reutilización en distintas situaciones. Así, por ejemplo, un cuestionario de preguntas donde sea posible modificar fácilmente las preguntas y respuestas es más reutilizable que un cuestionario cerrado.
- **Portabilidad**: los recursos digitales educativos deben ser elaborados atendiendo a estándares de desarrollo y empaquetado. De esta forma se incrementará considerablemente su difusión. Se pueden integrar con garantías y plena funcionalidad en distintos sistemas admitiendo también su uso local. A menudo se olvida que todavía actualmente existen muchos centros sin una conexión adecuada a Internet y que demandan recursos para su explotación local.
- **Interoperabilidad**: los contenidos educativos digitales deben venir acompañados de una ficha de metadatos que recoja todos los detalles de su uso didáctico, lo que facilita su catalogación en los repositorios y la posterior búsqueda por parte de terceros.

De manera general, las características antes mencionadas tienen como objetivo lograr la creación de recursos educativos de calidad que puedan ser actualizados, reutilizados y mantenidos a lo largo del tiempo, independientemente de las aplicaciones en las que sean manejados.

1.2. Origen y evolución de la interoperabilidad

En la actualidad, los repositorios trascienden las fronteras de la búsqueda y recuperación de los recursos educativos a nivel de institución, permitiendo además compartir los mismos mediante la interoperabilidad con otras aplicaciones e-learning.

La interoperabilidad puede ser definida como *"la capacidad de diferentes productos y servicios de las TIC para intercambiar y usar datos e información con el objetivo de funcionar juntos en un entorno conectado en red. En su acepción más simple, la interoperabilidad trata de asegurar que los sistemas trabajen juntos".*(9)

Otro autores se refieren a este termino como *"la garantía de independencia tecnológica y de proveedor que posibilita que dos elementos fabricados por distintas empresas tengan la capacidad de conectarse entre sí".*(10)

Partiendo de los conceptos anteriores, los autores del presente trabajo definen la interoperabilidad como: la habilidad o capacidad que tienen los sistemas o componentes de comunicar, compartir e intercambiar datos, información y conocimiento de forma precisa, efectiva y consistente, para funcionar correctamente con otros sistemas, aplicaciones y servicios.

Desde el surgimiento de la informática, la interoperabilidad ha atravesado por diferentes momentos divididos en fases o generaciones:(11)

- ✓ En la primera generación (hasta 1985), el alcance de la interoperabilidad era primordialmente departamental y casi siempre en el seno de una compañía. En esta etapa, era posible conectar directamente o en un área local bases de datos y ordenadores.
- ✓ En la segunda generación (1985-1995), con el impacto significativo de Internet, el alcance de la interoperabilidad se extiende a toda la empresa u organización e incluso a nivel inter-empresarial, conectando decenas de ordenadores y repositorios de datos.
- ✓ La tercera generación y última, la cual está enmarcada desde 1996 hasta la actualidad, se ha caracterizado por lograr mejoras significativas en la tecnología de la comunicación, así como en las infraestructuras globales de información y de distribución computacional. Asimismo, se ha logrado un alcance muy amplio relacionado con la distribución de los datos desde un único sistema global.

La comunicación entre las plataformas permite la reutilización y obsolescencia de los distintos materiales y recursos educativos con diferentes herramientas y en distintas plataformas, además de contar con un acceso y seguimiento del comportamiento del usuario. Para ello, se han definido un conjunto de estándares que facilitan la gestión de los recursos, la integración y el desarrollo tecnológico de los sistemas; que repercuten en el almacenamiento, intercambio y búsqueda de los contenidos.

1.3. Estándares para la interoperabilidad

Un estándar es *"un conjunto de directrices que orientan sobre los requisitos indispensables que debe cumplir determinado proceso, producto o servicio para alcanzar sus objetivos de calidad".*(12)

En la práctica, un estándar implica "*el reconocimiento de un problema común; la reunión de consejeros y expertos; la discusión, revisión y acuerdos respecto a una tecnología; la publicación de las*

especificaciones y el desarrollo e implementación de las especificaciones en software, que al desarrollarse de forma cíclica aseguren la interoperabilidad".(3)

Otros autores definen a un estándar como un *"grupo de reglas o normas que especifican como llevar a cabo un proceso".*(13) Dichas reglas de basan *"en definiciones, formatos o procesos que han sido aprobados por determinadas organizaciones de estandarización o aceptados de facto como tales por la industria".*(9)

Según Cañizares, un estándar regularmente proviene de una especificación y para que esta se convierta en un estándar debe pasar por un proceso complejo, donde intervienen organismos y consorcios, que se agrupan en tres niveles de trabajo: nivel de especificación, de validación y estandarización.(7)

Actualmente existen un gran número de organizaciones y consorcios que se han especializado en los estándares de interoperabilidad, con el objetivo de darle solución a los problemas que trae consigo la falta de comunicación entre las aplicaciones de un entorno e-learning.

1.3.1. Organizaciones y consorcios

En lo que respecta a las tecnologías de la información, los objetivos primordiales de los estándares consisten en lograr un ¨lenguaje común¨, que facilite la integración e interoperabilidad entre los diferentes sistemas y tecnologías, con pérdidas mínimas, tanto de contenido como de funcionalidad. La estandarización de las tecnologías aplicadas al aprendizaje, pretende posibilitar la reutilización de recursos educativos y la interoperabilidad entre sistemas heterogéneos.(14)

Las organizaciones y consorcios más importantes que trabajan en el campo de los estándares y especificaciones del e-learning, son:

Comité de Industria de Aviación

Las recomendaciones de AICC son publicadas en tres tipos de documentos: recomendaciones y guías, informes técnicos y documentos de trabajo. Dentro de dichas recomendaciones se contemplan, entre otros aspectos, la definición de requisitos de hardware y software para los ordenadores de los alumnos, los periféricos necesarios, los formatos aceptados para los elementos multimedia que componen los cursos, así como recomendaciones para las interfaces de usuario. La guía más reconocida de la AGR 010 hace referencia a la interoperabilidad de las plataformas de formación y los cursos, resolviendo los problemas de comunicación entre los mismos, solucionando también el Learning Management System (LMS) y el poder subir a un LMS un curso desarrollado por terceros.(14)

IEEE Comité para los estándares de la Tecnología del Aprendizaje

La IEEE está organizada en distintos comités, siendo el Comité para los Estándares de la Tecnología de Aprendizaje (LTSC, Learning Technologies Standards Committee) uno de ellos, este intenta facilitar el desarrollo y la reutilización de contenidos. Una de sus especificaciones más conocidas hace referencia a los Metadatos de los Objetos de Aprendizaje (LOM, Metadata for Learning Objects). El LTSC recogió el trabajo del comité de la AICC y lo completó, introduciendo la noción de metadatos y, mediante ella, una descripción más detallada que la ofrecida por la AGR 010 sobre los contenidos en un curso.(14)

Sistema de Gestión de la Información-Consorcio de Aprendizaje Global

El IMS es un consorcio formado por miembros provenientes de organizaciones educacionales y empresas tanto públicas como privadas. Las especificaciones IMS cubren un amplio rango de características que intentan hacer interoperables las diferentes plataformas, que van desde los metadatos, hasta la creación de cursos en línea para alumnos que tengan alguna discapacidad visual, auditiva, etc. Igualmente, adopta el trabajo de la IEEE. Su objetivo es la creación de especificaciones que engloben las recomendaciones de la propia IEEE y de la AICC, es decir, abarca especificaciones a través de diferentes grupos: empaquetamiento de contenidos, metadatos, interoperabilidad de preguntas y pruebas, empaquetamiento de información del alumno, secuencia simple, diseño del aprendizaje, repositorios digitales, definición de competencias, accesibilidad y gestión de grupos y alumnos.(14)

Aprendizaje Avanzado Repartido

ADL es una iniciativa auspiciada por el gobierno de los Estados Unidos para facilitar el desarrollo y la entrega de contenidos didácticos con el uso de tecnologías ya existentes. Este organismo recogió lo mejor de las anteriores iniciativas y lo refundió y mejoró creando su propia especificación de Modelo Referenciado de Objetos Compartidos (SCORM, Sharable Content Object Reference Model), es decir, SCORM integra especificaciones de AICC, IMS y IEEE y define asimismo las claves de interrelación entre estos estándares.(14)

Estas organizaciones y consorcios han dictado un gran número de estándares, los cuales se clasifican en diversas formas, aunque en esta investigación solo se estará haciendo alusión a los que tienen como propósito lograr la interoperabilidad. Las fuentes consultas proporcionan diferentes clasificaciones para los estándares, Cañizares (7) los agrupa según su arquitectura interna.

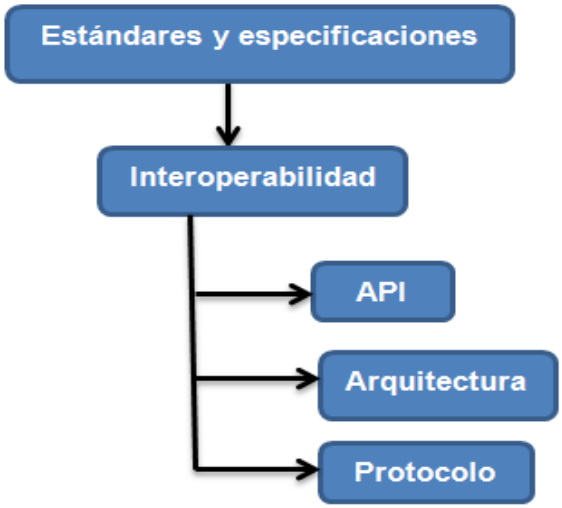


Figura 1. Clasificación de los estándares de interoperabilidad de los repositorios(7).

1.4. Estándares de interoperabilidad implementados en RHODA

A continuación se describen los estándares que fueron implementados en la versión actual de RHODA.

1.4.1. Interfaz de Consultas Simples

SQI es una Interfaz de Programación de Aplicaciones del inglés Application Programming Interface (API) para resolver las problemáticas relacionadas con las búsquedas de contenidos digitales en entornos heterogéneos, el establecimiento de sesión y la realización de consultas síncronas y asíncronas; definiendo los servicios que un repositorio puede tener disponibles para recibir y responder consultas de otros repositorios. La mejor práctica para publicar este estándar es usando servicios web bajo el Protocolo Simple de Acceso a Objetos del inglés Simple Object Access Protocol (SOAP).(15)

Para poder realizar una consulta con SQI a un repositorio, en primer lugar, la fuente de la consulta debe establecer una conexión con el destino a través de una sesión, donde los dos sistemas pueden comunicarse el uno con el otro. Las sesiones pueden ser persistentes (se mantienen indefinidamente), o no persistentes (se mantienen un periodo de tiempo limitado), estas últimas pueden ser destruidas manualmente o automáticamente si en un periodo de tiempo no se produce comunicación. Una vez que una sesión ha sido establecida, la interfaz de consultas en el destino espera que sea enviada la petición de búsqueda.

La interfaz de consultas del destino puede ser configurada, pues se mantienen válidas durante la sesión o hasta que sea configurada de otra forma diferente, y en caso de no configurarse, entonces se usa el comportamiento por defecto.

Las consultas pueden ser: síncronas o asíncronas.

- ✓ **Consultas síncronas**

En estas consultas el destino devuelve los resultados a la fuente, por lo tanto, la recuperación de los resultados es iniciada por la fuente. El proceso es el siguiente, la fuente realiza la consulta y a través de otros métodos accede después a los resultados.

- ✓ **Consultas asíncronas**

En las consultas asíncronas la transmisión de resultados es responsabilidad del destino, es decir una vez iniciada la consulta y a medida que se obtienen los resultados, estos son trasmitidos a la fuente. Para hacer posible este escenario, la fuente tiene que tener la capacidad de identificar una consulta enviada a un destino particular, por ejemplo, si la misma consulta fue enviada a diferentes destinos. También, la fuente, debe de implementar un punto de acceso para la recepción de los resultados.

Tanto en el escenario síncrono como asíncrono es importante hacer notar que es posible realizar múltiples consultas por sesión, inclusive simultáneamente.

Los métodos para la implementación del estándar SQI se encuentran en el Anexo I, donde se explican cada uno de ellos.

1.4.2. Interfaz de Publicaciones Simples

SPI es un protocolo que permite publicar recursos educativos y metadatos en un repositorio. Para ella se define una API formada por conjuntos de métodos independientes de la tecnología, lo que facilita la interoperabilidad semántica de las implementaciones de los mismos. Todos los métodos que ofrece la API requieren la creación de una sesión en el destino donde se va a publicar el recurso educativo. Asimismo, la fuente que publica el recurso siempre usa un parámetro targetSessionID para identificar la sesión. Los métodos pueden ser de dos tipos:(15)

- ✓ Orientados a la **publicación de OA** (setDataFormat, setSourceLocation, submintResourceByValue, submintResourceByReference, notifyRetrievalStatus, deleteResource).
- ✓ Orientados a la **publicación de metadatos** (setMetadataSchema, submintMetadataRecord, deleteMetadataRecord, associate, dissociate).

Los métodos antes mencionados del estándar SPI se encuentran en el Anexo II de la presente investigación, donde se brinda una explicación detallada de cada uno de ellos.

1.4.3. Iniciativa de Archivos Abiertos-Protocolo para la Recolección de Metadatos

OAI-PMH es otro de los estándares implementados en las versiones anteriores de RHODA, con la finalidad de recolectar metadatos en otros repositorios para posteriormente realizar búsquedas a partir de ellos. Este protocolo genera y promueve estándares de interoperabilidad que facilitan la difusión, intercambio y accesibilidad a documentos de diferente naturaleza. Igualmente, OAI-PMH permite almacenar en un solo lugar los metadatos y es allí en donde se realizan las diferentes consultas, el protocolo no define la creación de los metadatos, únicamente se ocupa de la gestión de la información. Utiliza transacciones del Protocolo de Transferencia de Hipertexto del inglés Hipertext Transfer Protocol (HTTP) en la transferencia de información de contenido web, donde está definida la sintaxis y la semántica que utilizan los clientes y servicios para comunicarse entre sí. El protocolo OAI está basado en un modelo cliente/servidor que transmite preguntas y respuestas entre un Proveedor de Datos (PD), y un Proveedor de Servicios (PS). El PS puede pedir al PD que le envíe metadatos según determinados criterios como por ejemplo la fecha de creación de registros, en respuesta, el PD envía un conjunto de registros codificados en formato de Lenguaje de Marcas Extensibles (XML, Extensible Markup Language).(15) Existen solamente seis peticiones que un PS puede realizar a un PD:

- ✓ **GetRecord:** se utiliza para recuperar un registro de metadatos individuales de un repositorio. Es necesario especificar el identificador del elemento del que se solicita el registro y el formato de los metadatos que deben incluirse en el registro.
- ✓ **Identify:** recupera información sobre un repositorio.
- ✓ **ListRecords:** recolecta los metadatos asociados a los registros almacenados en el repositorio. Los argumentos opcionales permiten la búsqueda selectiva de los registros basados en la colección a la que pertenecen y especificando la fecha de creación del mismo.
- ✓ **ListIdentifiers:** es una abreviatura del verbo ListRecords, devuelve solo las cabeceras de los registros, se pueden realizar búsquedas selectivas especificando la colección y la fecha en que se publicó el registro.
- ✓ **ListSets:** devuelve la estructura de las colecciones existentes en el repositorio.

- ✓ **ListMetadataFormats:** se emplea para devolver un listado de los metadatos soportados en el repositorio, se puede conocer opcionalmente el formato de un recurso especificando su identificador).

Como se mencionaba en el inicio del epígrafe, los métodos definidos por los estándares anteriormente descritos fueron implementados en forma de servicios web en versiones anteriores del repositorio. La selección de los mismos tuvo su basamento en estudios realizados en investigaciones que analizaron el comportamiento de los mismos en repositorios reconocidos a nivel internacional.

Como parte de dichas investigaciones, se definió además la arquitectura que debía presentar RHODA, de manera tal que se lograra la interoperabilidad con herramientas e-learning, teniendo en cuenta en la misma los estándares establecidos con anterioridad.

1.5. Arquitectura de RHODA

Cañizares en su tesis doctoral (7) estableció la arquitectura del repositorio RHODA para alcanzar su interoperabilidad con herramientas e-leaning. Esta consta de tres capas lógicas: servicios web, negocio y acceso a datos, las cuales se pueden observar en la figura 2.

Capa de servicios web: en la misma se publican los servicios que brinda la aplicación, para ser consumidos por un sistema externo. Estos servicios corresponden a los métodos de los estándares SQI, SPI y OAI-PMH. En esta capa se encuentran dos conectores, uno para los servicios publicados por Transferencia de Estado Representacional (REST, por sus siglas en inglés) y otro para los publicados por SOAP.

Capa de negocio: es la encargada de realizar cálculos basados en los datos de entrada o almacenado, validando los datos provenientes de la capa de servicio, así como la ejecución de algoritmos específicos para dar respuesta a estas peticiones. El módulo **Interoperabilidad**, está integrado por los componentes SPI, SQI, OAI-PMH, SESION y utiliza la librería Libtec para transformar los estándares de catalogación. Esta librería es desarrollada en la presente investigación para convertir los metadatos de los recursos que se importan en RHODA hacia el estándar LOM y de este a otros formatos durante la exportación de recursos desde RHODA.

Capa de acceso a datos: se ocupa de manejar los datos almacenados en la base de datos relacional Postgres SQL y la base de datos nativa XML eXist. Se utilizan los lenguajes de consulta SQL para base de datos relacional, apoyados por el ORM Propel y Xquery para la base de datos XML. En el caso de Apache Solr se utiliza para la generación de los índices a través de su indexador Lucene.

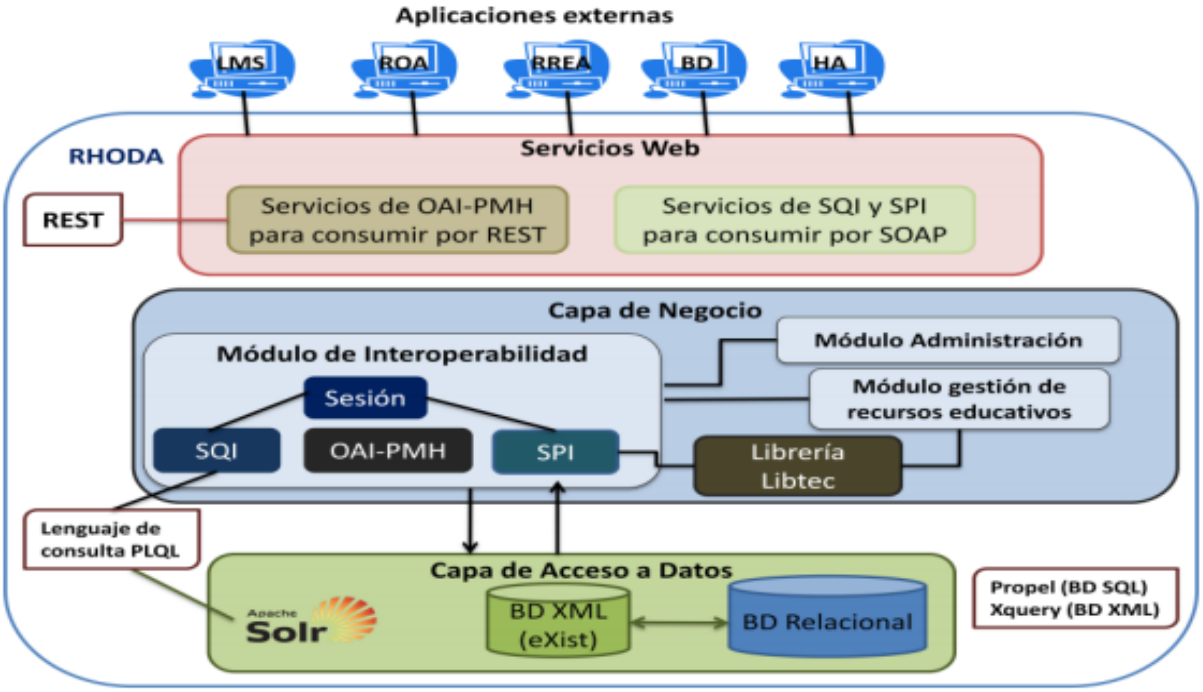


Figura 2. Interoperabilidad en RHODA.

(Fuente:(7))

1.5.1. Servicios web

Como se aprecia en la figura 2, para lograr la interoperabilidad es necesario contar con una capa de servicios, en la cual se publiquen los métodos propuestos por los estándares en forma de servicios web, de manera tal que puedan ser consumidos por otras aplicaciones.

Los servicios web son un *"conjunto de aplicaciones o tecnologías con capacidad para interoperar en la web*".(16)

Cuando una aplicación expone alguna funcionalidad, se puede acceder a ella a través de ciertas operaciones que requieren que se le pase alguna información. Una vez que las operaciones se completan, la aplicación devuelve los resultados al cliente. Esos intercambios de información necesitan un protocolo, lo cual implica que el desarrollador debe haber descrito las interfaces de acceso al servicio.(17) En la propuesta de arquitectura antes mencionada, se definió la utilización de los protocolos SOAP para el

consumo de los servicios correspondientes a los estándares SQI y SPI, y REST para los servicios correspondientes a OAI-PMH.

Transferencia de Estado Representacional

REST es utilizado en la actualidad para describir cualquier interfaz web simple que utiliza XML y HTTP, sin las abstracciones adicionales de los protocolos basados en patrones de intercambio de mensajes como el protocolo de servicios web SOAP.(18) Los sistemas que siguen los principios REST son conocidos como RESTful.

Una implementación concreta de un servicio web REST sigue cuatro principios de diseño fundamentales:(19)

- Un protocolo cliente/servidor sin estado: cada mensaje HTTP contiene toda la información necesaria para comprender la petición. Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes. Sin embargo, en la práctica, muchas aplicaciones basadas en HTTP utilizan cookies y otros mecanismos para mantener el estado de la sesión (algunas de estas prácticas, como la reescritura de URL, no son permitidas por REST).
- Expone URL con forma de directorios.
- En REST sobre HTTP, se utilizan los siguientes métodos de manera explícita:
    - ✓ POST: crea nuevos recursos, como retorno, se ofrece el ID automáticamente creado.
    - ✓ GET: lista un recurso.
    - ✓ PUT: reemplaza un recurso con otro (útil para el update).
    - ✓ DELETE: elimina un recurso.
- El uso de hipermedios, tanto para la información como para las transiciones de estado de la aplicación, permite navegar de un recurso REST a muchos otros, simplemente siguiendo enlaces sin requerir el uso de registros u otra infraestructura adicional.

Existe un enorme número de aplicaciones REST en la red, tal es el caso de Google, Facebook, Amazon, eBay, Yahoo, Microsoft, entre otros. Un ejemplo de servicio web tipo RESTful en repositorios son los provistos por el repositorio Recursos para la Educación Multimedia para el Aprendizaje y Enseñanza en Línea o Multimedia Education Resource for Learning and Online Teaching (MERLOT), para la recolección de metadatos con el protocolo OAI-PMH.

Protocolo Simple de Acceso a Objetos

SOAP es un protocolo de comunicaciones que permite el envío de mensajes XML, lo cual es la base sobre la que se sustentan los servicios web en la mayoría de las aplicaciones empresariales. Los mensajes SOAP son unidireccionales, aunque todas las aplicaciones pueden participar como emisoras o receptoras indistintamente. Este pueden servir para muchos propósitos: petición/respuesta, mensajería asíncrona y notificaciones. Es un protocolo de alto nivel, que sólo define la estructura del mensaje y algunas reglas básicas de procesamiento de éste, siendo completamente independiente del protocolo de transporte. Ello posibilita que los mensajes SOAP puedan ser intercambiados a través de los protocolos HTTP, SMTP, JMS, entre otros. Un mensaje SOAP consta de un sobre (envelope) obligatorio, una cabecera (header) opcional, y un cuerpo (body) obligatorio.(18)

Para acceder a los servicios web publicados por una aplicación utilizando el protocolo SOAP es necesario conocer la URL del servidor, el módulo (package) que da servicio, y el nombre de los métodos para hacer las llamadas. En SOAP no existe una forma estándar de hacer especificaciones. No obstante, puede ser utilizado el Lenguaje de Descripción de Sevicios Web (WSDL, por sus siglas en inglés).

Lenguaje de Descripción de Servicios Web

WSDL es un lenguaje basado en XML, creado para definir la interfaz de los servicios. Describe la forma de comunicación, es decir, los requisitos del protocolo y los formatos de los mensajes necesarios para interactuar con los servicios listados en su catálogo. Las operaciones y mensajes que soporta se describen en abstracto y se mezclan después al protocolo concreto de red y al formato del mensaje. WSDL se usa a menudo en combinación con SOAP y XML Schema. Un programa cliente que se conecta a un servicio web puede leer el WSDL para determinar qué funciones están disponibles en el servidor. Los tipos de datos especiales se incluyen en el archivo WSDL en forma de XML Schema. El cliente puede usar SOAP para hacer la llamada a una de las funciones listadas en el WSDL.(17)

Descripción, Descubrimiento e Integración Universal

En la arquitectura propuesta se definió además, la utilización de un registro para publicar los diferentes servicios implementados, a los cuales se puede acceder mediante la utilización de los protocolos descritos anteriormente. De manera general, este registro es conocido internacionalmente como UDDI, es decir, Descripción, Descubrimiento e Integración Universal del inglés Universal Description, Discovery, and Integration y es utilizado para almacenar de forma estructurada la información sobre empresas y los

servicios que éstas ofrecen. Para facilitar el descubrimiento de los servicios publicados en una UDDI se pueden emplear sistemas taxonómicos, que permiten clasificarlos mismo en un conjunto de categorías.(20)

1.6. Metodología de desarrollo de software, tecnologías y herramientas a utilizar

Como parte del marco de trabajo definido en el centro FORTES, se establecieron un conjunto de tecnologías y herramientas con el objetivo de homogeneizar el proceso de desarrollo de software. A continuación se hará referencia a las tecnologías y herramientas, así como la metodología de desarrollo de software a utilizar para dar solución al problema planteado en la presente investigación.

1.6.1. Metodología de desarrollo de software. Proceso Unificado de Software

En la realización de proyectos de software, es necesario basarse en una metodología que ayude a organizar y planificar todo el proceso de desarrollo, para de este modo obtener un producto de calidad y lograr que los clientes se sientan satisfechos con el resultado. Un proceso de desarrollo de software, es la definición de un conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de artefactos que explican los pasos necesarios para terminar el proyecto, y tiene la misión de transformar los requerimientos del usuario en un producto de software. Un proceso define ¨quién¨ está haciendo ¨qué¨, ¨cuándo¨ y ¨cómo¨ para alcanzar un determinado objeto.(21)

El Proceso Unificado de Software o Rational Undefined Proccess (RUP) es una metodología flexible, y fácil de adaptarse a las necesidades de cualquier proyecto, esta brinda al equipo de desarrollo guías consistentes y personalizadas del proceso. El mismo, en conjunto con el Lenguaje Unificado de Modelado del inglés Unified Modeling Language (UML), constituye la metodología estándar utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP utiliza UML para definir los modelos de software y puede definirse como un modelo dirigido por casos de uso, iterativa, incremental y centrado en la arquitectura.(22)

1.6.2. Tecnologías y herramientas

Herramienta de modelado: Visual Paradigm for UML v8.0

Visual Paradigm for UML es una herramienta de Ingeniería de Software Asistida por Computadoras (CASE, por sus siglas en inglés) que soporta el modelado mediante UML y proporciona asistencia a los

analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software. Las ventajas proporcionadas por el Visual Paradimg for UML son:(23)

- ✓ Dibujo: permite la estandarización de la documentación, ya que la misma se ajusta al estándar soportado por la herramienta.
- ✓ Corrección sintáctica: controla que el modelado con UML sea correcto.
- ✓ Coherencia entre diagramas: al disponer de un repositorio común, es posible visualizar el mismo elemento en varios diagramas, evitando duplicidades.
- ✓ Integración con otras aplicaciones: permite integrarse con otras aplicaciones, como herramientas ofimáticas, lo cual aumenta la productividad.
- ✓ Trabajo multiusuario: permite el trabajo en grupo, proporcionando herramientas de compartición de trabajo.
- ✓ Reutilización: facilita la reutilización, ya que constituye una herramienta centralizada donde se encuentran los modelos utilizados para otros proyectos.
- ✓ Generación de código: permite generar código de forma automática, reduciendo los tiempos de desarrollo y evitando errores en la codificación del software.
- ✓ Generación de informes: permite generar diversos informes a partir de la información introducida en la herramienta.

Lenguaje unificado de modelado: UML v2.1

UML es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. Es un lenguaje expresivo, claro y uniforme, que no garantiza el éxito de los proyectos pero si mejora sustancialmente el desarrollo de los mismos, al permitir una nueva y fuerte integración entre las herramientas, los procesos y los dominios.(24)

Lenguaje para las hojas de estilos: CSS3

Hojas de Estilos en Cascada o Cascading Style Sheets (CSS), es un lenguaje para definir el estilo o la apariencia de las páginas web, escritas con HTML o documentos XML. Se creó para separar el contenido de la forma, permitiendo a los diseñadores mantener un control mucho más preciso sobre la apariencia de las páginas. Dicho objetivo se cumplió con las primeras especificaciones del lenguaje, sin embargo, el objetivo de ofrecer un control total a los diseñadores sobre los elementos de la página ha sido más difícil de cubrir. Posibilita la incorporación de nuevos mecanismos para mantener un mayor control sobre el

estilo con el que se muestran los elementos de las páginas, sin tener que recurrir a trucos, que a menudo complicaban el código de las web.(25)

Lenguaje de programación del lado del cliente: JavaScript

JavaScript es un lenguaje de programación del lado del cliente, utilizado para crear pequeños programas que luego son insertados en una página web y en programas más grandes. Este lenguaje está centrado en describir objetos, escribir funciones que respondan a movimientos del mouse, aperturas, utilización de teclas, cargas de páginas, entre otros. Para el desarrollo de la solución se utiliza JavaScript en su versión 1.8, esto se determina según el navegador utilizado que soporte algunas de las versiones existentes. Es necesario hacer uso de este lenguaje para crear diferentes efectos e interactuar con los usuarios a la hora de realizar algunas acciones no válidas que puedan atentar contra la seguridad del sistema, como la entrada de datos en los formularios.(26)

Lenguaje de programación del lado del servidor: PHP v5.4

Preprocesador de Hipertexto (PHP, por sus siglas en inglés) es un lenguaje multiplataforma, está orientado al desarrollo de aplicaciones web manteniendo acceso a información almacenada en una base de datos. El código fuente escrito en PHP es transparente al navegador y al usuario, haciendo la programación segura y confiable.(27) Presenta capacidad de conexión con la mayoría de los motores de base de datos, principalmente con MySQL y PostgreSQL.

Posee una amplia documentación en su página oficial, donde todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda. Es libre, por lo que se presenta como una alternativa de fácil acceso para todos. No requiere definición de tipos de variables aunque las mismas se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.

El principal objetivo de PHP5 ha sido mejorar los mecanismos de POO[4] para solucionar las carencias de las versiones anteriores. Un paso necesario para conseguir que PHP sea un lenguaje apto para todo tipo de aplicaciones y entornos, incluso los más exigentes.(28)

[4] **POO:** La Programación Orientada a Objetos trata de modelar los procesos de programación de una manera cercana a la realidad: tratando a cada componente de un programa como un objeto con sus características y funcionalidades.

Lenguaje de Marcas de Hipertexto: HTML5

El HTML es un lenguaje muy sencillo que permite describir hipertexto, es decir, texto presentado de forma estructurada, agradable, con enlaces que conducen a otros documentos o fuentes de información relacionadas y con inserciones multimedia; es el lenguaje que se utiliza para presentar información en la web.(29)

HTML se basa en especificar en el texto la estructura lógica del contenido (títulos, párrafos de texto normal, enumeraciones, definiciones y citas), así como los diferentes efectos que se quieren proporcionar (cursiva, negrita, o un gráfico determinado), para que la presentación final la realice un programa especializado como los navegadores Internet Explorer o Mozilla Firefox.

Este lenguaje indica al navegador dónde colocar el texto, las imágenes y los videos, atendiendo a la disposición deseada para colocarlos en la página. Es decir, una serie de etiquetas que se utilizan para definir la forma o estilo que se quiera aplicar al documento.

HTML5 lleva a un nivel más alto que el código HTML4 del lenguaje de marcado, ya que será capaz de controlar los eventos y las iteraciones con el usuario. La diferencia esencial no está en el lenguaje, sino en la incorporación de etiquetas nuevas en comparación con HTML4, además en que no requiere un tipo de documento específico. En resumen, HTML5 conduce a una fusión entre JavaScript como lenguaje de programación, HTML como modelo semántico y CSS3 que es la evolución del CSS como el lenguaje de los estilos, que se dedica a dar un mejor aspecto a los proyectos.(30)

Sistema gestor de bases de datos: PostgreSQL v9.2

PostgreSQL es un sistema gestor de base de datos relacional orientado a objetos, libre y de código abierto, posee una extensa documentación, incluyendo un manual completo y algunos ejemplos prácticos. Se ejecuta en la mayoría de los sistemas operativos más utilizados en el mundo incluyendo Linux, varias versiones de UNIX y en Windows. A continuación se enuncian las características del gestor PostgreSQL:(31)

1. Es altamente adaptable a las necesidades del cliente.
2. Presenta un soporte nativo para los lenguajes más populares del medio: PHP, C, Perl, Python.
3. Soporta todas las características de una base de datos profesional (triggers, store procedures, funciones, secuencias, relaciones, reglas, tipos de datos definidos por usuarios).
4. Permite realizar transacciones, subselects, disparadores, vistas, claves foráneas.

Servidor web: Apache v2.2

Servidor web gratuito desarrollado por el Apache Server Project, fiable, eficiente y fácilmente extensible con código fuente abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras. (32)
Ventajas:

- ✓ Su licencia, que permite el uso comercial y no comercial del Apache.
- ✓ Una talentosa comunidad de desarrolladores.
- ✓ Su arquitectura modular, que permite a los usuarios adicionar funcionalidades a sus ambientes específicos.
- ✓ Portabilidad, pues trabaja sobre todas las versiones recientes de Unix y Linux, Windows, mainframes.
- ✓ Es robusto y seguro.
- ✓ Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.
- ✓ Tiene un alto nivel de configuración en la creación y gestión de logs".

Framework de desarrollo: Symfony v2.3.7

Symfony es un framework diseñado para optimizar el desarrollo de las aplicaciones web basado en el patrón Modelo Vista Controlador. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja.(33) También automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.
Symfony está desarrollado completamente en PHP 5.3, y ha sido probado en numerosos proyectos. Es compatible con la mayoría de los gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server, y se puede ejecutar en sistemas operativos como Linux y Windows.

Entorno Integrado de Desarrollo: NetBeans v7.3

NetBeans es un entorno de desarrollo libre, gratuito y sin restricciones de uso, creado principalmente para el lenguaje de programación Java. Permite que las aplicaciones sean desarrollas a partir de un conjunto de componentes de software llamados módulos[5].(34)

[5] **Módulo:** es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (archivo de manifiesto) que lo identifica.

La integración de NetBeans con Symfony permite desarrollar aplicaciones de forma más sencilla y productiva. En primer lugar, es posible crear nuevos proyectos y aplicaciones directamente desde el Ambiente Integrado de Desarrollo (IDE por sus siglas en inglés).Posibilita además, ejecutar todas las tareas de Symfony, incluso pasándole argumentos y opciones, visualizando el resultado sin necesidad de utilizar una consola de comandos externa.(35)

Framework de CSS: Bootstrap v3.0

Bootstrap es una colección de herramientas de software libre para la creación de sitios y aplicaciones web. Contiene plantillas de diseño basadas en HTML y CSS con tipografías, formularios, botones, gráficos, barras de navegación y demás componentes de interfaz, así como extensiones opcionales de JavaScript. Proporciona un conjunto de hojas de estilo que proveen definiciones básicas de estilo para todos los componentes de HTML. Esto otorga una uniformidad independiente del navegador, lo que da una apariencia moderna para el formato de los elementos de texto, tablas y formularios.(36)

Conclusiones parciales

El estudio del estado del arte sobre los estándares de interoperabilidad desarrollados en la versión 2.2 de RHODA, permitió entender el funcionamiento de los mismos.

El estudio de las tecnologías, herramientas, lenguajes y metodología posibilitó la adquisición de conocimientos y experiencias en el trabajo de las mismas.

Capítulo 2. Propuesta de solución

Introducción del capítulo

El proceso de análisis y diseño de sistemas es una guía que permite estructurar el proceso de desarrollo de sistemas de información. El análisis ofrece las descripciones de los servicios y restricciones del sistema, es decir, los requerimientos del mismo. Por medio de la especificación de los requerimientos se obtiene un análisis detallado de las necesidades y funciones del sistema, en otras palabras, se decide qué es lo que se quiere hacer.

En el diseño se analizan los requerimientos refinándolos y estructurándolos, dando forma al sistema de manera que proporcione vida a todos los requisitos, incluidos todos los no funcionales que incorpora el mismo.(37) De esta forma, la disciplina de análisis y diseño específica y describe cómo se va a implementar el sistema, es decir, los diseñadores de software determinan la mejor solución técnica a partir de los requerimientos, de la arquitectura del sistema más adecuada y el diseño detallado para las actividades de implementación. En este capítulo, se describe la solución que se propone a partir de los diagramas de clases del análisis y se representan los diagramas de clases del diseño, que reflejan una vista interna del sistema.

2.1. Modelo Conceptual

Un modelo conceptual es un conjunto de conceptos y reglas destinados a representar de forma global los aspectos lógicos de los diferentes tipos de elementos existentes en la realidad que está siendo analizada. Es una representación figurada de una experiencia empírica, que tiene como objetivo ayudar a comprender la realidad.

Para una mejor comprensión de los estándares a implementar, se realizó un modelo conceptual que evidencia el funcionamiento de los mismos. A continuación se describen las clases que intervienen en el modelo realizado:

Fuente: repositorio que brinda los recursos educativos.

Destino: repositorio que almacena los recursos educativos.

Sesión: vía de comunicación establecida entre la fuente y el destino.

Persistentes: las sesiones se mantienen indefinidamente,

No Persistentes: las sesiones se mantienen un periodo de tiempo limitado.

Petición de búsqueda: vía mediante la cual se realiza la búsqueda de los recursos educativos en el destino.

API: conjunto de funciones o métodos que permite al programador acceder a servicios de una aplicación a través del uso de un lenguaje de programación.

Métodos: funciones que ofrece la API.

Publicación OA: cualquier recurso digital que puede ser reutilizado para apoyar el aprendizaje. Estos son publicados a través de los métodos que ofrece la API del estándar.

Publicación Metadato: información descriptiva sobre el contexto, calidad, condición o características de un recurso, dato u objeto que tiene la finalidad de facilitar su recuperación, autentificación, evaluación, preservación e interoperabilidad. Estos son publicados a través de los métodos que ofrece la API del estándar.

PD: son los sistemas que soportan el estándar OAI-PMH como un medio para exponer los metadatos correspondientes a los recursos educativos.

PS: son entidades que recolectan metadatos de los PD.

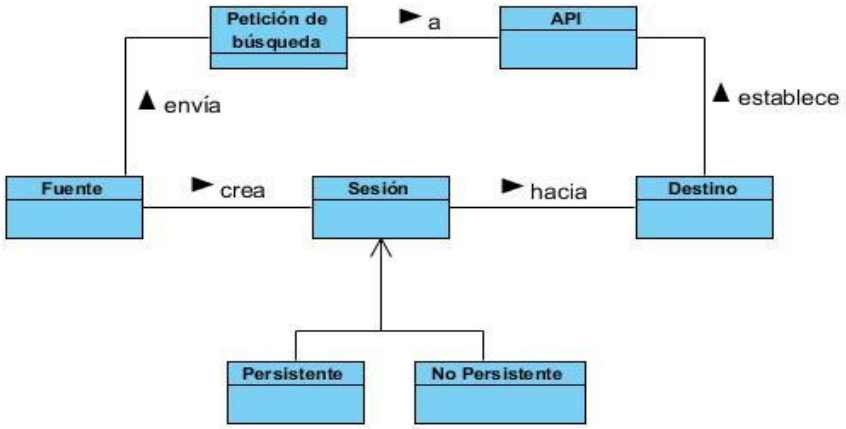


Figura 3. Modelo conceptual del estándar SQI.

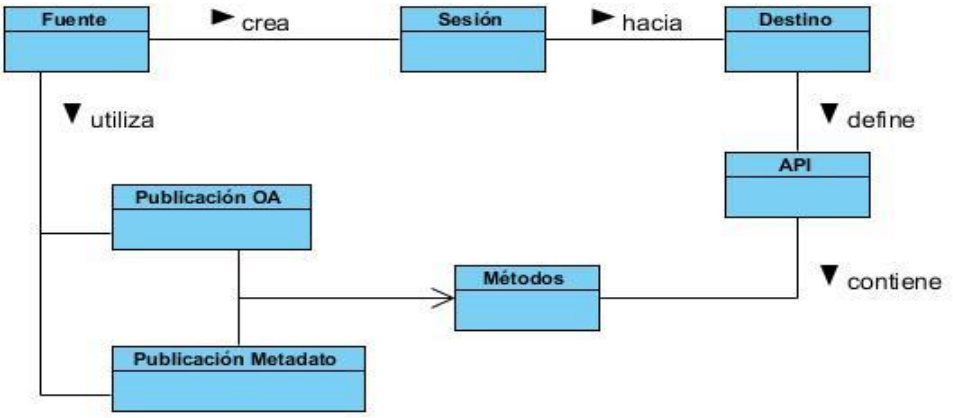


Figura 4. Modelo conceptual del estándar SPI.

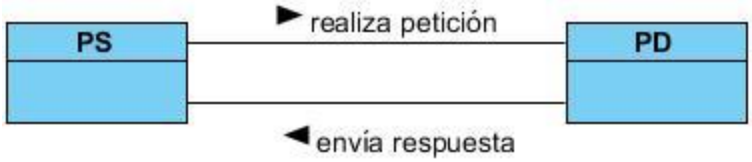


Figura 5. Modelo conceptual del estándar OAI-PMH.

2.2. Descripción del sistema

El módulo tiene como principal objetivo la implementación de los estándares OAI-PMH para la recolección de metadatos, SQI para la realización de búsquedas federadas en otros repositorios y SPI para permitir la publicación en RHODA de recursos educativos. De igual modo, el módulo debe permitir la configuración de los repositorios a partir de los cuales se podrán recolectar los metadatos y realizar las búsquedas federadas. Por último, deberá posibilitar la actualización tanto automática como manual de los metadatos que sean recolectados mediante el estándar OAI-PMH.

A continuación se definen por cada estándar los métodos implementados:

Servicios del componente SQI:

- ✓ setQueryLanguage ($targetSessionID, $queryLanguageID): permite a la fuente controlar la sintaxis usada en las instrucciones de la consulta por identificación del lenguaje de consulta.
- ✓ setMaxQueryResults ($targetSessionID, $maxQueryResults): define el número máximo que una consulta puede producir, especificándose como un porcentaje. El parámetro maxQueryResults debe ser cero o más.

- ✓ setMaxDuration ($targetSessionID, $maxDuration): permite a la fuente configurar un tiempo de espera en caso de que la consulta opera en interfaz de consultas asíncronas. El parámetro que controla el tiempo de espera se expresa en milisegundos.
- ✓ setResultsFormat ($targetSessionID, $resultsFormat): permite a la fuente controlar el formato de los resultados retornados por el destino, el repositorio soportara LOM como formato para devolver los resultados.
- ✓ setResultsSetSize ($targetSessionID, $resultsSetSize): define el máximo número de resultados, que por defecto son 25. Si toma el valor de cero, entonces se entiende que quiere recuperar todos los resultados.
- ✓ synchronousQuery ($targetSessionID, $queryStatement): envía una consulta al destino a través del parámetro queryStatement. Dentro de una sesión identificada por el targetSessionID. El método retorna un conjunto de instancias de metadatos acordes con la consulta realizada.
- ✓ GetTotalResultsCount($targetSessionID, $queryStatement): retorna el número total de resultados existentes de una consulta.

Servicios del componente Sesión

- ✓ createSession($userID, $password): permite crear una sesión asociada a un usuario, para lo cual requiere del usuario y la contraseña como parámetros, retorna un identificador de sesión si el usuario es autenticado.
- ✓ createAnonymousSession(): permite crear una sesión no asociada a un usuario, y puede ser usado en un sistema que permite a todo las personas crear consultas.
- ✓ destroySession($sessionID): permite a la fuente que inició la sesión finalizar la sesión, para lo que es necesario eliminar el identificador de sesión sessionID.

Servicios del componente SPI

- ✓ setDataFormat ($sessionId, $setdataFormatID): permite indicar el tipo de objetos que se va a almacenar en un repositorio. Dispone del parámetro setDataFormatID, que establece el formato de datos para objetos compuestos como los paquetes SCORM, AICC o IMS-QTI, el repositorio soporta SCORM 1.2 y SCORM 1.3.
- ✓ setSourceLocation ($sessionId, $sourceLocation): se usa antes de que un objeto de aprendizaje sea enviado en modo por referencia. Dispone del parámetro sourceLocation que permite resolver

la localización de la fuente del método notifyRetrievalStatus y así enviar un mensaje de reconocimiento.

- ✓ submitResourceByReference ($sessionId, $reference): permite publicar un OA en el repositorio por referencia, el parámetro reference posee la referencia del OA en el sistema externo.
- ✓ submitResourceByValue ($sessionId, $data): permite publicar un OA en el repositorio por valor, se almacenera y se retornará un identificador que corresponde a dicho OA.
- ✓ setMetadataSchema ($sessionId, $metadataSchema): permite a la fuente controlar el esquema de los metadatos que será publicado en el destino, y al destino le permite validar las instancias de metadatos, actualmente RHODA soporta LOM-ES como esquema de metadatos.
- ✓ submitMetadataRecord ($sessionId, $metadataRecord): permite enviar para publicar una instancia de metadatos. El repositorio genera un identificador que será retornado y el cual identificará la instancia de metadatos.
- ✓ associate ($sessionId, $resourceIdentifier, $metadataIdentifier): permite asociar una instancia de metadatos con un OA publicado por referencia.
- ✓ dissociate ($sessionId, $resourceIdentifier, $metadataIdentifier): permite disociar una instancia de metadatos con un OA publicado por referencia.

Servicios del componente OAI-PMH

- ✓ Identify: se utiliza para recuperar información sobre un repositorio.
  Ej. http://10.55.10.236:5801/app.php/interoperability/oai?verb=Identify
- ✓ ListRecords: se utiliza para recolectar los metadatos asociados a los registros almacenados en el repositorio. Los argumentos opcionales permiten la búsqueda selectiva de los registros basados en la colección a la que pertenecen y especificando la fecha de creación del mismo.
  Parámetros:
  - ➢ metadataPrefix (requerido)
  - ➢ from (opcional) until (opcional)
  - ➢ set (opcional)
  - ➢ resumptionToken
  - ➢ (exclusivo)

  Ej.
  http://10.55.10.236:5801/app.php/interoperability/oai?verb=ListRecords&metadataPrefix=oai_dc

- ✓ ListIdentifiers: abreviatura del verbo ListRecords, devuelve solo las cabeceras de los registros, se pueden realizar búsquedas selectivas especificando la colección y la fecha en que se publicó el registro.

  Parámetros:

  - metadataPrefix (requerido)
  - from (opcional) until (opcional)
  - set (opcional)
  - resumptionToken (exclusivo)

  Ej. http://10.55.10.236:5801/app.php/interoperability/oai?verb=ListIdentifiers&metadataPrefix=oai_dc
- ✓ ListSets: se utiliza para devolver la estructura de las colecciones existentes en el repositorio.

  Parámetros:

  - resumptionToken (exclusivo)

    Ej. http://10.55.10.236:5801/app.php/interoperability/oai?verb=ListSets
- ✓ ListMetadataFormats: se utiliza para devolver un listado de los metadatos soportados en el repositorio, se puede conocer opcionalmente el formato de un recurso especificando su identificador.

  Parámetros:

  - identifier (opcional)

    Ej.http://10.55.10.236:5801/app.php/interoperability/oai?verb=ListMetadataFormats&metadataPrefix=oai_dc

A continuación se muestra un ejemplo ideal (sin error) de un cliente para consumir los servicios que brinda el repositorio, donde se realiza todo el proceso correctamente.

```php
<?php

//url del servicio
$servicio = "http://10.55.10.236:5800/ws/RHODA";

//parámetros de la llamada
$user = "Tesis";
$password = "tuclave";

//crea una instancia de la clase SoapClient de php.
$client = new SoapClient($servicio);

//llamamos al método para crear la sesión.
$targetSessionID = $client->createSession($user, $password);

//configurar la consulta.
$languaje = $client->setQueryLanguage($targetSessionID, "PLQL");
$maxQueryResults = $client->setMaxQueryResults($targetSessionID, "0-100");
$resultFormat = $client->setResultsFormat($targetSessionID, "IMS-MD");
$resultsSetSize = $client->setResultsSetSize($targetSessionID, "0-25");

//despues de configurar la consulta hacemos la consulta.
$result = $client->synchronousQuery($targetSessionID, "consulta");

//destruir la sesión.
$session = $client->destroySession($targetSessionID);

?>
```

Figura 6. Cliente SQI.

```php
<?php

//url del servicio.
$servicio = "http://10.55.10.236:5800/ws/RHODA";

//parámetros de la llamada.
$user = "Tesis";
$password = "tuclave";

//crea una instancia de la clase SoapClient de php.
$client = new SoapClient($servicio);

//llamamos al método para crear la sesión.
$targetSessionID = $client->createSession($user, $password);

//configurar la publicación
$dataFormat = $client->setDataFormatID($targetSessionID, $setdataFormatID);
$metadataSchema= $client->setMetadataSchema($targetSessionID , $metadataSchema);

//se envía el recurso por valor
$resource = $client->submitResourceByValue($data, $targetSessionID, $identifier);

//destruir la sesión
$session = $client->destroySession($targetSessionID);

?>
```

Figura 7. Cliente SPI.

```php
<?php
require_once './Client/OAIPMHClient.php';

$baseUrl = "http://publicaciones.uci.cu/index.php/SC/oai";
//se crea una instacia del cliente pasandole la url_base
$client = new OAIPMHClient($baseUrl, false);

//se hace la petición con el método que desehamos ejecutar, junto con le prefijo de metadato.
$response = $client->ListRecords('oai_dc');

?>
```

Figura 8. Cliente OAI-PMH.

2.3. Modificaciones realizadas a la arquitectura de RHODA

Debido a que la arquitectura descrita en el capítulo anterior fue pensada teniendo en cuenta el marco de trabajo de Symfony 1.4, se hace necesario realizar cambios en la misma. Primeramente, se modifica la capa de acceso a los datos, descartando la inclusión de la base de datos xml eXist, debido a que en la propuesta de solución se define que toda la información manejada en el sistema sea almacenada utilizando el gestor Postgresql. De igual modo, se sustituye la utilización del ORM Propel por Doctrine. En la capa del negocio, se sustituye el módulo para la gestión de OA, ya que el mismo deberá permitir el manejo de cualquier tipo de recurso educativo. En la figura 9, se presenta la propuesta de arquitectura que incluye los cambios anteriormente mencionados.

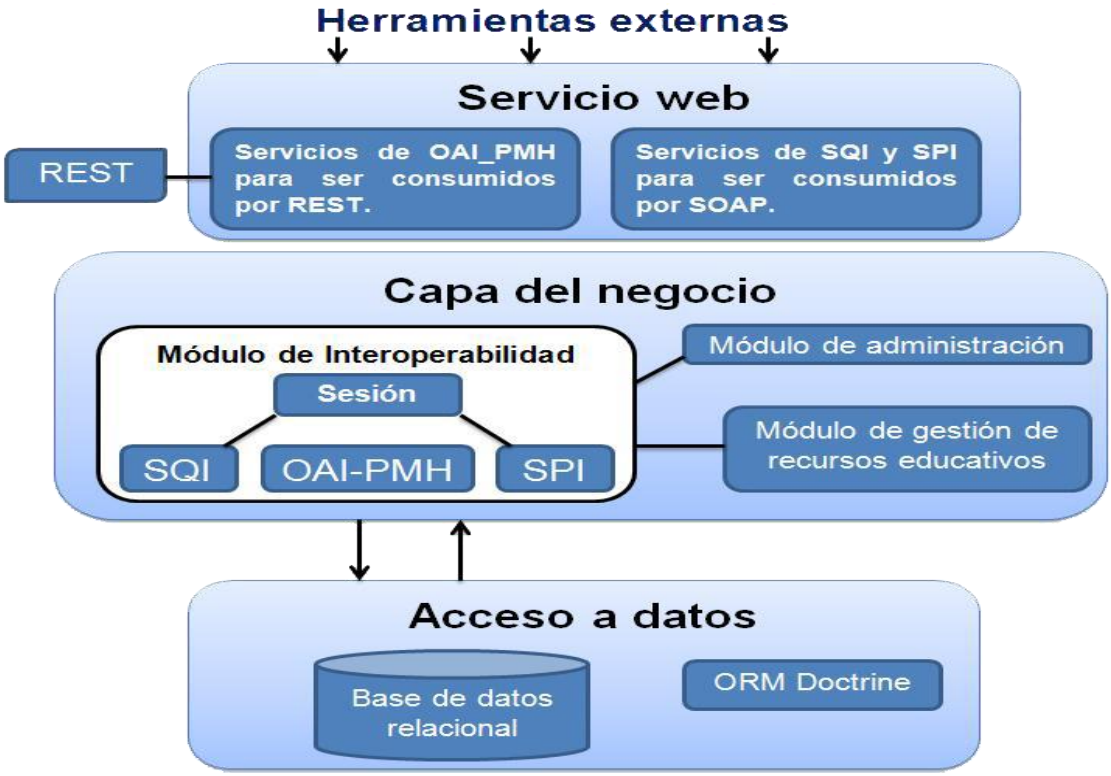


Figura 9. Propuesta de la arquitectura de RHODA.

2.4. Especificación de requisitos

El propósito general de la captura de requisitos es obtener una descripción correcta de lo que debe hacer el sistema y delimitar su alcance. Los requisitos juegan un papel importante durante el ciclo de vida de un proyecto, ya que establece las características que debe poseer el sistema, así como las condiciones que debe cumplir. Los requisitos se clasifican en funcionales y no funcionales.

2.4.1. Requisitos funcionales

Los Requisitos Funcionales (RF) son aquellos que describen qué debe hacer el sistema o sea especifican el comportamiento de entrada y salida del sistema y surgen de la razón fundamental de la existencia del producto, desde el punto de vista de las necesidades del usuario.(38) A continuación se muestran los requisitos funcionales del sistema.

RF-1. Implementar el estándar de interoperabilidad OAI-PMH.

Incorporar el estándar OAI-PMH al repositorio.

RF-2. Gestionar repositorios para recolectar metadatos con el estándar OAI-PMH.

Permite al administrador adicionar, editar y eliminar los repositorios para la búsqueda por metadatos recolectados, así como definir la periodicidad de actualización de los catálogos por metadatos recolectados.

RF-2.1. Adicionar repositorio para recolectar metadatos.

Permite al administrador introducir datos de los repositorios.

RF-2.2. Editar repositorio para recolectar metadatos.

Permite al administrador editar datos de los repositorios.

RF-2.3. Eliminar repositorio para recolectar metadatos.

Permite al administrador eliminar los repositorios a partir de los cuales se recolectarán metadatos para la búsqueda por metadatos recolectados.

RF-2.4. Listar repositorios para recolectar metadatos.

Permite al administrador visualizar el listado de los repositorios.

RF-3. Definir la periodicidad de actualización del catálogo de metadatos.

Permite al administrador definir cada cuántos días se actualizará el catálogo de metadatos.

RF-4. Actualizar el catálogo de metadatos de forma manual.

Permite al administrador actualizar el catálogo de metadatos manualmente.

RF-5. Actualizar el catálogo de metadatos de forma automática.

Permite al administrador actualizar el catálogo de metadatos automáticamente.

RF-7. Ofrecer un catálogo de metadatos a los proveedores de servicios.

Permite al administrador ofrecer un catálogo de metadatos.

RF-8. Implementar el estándar de interoperabilidad SPI.

Incorporar el estándar SPI al sistema.

RF-9. Almacenar un recurso educativo en el repositorio mediante el estándar SPI.

Permite el almacenamiento de recursos educativos en el repositorio desde otros sistemas.

RF-10. Notificar al administrador al ocurrir las publicaciones y actualizaciones automáticas.

Permite al administrador recibir notificaciones que indiquen las publicaciones automáticas en su bandeja de entrada.

RF-11. Implementar el estándar de interoperabilidad SQI.

Incorporar el estándar SQI al sistema.

RF-12. Gestionar los repositorios para realizar búsquedas federadas con el estándar SQI.

Permite al administrador adicionar, editar y eliminar los repositorios para las búsquedas federadas.

RF-12.1. Adicionar repositorio para realizar búsqueda federada.

Permite al administrador introducir los repositorios a partir de los cuales se realizarán las búsquedas federadas.

RF-12.2. Editar repositorio para realizar búsqueda federada.

Permite al administrador editar los repositorios a partir de los cuales se realizarán las búsquedas federadas.

RF-12.3. Eliminar repositorio para realizar búsqueda federada.

Permite al administrador eliminar los repositorios a partir de los cuales se realizarán las búsquedas federadas.

RF-12.4. Listar los repositorios para realizar búsquedas federadas.

Permite al administrador visualizar el listado de repositorios a partir de los cuales se realizarán las búsquedas federadas.

RF-13. Gestionar los formatos de catalogación soportados por el repositorio.

Permite al administrador adicionar, editar y eliminar los formatos de catalogación soportados por el repositorio.

RF-13.1. Adicionar formato de catalogación.

Permite al administrador adicionar un formato de catalogación.

RF-13.2. Editar formato de catalogación.

Permite al administrador editar un formato de catalogación.

RF-13.3. Eliminar formato de catalogación.

Permite al administrador eliminar un formato de catalogación.

RF-13.4. Listar formatos de catalogación.

Permite al administrador visualizar el listado de los formatos de catalogación soportados por el sistema.

RF-14. Permitir la realización de búsquedas en el repositorio a través del estándar SQI.

Permite que otros repositorios puedan realizar búsquedas en RHODA.

2.3.2. Requisitos no funcionales

Los Requisitos No Funcionales (RnF) definen las restricciones del repositorio como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que utiliza en las interfaces del repositorio.(38) A continuación se definen cada uno de los RnF que presenta el repositorio.

- ✓ **Usabilidad**

RnF 1. Tipo de aplicación informática.

Tipo de aplicación informática: aplicación WEB.

RnF 2. Ambiente

Se debe contar con un servidor de Bases de Datos Relacional PostgreSQL y un Servidor de Aplicaciones, cada uno con 2 GB de memoria RAM, un microprocesador dual-core a 2.0 o superior y 160 GB de disco duro en el caso del Servidor de Bases de Datos. El acceso a la aplicación se realizará desde varias estaciones clientes, las que deberán tener más de 512 MB de memoria RAM y un navegador actual, preferentemente Mozilla Firefox v15 o superior y una tarjeta de red. El tiempo de respuesta de la aplicación no debe exceder los 5 segundos para cada petición realizada.

- ✓ **Confiabilidad**

RnF 3. Respuesta ante inactividad del usuario

Ante la inactividad de un usuario, el sistema deberá cerrar la sesión después de transcurridos 10 minutos, mostrando un mensaje de información.

RnF4. Respuesta ante ocurrencia de una excepción

Cuando ocurre una excepción el sistema debe mostrar un mensaje de error que indique el posible error en un formato entendible para el usuario, sin llegar a especificaciones técnicas que puedan constituir una brecha en la seguridad del sistema.

- ✓ **Eficiencia**

RnF 5. Tiempo de respuesta por transacción

El tiempo promedio que debe demorar el sistema en realizar una transacción debe ser de dos segundos. El tiempo máximo que puede demorar es de 5 a 10 segundos durante la publicación e importación de un recurso educativo.

RnF 6. Rendimiento

La cantidad de datos que son transmitidos por segundos es de 500 KB.

RnF 7. Capacidad

El sistema consume por cada sesión de usuario alrededor de 30 MB.

- ✓ **Soporte**

RnF 8. Estándar de codificación

El estándar de codificación utilizado es el definido en el marco de trabajo XALIX.

- ✓ **Restricciones de diseño**

RnF-9. Lenguaje de programación

El lenguaje de programación utilizado es: PHP v 5.4

RnF-10. Entorno de Desarrollo Integrado

El Entorno de Desarrollo Integrado utilizado es: Netbeans v 7.3.

RnF-11. Framework de desarrollo

El framework de desarrollo utilizado es: Symfony v 2.3.7.

RnF-12. Estilo arquitectónico

El estilo arquitectónico utilizado es: Modelo-Vista-Controlador.

RnF-13. Framework CSS

Para el diseño de las interfaces gráficas se empleó Bootstrap v 3.0.

- ✓ **Requisitos de licencia**

RnF-14. Licencia

El repositorio está registrado bajo la Licencia Pública General (GPL) versión 3.0.

- ✓ **Estándares aplicables**

RnF-15. Estándar para la realización de búsquedas SQI.

El repositorio permite la realización de búsquedas federadas en otros repositorios mediante el estándar SQI.

RnF.16 Estándar para la publicación de contenidos SPI.

El repositorio permite la publicación de recursos educativos y metadatos utilizando el estándar SPI.

RnF-17. Estándar para la realización de búsquedas por metadatos recolectados OAI-PMH.

El repositorio es capaz de recolectar metadatos en otros repositorios y aplicaciones e-learning para posteriormente realizar búsquedas a partir de los mismos.

2.5. Modelado de casos de uso del sistema

El modelo de casos de uso es la especificación de los requisitos con la perspectiva del usuario, cuyo objetivo es delimitar y definir las funcionalidades. Durante el modelado, se describe el sistema como un conjunto de casos de uso (CU) que son ejecutados por diferentes de actores.(39)

2.5.1. Actores del sistema

Los actores representan entidades externas que interactúan directamente con el sistema (personas, máquinas u otros sistemas).(39) Una vez identificado los actores, queda definido el entorno externo del sistema.

Tabla 1. Actores del Sistema

Actor	Objetivos
Administrador	Persona registrada en el sistema que cuenta con todos los privilegios del mismo, es el encargado de las configuraciones necesarias para el funcionamiento de los estándares.

2.5.2. Casos de uso del sistema

Un CU es una secuencia de interacciones que se desarrollan entre el sistema y sus actores, en respuesta a un evento que inicia un actor principal sobre el propio sistema.(39)

CU 1. Gestionar repositorio para OAI-PMH.

CU 2. Definir la periodicidad de actualización del catálogo de metadatos.

CU 3. Actualizar catálogo de metadatos.

CU 4. Ofrecer catálogo de metadatos.

CU 5. Almacenar un recurso educativo mediante SPI.

CU 6. Gestionar repositorio para SQI.

CU 7. Gestionar catálogos.

CU 8. Notificar al administrador las publicaciones y actividades automáticas.

El diagrama de CU sirve para especificar la comunicación y el comportamiento del repositorio mediante su interacción con los usuarios y otros sistemas, es decir, un diagrama que muestra la relación entre los actores y los CU.

2.5.3. Diagrama de casos de uso del Sistema

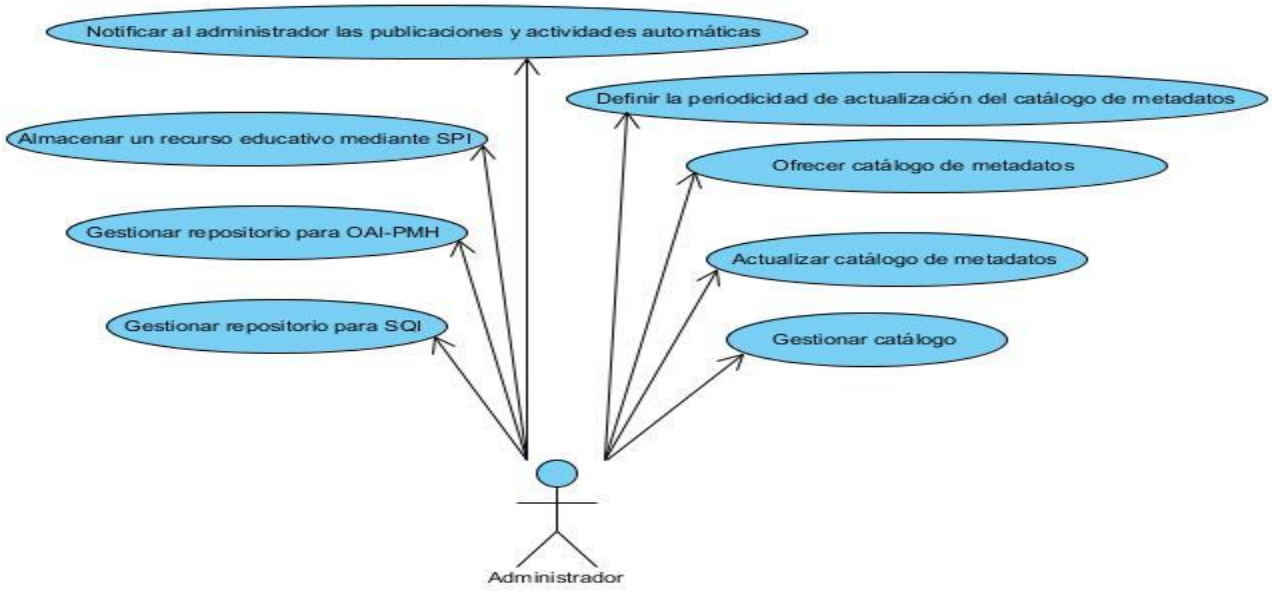


Figura 10. Diagrama CU del sistema.

2.5.4. Descripción de casos de uso

La descripción de los CU contiene las propiedades textuales de los CU, incluyendo una descripción del flujo de eventos y describiendo la interacción entre los actores y el sistema. A continuación se describen cada uno de los CU del sistema.

Tabla 2. Especificación de CU Gestionar repositorio para SQI.

Objetivo	Administrar los repositorios para buscar con SQI.
Actores	Administrador
Resumen	El caso de uso inicia cuando el administrador selecciona en el menú la opción *Repositorio SQI*. El sistema muestra las opciones de agregar nuevo, editar, borrar y un listado de los repositorios. El administrador realiza las operaciones que desee y de esta forma se gestionan los repositorios. El caso de uso termina.
Complejidad	Media

Prioridad	Alta	
Precondiciones	El usuario debe estar autenticado como administrador y tener el rol de administrador.	
Postcondiciones	Ha quedado adicionado, editado o eliminado uno o varios repositorios.	
Flujo de eventos		
Flujo básico "Gestionar repositorio para SQI"		
	Actor	**Sistema**
1	Selecciona la opción: *Repositorios SQI*	1.1. Muestra una interfaz con los repositorios existentes mostrando de cada uno los siguientes datos: - Url Authentication (Url por la que se autentifica el usuario SQI al repositorio). - Url Search (Url de búsqueda). - Name (Nombre del repositorio). - Description (Breve descripción del repositorio). - Query Language (Lenguaje de consulta). - User Sqi (Usuario SQI). - Password (Contraseña del usuario SQI). 1.2. Permite por cada repositorio realizar las acciones: - Agregar un nuevo repositorio (Ver sección 1: Agregar nuevo repositorio). - Editarlo (Ver sección 2: Editar repositorio). - Borrarlo (Ver sección 3: Borrar repositorio). 1.3. El botón "Cancelar" para realizar las acciones correspondientes.
2	Hace clic en el botón ¨Cancelar¨.	2.1. Termina el caso de uso.
Sección 1: "Agregar nuevo repositorio"		
Flujo básico "Gestionar repositorio para SQI"		

	Actor	**Sistema**
2	Selecciona la opción: *Agregar nuevo*.	2.1. Muestra un formulario para que se introduzca los datos del nuevo repositorios, con SQI, para ello debe ingresar los siguientes datos: - Url Authentication (Tipo Dato: String). - Url Search (Tipo Dato: String). - Name (Tipo Dato: String). - Description (Tipo Dato: String). - Query Language (Tipo Dato: String). - User Sqi (Tipo Dato: String). - Password (Tipo Dato: String). - Los botones “Aceptar” y “Cancelar” para realizar las acciones correspondientes.
3	Ingresa todos los datos y hace clic en el botón *Aceptar.*	3.1. Muestra el mensaje de notificación: *“Elemento creado satisfactoriamente”*. 3.2. Guarda el repositorio en la base de datos y regresa al listado de los repositorios existentes. 3.3. Termina el caso de uso.
Flujos alternos		
Evento 3.a Introducir datos incorrectos		
	Actor	**Sistema**
3.	Ingresa todos los datos y hace clic en el botón *Aceptar* con datos incorrectos.	3.1. Muestra el mensaje de error: *“Se ha producido un error durante la creación del elemento”*. 3.2. Diferencia los campos en rojo, donde están los datos incorrectos. 3.3. Permite que sean modificados los datos introducidos incorrectamente. 3.4. Se mantiene en ejecución el flujo básico de eventos para el paso 2.
Evento 3.b Campos vacíos		

3	Ingresa todos los datos y hace clic en el botón *Aceptar* con campos vacíos.	3.1. Muestra el mensaje de error*: "Existen campos en blanco"*. 3.2. Diferencia en rojo los campos donde están los datos faltantes. 3.3. Permite que sean entrados los datos faltantes.
Evento 3.c Cancelación de acción		
3	Hace clic en el botón *Cancelar*.	3.1. Termina la acción de adicionar repositorio SQl.
Sección 2: "Editar repositorio"		
Flujo básico "Gestionar repositorio para SQl"		
	Actor	**Sistema**
2	Selecciona la opción: *Editar*.	2.1 Muestra una interfaz con los datos que pueden modificarse del repositorio: - Url Authentication (Tipo Dato: String). - Url Search (Tipo Dato: String). - Name (Tipo Dato: String). - Description (Tipo Dato: String). - Query Language (Tipo Dato: String). - User Sqi (Tipo Dato: String). - Password (Tipo Dato: String). - Los botones "Aceptar" y "Cancelar" para realizar las acciones correspondientes.
3	Modifica los datos necesarios y hace clic en el botón *Aceptar*.	3.1. Se muestra el mensaje de notificación: "*Elemento actualizado satisfactoriamente"*. 3.2. Termina el caso de uso
Flujos alternos		
Evento 3.a Introducir datos incorrectos		
	Actor	**Sistema**
3.	Modifica los datos necesarios y hace clic en el botón *Aceptar* con datos incorrecto.	3.1. Muestra mensaje de error: *"Se ha producido un error durante la actualización del elemento".* 3.2. Diferencia los campos en rojo, donde están los

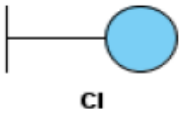
		datos incorrectos. 3.3. Permite que sean modificados los datos introducidos incorrectamente. 3.4. Se mantiene en ejecución el flujo básico de eventos para el paso 2.
Evento 3.b Campos vacíos		
3	Modifica los datos y hace clic en el botón *Aceptar* con campos vacíos.	3.1. Muestra el mensaje de error*: "Existen campos en blanco"*. 3.2. Diferencia en rojo los campos donde están los datos faltantes. 3.3. Permite que sean entrados los datos faltantes.
Evento 3.c Cancelación de acción		
3	Hace clic en el botón *Cancelar*.	3.1. Termina la acción de modificar repositorio SQI.
Sección 3: "Borrar repositorio"		
Flujo básico "Gestionar repositorio para SQI"		
	Actor	**Sistema**
2	Selecciona la opción: *Borrar*	2.1 Muestra el mensaje*: "¿Está seguro que quiere borrar el elemento seleccionado?"*. 2.2. Muestra los botones "Si" y "No" para las acciones correspondientes.
3	Selecciona la opción: *Si*	3.1. Muestra el mensaje: "*Elemento eliminado satisfactoriamente".* 3.2. Termina el caso de uso.
Flujos alternos		
Evento 3.a Cancelación de acción		
	Actor	**Sistema**
3	Hace clic en el botón: ¨No¨	3.1 Regresa a la interfaz del listado de repositorios para buscar con SQI. 3.2 Termina el caso de uso.

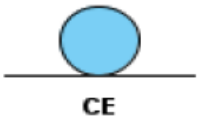
Relaciones	**CU Incluidos**	
	CU Extendidos	
Requisitos no funcionales		
Asuntos pendientes		

2.6. Modelo de análisis.

El modelo de análisis es una abstracción del sistema que define una estructura para modelarlo y puede o no mantenerse durante todo el ciclo de vida del software. Por esta razón, sólo se centra en qué debe hacer el sistema mediante los requisitos funcionales que se presentan en los diagramas de clases del análisis de cada uno de los CU. Los diagramas de clases del análisis representan las relaciones entre los actores y el sistema, que se muestra seccionado en clases: interfaz, controladoras y entidades; que pueden ser la abstracción de una o varias clases o subsistemas del diseño del sistema.(40)

Tabla 3. Modelo de análisis.

Nombre	Figura	Descripción
Clases interfaz		Se utilizan para modelar la interrelación entre el sistema y sus actores. Modelan las partes del sistema que dependen de sus actores lo que implica que clarifiquen y reúnan los requisitos en los límites del sistema.
Clases control		Representan la coordinación, secuencia, transacciones y control de objetos, además se usan con frecuencia para encapsular el control de un caso de uso en concreto.

Clases entidad		Se utilizan para modelar la información que posee una vida larga. En la mayoría de los casos se derivan directamente de una clase entidad del negocio.

Seguidamente se presentan los diagramas de clases del análisis correspondiente al CU Gestionar repositorio para SQI. Los restantes diagramas pueden ser consultados en el Anexo 5 diagramas de clases de análisis.

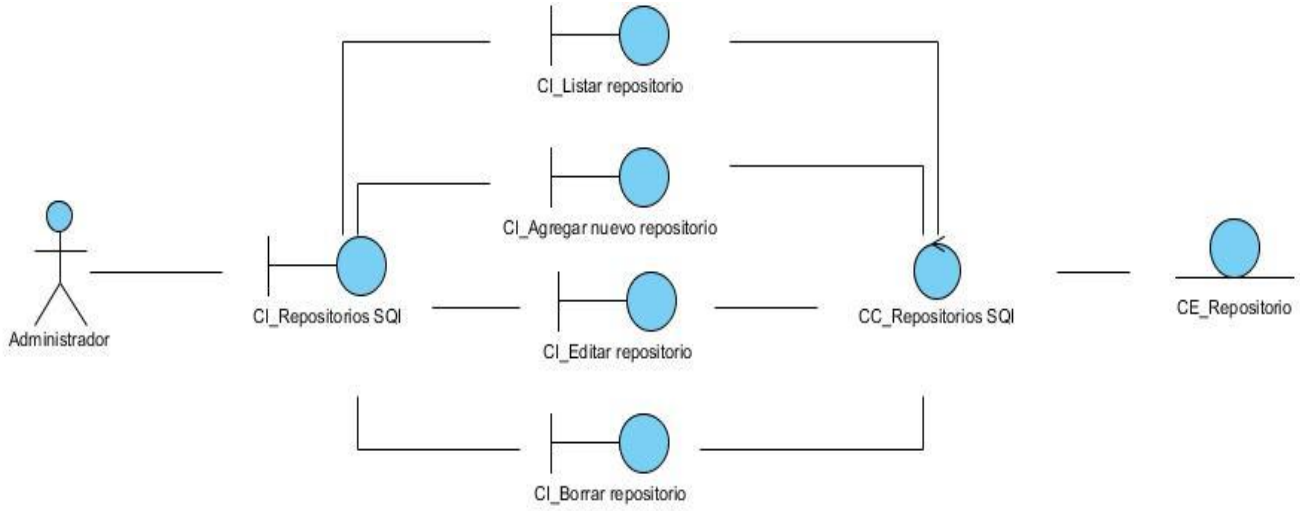


Figura 11. Diagrama de clase de análisis: CU Gestionar repositorio para SQI.

2.7. Diagramas de colaboración

Un diagrama de colaboración (DC) representa las interacciones entre objetos organizadas alrededor de los objetos y los enlaces que existen entre ellos. A diferencia de un diagrama de secuencias, un DC muestra las relaciones entre los objetos, no la secuencia en el tiempo en que se producen los mensajes. Los diagramas de secuencia y los diagramas de colaboración expresan información similar, pero en una forma diferente.(41)

A continuación se representan los diagramas de colaboración correspondientes al CU Gestionar repositorio para SQI. Los restantes diagramas pueden ser consultados en el Anexo 6 diagramas de colaboración.

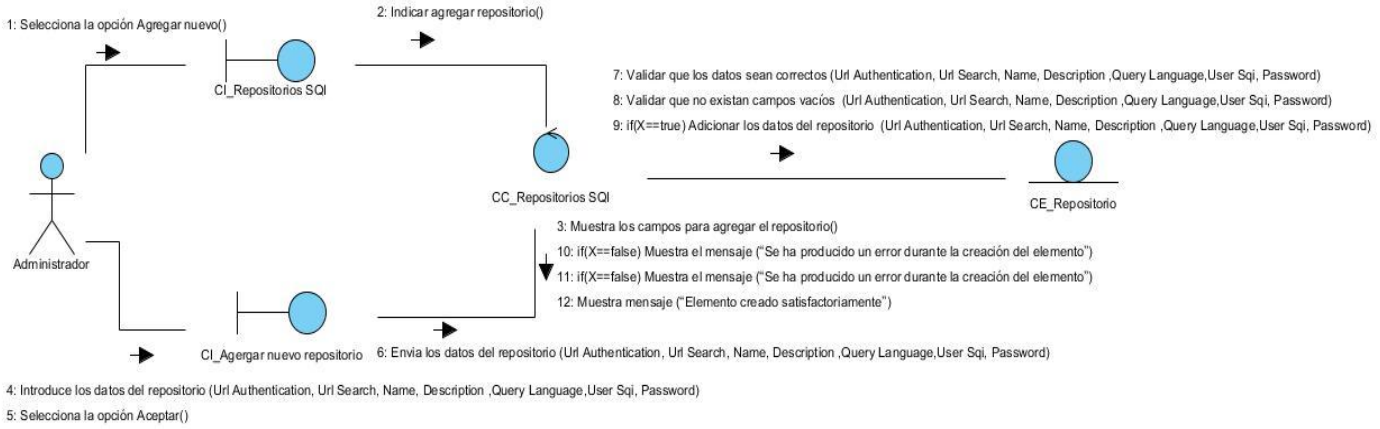


Figura 12. DC del CU Gestionar repositorio para SQI. Sección agregar repositorio.

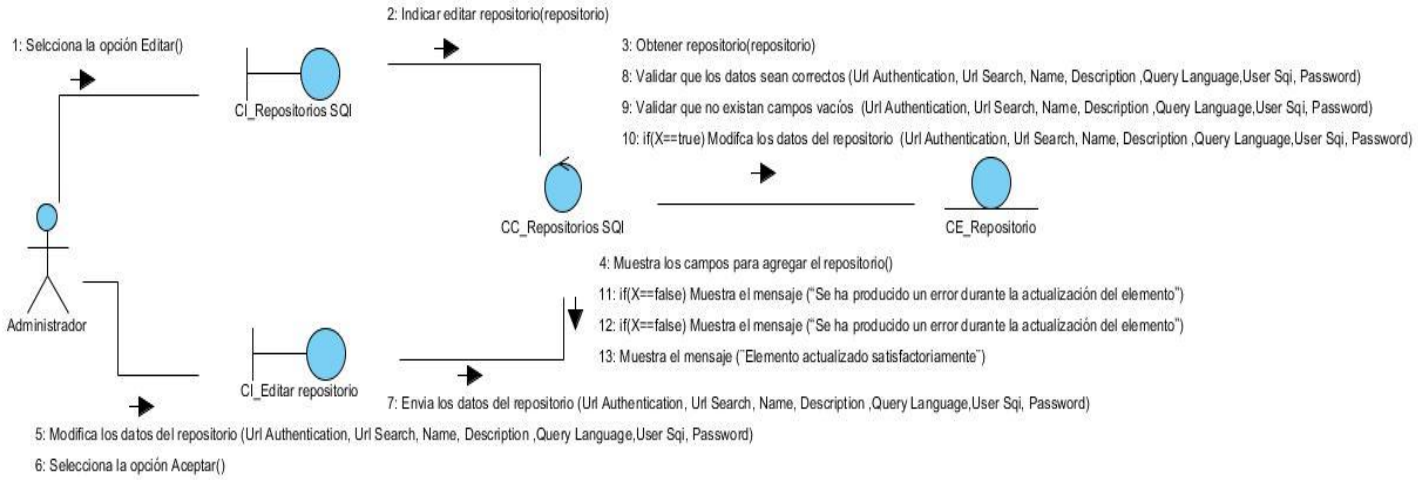


Figura 13. DC del CU Gestionar repositorio para SQI. Sección editar repositorio.

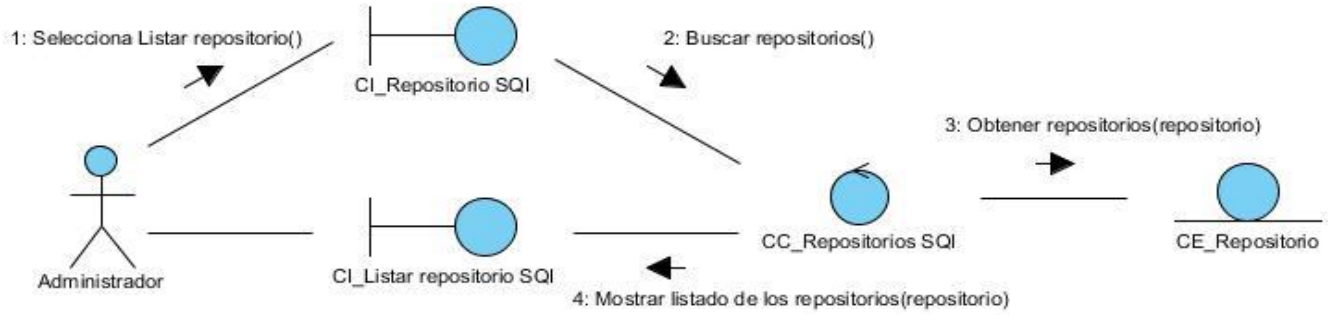


Figura 14. DC del CU Gestionar repositorio para SQI. Sección listar repositorio.

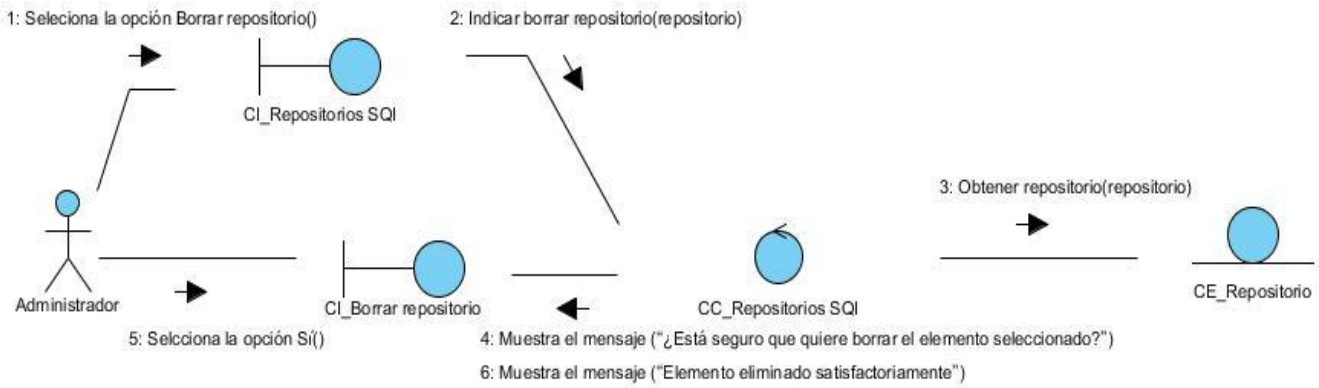


Figura 15. DC del CU Gestionar repositorio para SQI. Sección borrar repositorio.

2.8. Modelo del diseño

El modelo de diseño es un modelo de objetos que describe la realización física de los CU centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen un impacto en el sistema a considerar. Este modelo sirve como abstracción de la implementación del sistema y es, de ese modo, utilizado como una entrada fundamental de las actividades de implementación.(42)

2.8.1. Patrón arquitectónico

Symfony está basado en un patrón clásico de arquitectura web conocido como Modelo-Vista-Controlador (MVC). La utilización de este marco de trabajo para el desarrollo del producto conlleva por ende a la

implementación de este patrón, que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos:(43)

- ✓ **Modelo:** es un conjunto de clases que representan la información del mundo real o que el sistema debe procesar, sin tomar en cuenta la forma en la que esa información va a ser mostrada ni los mecanismos que hacen que esos datos estén dentro del modelo. En este se encuentran las clases que son generadas de forma automática según la estructura de la base de datos.
- ✓ **Vista:** presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario. La vista es la encargada de originar las páginas que son mostradas como resultado de las acciones.
- ➢ **Controlador:** propone como solución asignar la responsabilidad de recibir o manejar un mensaje de los eventos del sistema a otra clase que representa una de las siguientes opciones:
  - representa el sistema global (controlador de fachada).
  - representa un escenario de caso de uso en el que tiene lugar un evento del sistema.

La arquitectura MVC proporciona grandes ventajas, como la organización del código, la reutilización y la flexibilidad. Symfony2 específicamente, toma lo mejor de la arquitectura MVC y la implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo.

2.8.2. Patrones de diseño

Los patrones ayudan a capturar conocimiento y a crear un vocabulario técnico, hacen el diseño orientado a objetos más flexible, elegante y en algunos casos reusable. "*Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular".*(44)

Seguidamente se presentan los patrones de diseño que se tuvieron en cuenta en la realización del diseño del sistema.

- ✓ **Patrones GRASP**

Los Patrones Generales de Software para Asignar Responsabilidades (GRASP, General Responsibility Assignment Software Patterns) "*son parejas de problema de solución con un nombre, que codifica buenos principios y sugerencias relacionadas frecuentemente con la asignación de responsabilidades".*(45)

Los patrones utilizados en la presente investigación son:

- **Experto:** es uno de los patrones que más se utiliza cuando se trabaja con Symfony, con la inclusión de la librería Doctrine para mapear la base de datos. En el modelo existen dos clases fundamentales: las clases encargadas de hacer las consultas a la base de datos utilizando Doctrine, ya que tienen los atributos necesarios para efectuar dicha función, por tanto tienen la responsabilidad de realizar directamente las acciones con la base de datos y las clases de acceso a datos, son las responsables de interactuar con las clases de abstracción de datos y devolver los objetos que necesitan los controladores en su forma original.
- **Creador:** este patrón establece la responsabilidad de quién es el encargado de crear objetos de una clase determinada. Es usado para crear objetos entidades, de formularios y de clases de abstracción a la base de datos.
- **Bajo Acoplamiento:** la solución de este patrón consiste en asignar una responsabilidad para mantener el bajo acoplamiento, lo que permite el diseño de clases independientes, y la reutilización de las mismas. Además, mejora la comprensión de las clases aisladas, facilita la reutilización de código y no afecta a los cambios en otros componentes.
- **Alta Cohesión**: Symfony permite la organización del trabajo en cuanto a la estructura del proyecto y la asignación de responsabilidades con una alta cohesión. Un ejemplo de ello es la clase Actions, la cual está formada por varias funcionalidades que están estrechamente relacionadas, siendo la misma la responsable de definir las acciones para las plantillas y colaborar con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades.
- **Controlador**: todas las peticiones web son manipuladas por un solo controlador frontal (sfActions), que es el punto de entrada único de toda la aplicación en un entorno determinado.

2.8.3. Diagramas de clases de diseño

El Diagrama de Clases de Diseño (DCD) describe gráficamente las especificaciones de las Clases de Software y las Interfaces en una aplicación. Normalmente contiene clases, asociaciones y atributos, interfaces con sus operaciones y constantes, información sobre los tipos de atributos, navegabilidad y dependencia.

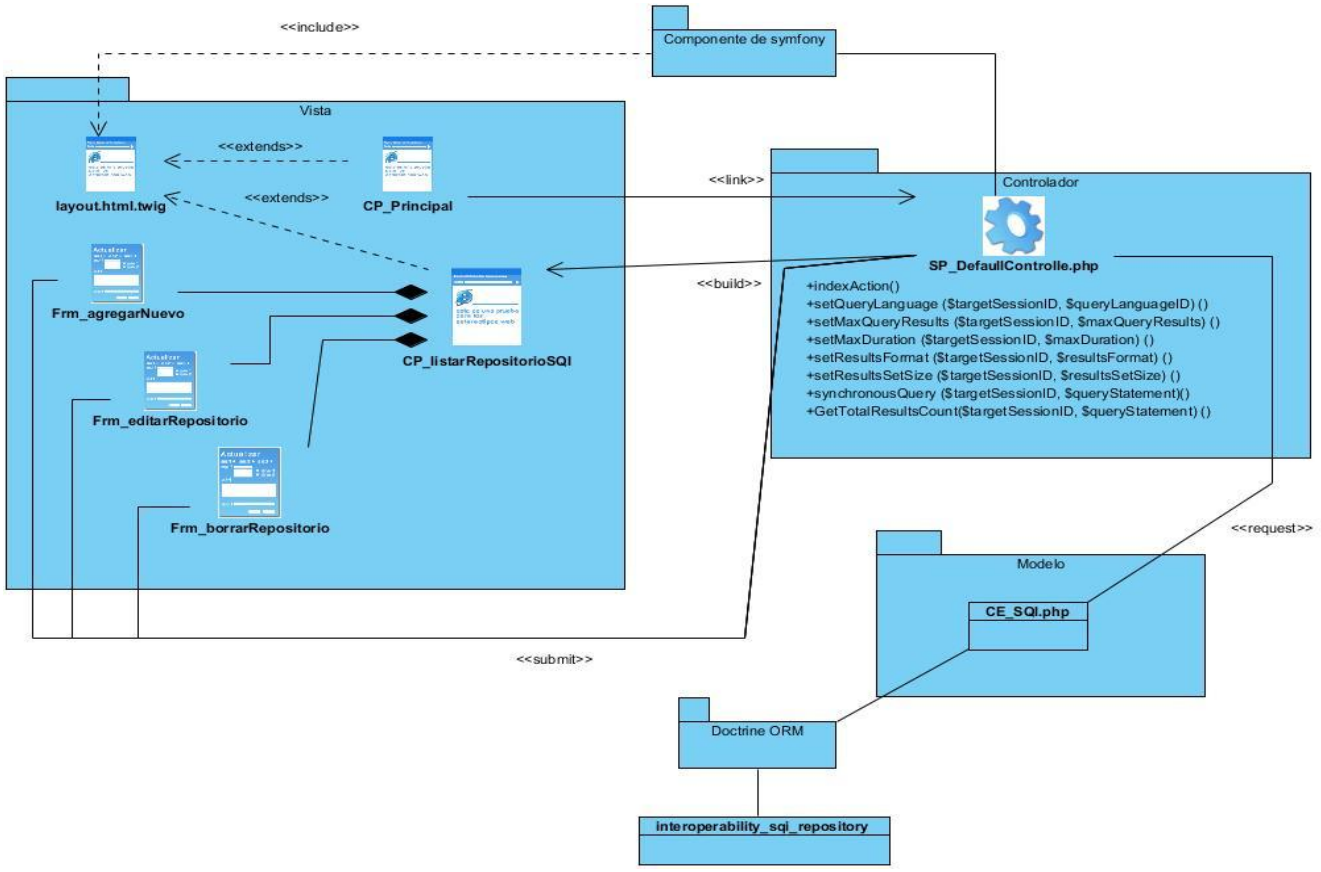


Figura 16. DCD del CU: Gestionar repositorio para SQI.

Conclusiones Parciales

El modelo conceptual, elaborado a partir de los principales conceptos involucrados en el entorno del problema, contribuyó a la identificación de los RF y RnF para la implementación de los estándares en RHODA. Estos elementos permitieron la concepción de los CU representados en el modelo de CU del sistema, así como las descripciones textuales de cada uno de ellos. La unión de estos artefactos guió la elaboración del modelo de análisis con sus respectivos diagramas de clases del análisis y del diseño.

Capítulo 3. Implementación y prueba

Introducción del capítulo

En el presente capítulo se abordaran los elementos referentes a la implementación de la solución anteriormente analizada y modelada, en la cual se hará evidente el uso de los estándares para la interoperabilidad. Esto será posible mediante la traducción del diseño, tomando guía los diagramas de interacción como base para llevar a cabo la interoperabilidad. Además, serán abordadas las pruebas desarrolladas que permitirán comprobar el funcionamiento de la propuesta de solución, así como los resultados obtenidos.

3.1. Modelo de Implementación

El proceso de implementación parte como resultado del análisis y diseño de la propuesta de solución planteada. Las conclusiones de la misma son dadas teniendo en cuenta términos de componentes, posee como objetivo principal llevar cabo la arquitectura y el sistema como un todo.(46) Obteniendo como artefacto generado el diagrama de componente y el modelo de dato**.**

3.1.1. Diagrama de componentes

El diagrama de componentes es uno de los principales artefactos generados durante la implementación. Entre sus principales objetivos comprende: mostrar la dependencia lógica entre los distintos componentes del software y representar las relaciones entre los elementos que forman el código del sistema implementado. Estos esquemas describen unidades físicas del sistema y las relaciones existentes dentro del mismo. Cada unidad puede representar archivos simples, paquetes y librerías.(32)

El diagrama de componentes del sistema desarrollado se estructuró en paquetes, donde a cada clase definida en el diseño se le asignó un componente representando además las bibliotecas que necesita el sistema para su correcto funcionamiento. Los paquetes resultan ser los identificados en el diseño.

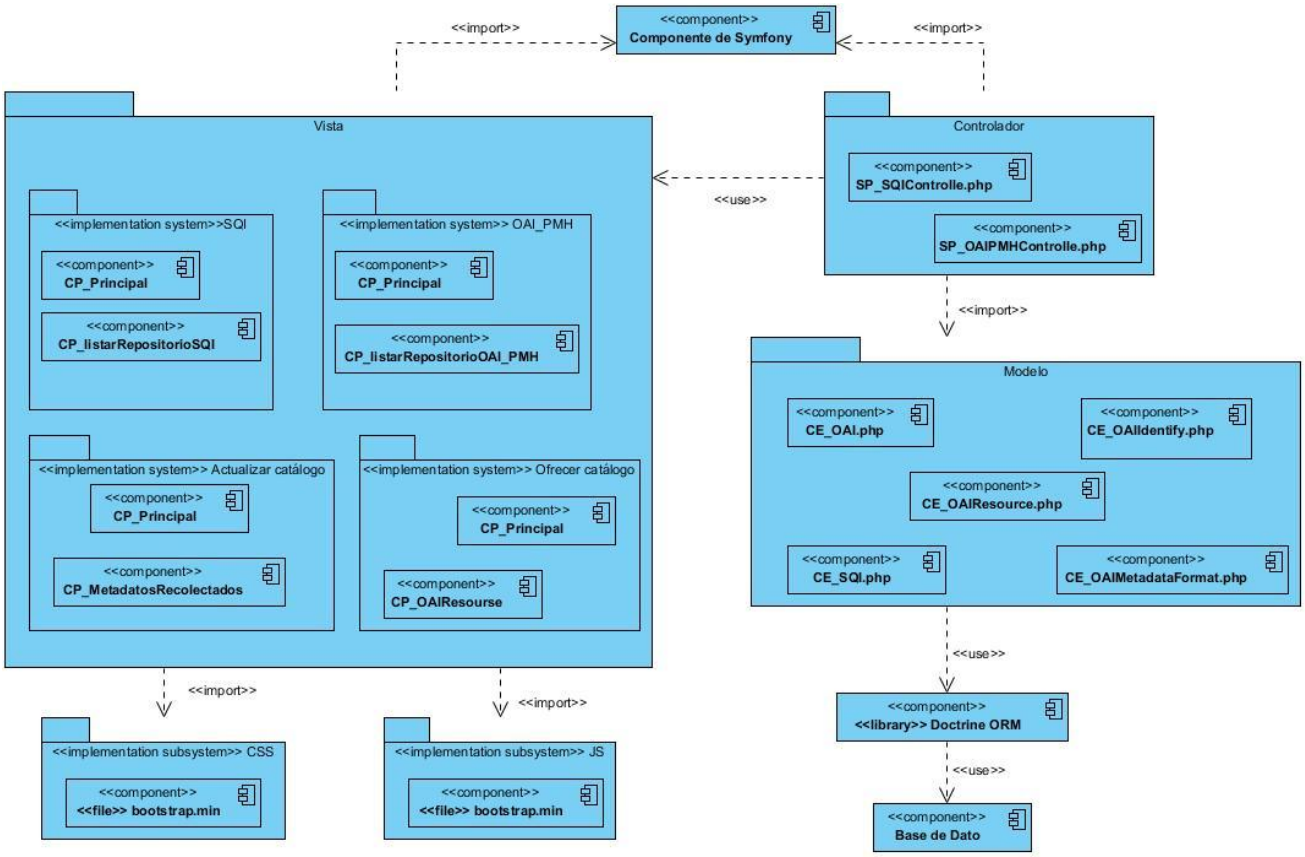


Figura 17. Diagrama de Componente.

3.1.2. Modelo de datos

Un modelo de datos es un conjunto de conceptos, reglas y convenciones que describen los datos. Constituye una herramienta que facilita la interpretación y representación en forma de datos, permitiendo definir formalmente las estructuras aprobadas y sus restricciones a fin de representar los datos de los elementos básicos en el diseño de una base de datos.

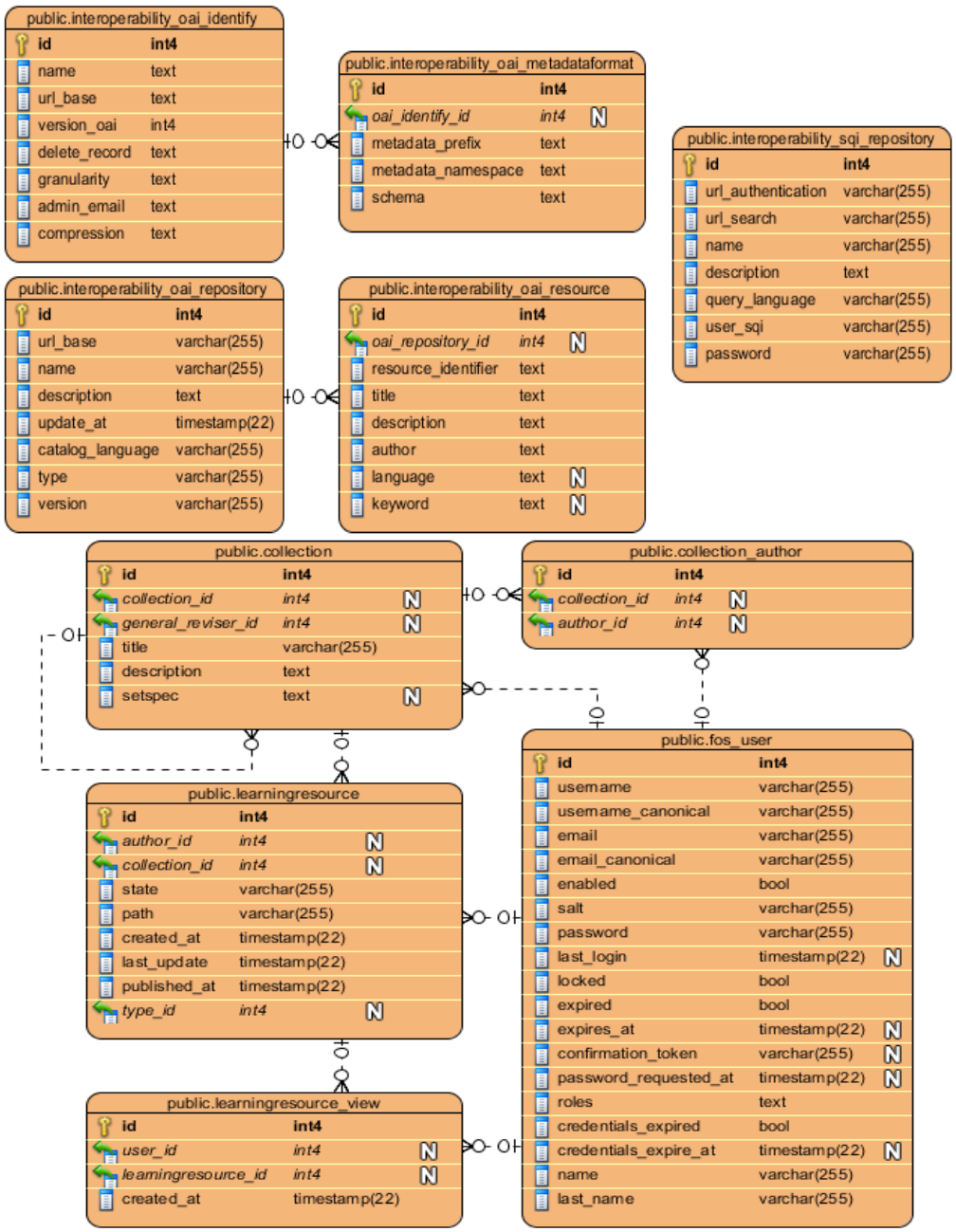


Figura 18. Modelo de datos del sistema.

3.2. Pruebas de software

La realización de las pruebas es una de las etapas principales durante todo el ciclo de vida del software. Las mismas constituyen una serie de actividades donde cada componente, o parte de la aplicación a comprobar, son ejecutados bajo determinadas condiciones. Su objetivo principal es evaluar y elevar la calidad del producto final.(8)

Los procedimientos que se realicen para ejecutar las pruebas deben verificar el correcto funcionamiento de los requisitos implementados y garantizar la ejecución correcta, al menos una vez, de los posibles caminos o rutas que se realizan para el intercambio de recursos educativos de RHODA con alguna herramienta externa, ejemplo CRODA.

Dentro de los objetivos fundamentales que se persiguen al aplicarle las pruebas a un software se encuentran los siguientes:(47)

- ✓ Descubrir un error que aún no ha sido descubierto.
- ✓ Encontrar el mayor número de errores con la menor cantidad de tiempo y esfuerzo posibles.
- ✓ Mostrar hasta qué punto las funciones del software operan de acuerdo con las especificaciones y requisitos del cliente.

RUP define cuatro niveles de pruebas: las pruebas de unidad, pruebas de integración, pruebas de sistema y pruebas de aceptación. Para detectar errores y validar el cumplimiento de los requisitos se decidió realizar pruebas de sistema, utilizando el método pruebas de caja negra. Además, se realizaron pruebas de unidad con el objetivo de verificar el funcionamiento interno de los requisitos implementados.

3.2.1. Pruebas de unidad

Antes de iniciar cualquier otra prueba es preciso probar el flujo de datos de la interfaz del componente. Si la entrada de los datos no se producen correctamente, las demás pruebas carecen de sentido. El diseño de casos de prueba de una unidad comienza una vez que se ha desarrollado, revisado y verificado en su sintaxis el código a nivel fuente.

Las pruebas unitarias aseguran que un único componente de la aplicación produzca una salida correcta para una determinada entrada. Este tipo de pruebas validan la forma en la que las funciones y métodos trabajan en cada caso particular. Se encargan de un único caso cada vez, lo que significa que un único método puede necesitar varias pruebas unitarias si su funcionamiento varía en función del contexto.

Symfony 2 ha optado por utilizar la librería PHPUnit, que prácticamente se ha convertido en un estándar en el mundo PHP. De esta forma, los *tests* (pruebas) unitarios de Symfony 2 combinan la potencia de PHPUnit con las utilidades y facilidades proporcionadas por Symfony 2. Por convención, cada test unitario y funcional de Symfony 2 se define en una clase cuyo nombre acaba en Test y se encuentra dentro del directorio Tests/ del bundle. Además, se recomienda utilizar dentro de Tests/ la misma estructura de directorios del elemento que se quiere probar. PHPUnit utiliza *assertions* (aserciones) para verificar que el comportamiento de una unidad de código es el esperado. Cuando se produce un error, PHPUnit muestra el texto *"Failures"* (Fallo) como resumen de la ejecución. Antes muestra el listado de todos los *tests* que han fallado, indicando para cada error la clase y método erróneos, el mensaje propio que se incluyó en el *test* e información adicional como el valor esperado y el valor obtenido. Un test que no pasa satisfactoriamente es la mejor señal de que algo no funciona bien en la aplicación. Esta es la gran ventaja de los *tests* unitarios, que te avisan cada que vez hay problemas en la aplicación. El mensaje *"Ok"* (Bien) indica que todos los *tests* se han ejecutado correctamente.(48)

Estas pruebas fueron realizadas a 5 de las clases de la capa de abstracción. A continuación se muestran ejemplos de las pruebas realizadas haciendo uso de la librería PHPUnit.

```
amet@d5-l206-15:/var/www/rhoda3.0$ phpunit -c app src/Rhoda/InteroperabilityBundle/Tests
/Entity/OAIMetadataFormatTest.php
PHPUnit 3.6.10 by Sebastian Bergmann.

Configuration read from /var/www/rhoda3.0/app/phpunit.xml.dist

...

Time: 0 seconds, Memory: 3.25Mb

OK (3 tests, 9 assertions)
```

Figura 19. Prueba de unidad realizada a la entidad OAIMetadaFormatTest.

```
amet@d5-l206-15:/var/www/rhoda3.0$ phpunit -c app src/Rhoda/InteroperabilityBundle/Tests
/Entity/OAIIdentifyTest.php
PHPUnit 3.6.10 by Sebastian Bergmann.

Configuration read from /var/www/rhoda3.0/app/phpunit.xml.dist

.......

Time: 0 seconds, Memory: 3.50Mb

OK (7 tests, 21 assertions)
```

Figura 20. Prueba de unidad realizada a la entidad OAIIdentifyTest.

3.2.2. Pruebas de caja negra

Conocidas también como pruebas de comportamiento, se centran principalmente en los requisitos funcionales detectados con anterioridad. El mismo no incluye revisión de codificación interna del software ya que se especializa en la comprobación de interfaces, contando con el apoyo de casos de prueba de cada funcionalidad. Su objetivo principal es detectar errores de las siguientes categorías:(49)

- ✓ Errores de funciones incorrectas o ausentes.
- ✓ Errores de interfaz.
- ✓ Errores en estructura de datos o en accesos de datos externas.
- ✓ Errores de rendimiento.
- ✓ Errores de inicialización y de terminación.

Con la aplicación de las cajas negras queda validado el cumplimiento de los requerimientos funcionales del software se verifica si el módulo cumple con los objetivos esperados. Para la realización de dichas pruebas se utilizó la técnica de partición equivalente, esta técnica divide el campo de entrada de un sistema en clases de datos que permitan derivar casos de prueba.

3.2.3. Diseño de casos de pruebas

Un caso de prueba cubre el software más a fondo y con más detalle que un CU. Los casos de prueba incluyen todas las funciones que el programa es capaz de realizar. Estas pruebas deben tener en cuenta el uso de todo tipo de datos de entrada/salida, cada comportamiento esperado, todos los elementos de diseño, y cada clase de defecto.(50)

Caso de prueba: Gestionar repositorio para SQI.

Descripción General: Permite al administrador agregar, editar, borrar y listar los repositorios.

Condiciones de Ejecución: El usuario debe estar autenticado como administrador y tener el rol de administrador.

Tabla 4. Sección 1: Agregar repositorio SQI.

Escenario	Descripción	Url Authentication	Url Search	Name	Description	Query Language	User Sqi	Password	Respuesta del sistema	Flujo central
EC 1.1 Agregar nuevo repositorio SQI	Adiciona correctamente el repositorio SQI.	V http://repositorio.uci.cu	V http://croda.uci.cu	V CRODA	V Herramienta de autor	V PLQL	V candido	V *********	Muestra el mensaje de notificación: "Elemento creado satisfactoriamente"	Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Agregar nuevo /Aceptar
EC 1.2 Ingresa todos los datos del repositorio y selecciona la opción Aceptar con datos incorrectos.	No adiciona el repositorio por tener datos incorrectos	I V V V V V V I	V I V V V V V I	V V I V V V V I	V V V I V V V I	V V V V I V V I	V V V V V I V I	V V V V V V I I	Muestra el mensaje de error: "Se ha producido un error durante la creación del elemento".	Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Agregar nuevo/Aceptar
EC 1.3 Ingresa todos los datos del repositorio y selecciona la opción Aceptar con campos vacíos	No adiciona el repositorio por tener campos en blanco.	I V V V V V V I	V I V V V V V I	V V I V V V V I	V V V I V V V I	V V V V I V V I	V V V V V I V I	V V V V V V I I	Muestra el mensaje de error: "Existen campos en blanco".	Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Agregar nuevo/Aceptar
EC 1.4 Selecciona la opción cancelar	Cancela la opción de agregar un nuevo repositorio.	NA	NA	NA	NA	NA	NA	NA		Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Agregar nuevo/Cancelar

Tabla 5. Sección 2: Editar repositorio SQI.

Escenario	Descripción	Url Authentication	Url Search	Name	Description	Query Language	User Sqi	Password	Respuesta del sistema	Flujo central
EC 1.1 Modificar datos del repositorio SQI.	Modificar correctamente datos del repositorio SQI	V http://repositorio.uci.cu	V http://croda.uci.cu	V CRODA	V Herramienta de autor	V PLQL	V candido	V *********	Se muestra el mensaje de notificación: "Elemento actualizado satisfactoriamente"	Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Editar /Aceptar
EC 1.2 Modifica los datos del repositorio y hace clic en el botón Aceptar con datos incorrectos.	No actualiza el repositorio por tener datos incorrectos.	I V V V V V V	V I V V V V V	V V I V V V V	V V V I V V V	V V V V I V V	V V V V V I V	V V V V V V I	Muestra mensaje de error: "Se ha producido un error durante la actualización del elemento".	Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Editar /Aceptar
EC 1.3 EC 1.2 Modifica los datos del repositorio y hace clic en el botón Aceptar con campos vacíos	No actualiza el repositorio por tener campos en blanco.	I V V V V V V I	V I V V V V V I	V V I V V V V I	V V V I V V V I	V V V V I V V I	V V V V V I V I	V V V V V V I I	Muestra el mensaje de error: "Existen campos en blanco".	Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Editar /Aceptar
EC 1.4 Selecciona la opción cancelar	Cancela la opción de agregar un nuevo repositorio.	NA	NA	NA	NA	NA	NA	NA		Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Editar/Cancelar

Tabla 6. Sección 3: Borrar repositorio SQI.

Escenario	Descripción	Url Authentication	Url Search	Name	Description	Query Language	User Sqi	Password	Respuesta del sistema	Flujo central
EC 1.1 Borrar repositorio SQI.	Borra el repositorio SQI	V http://repositorio.uci.cu	V http://croda.uci.cu	V CRODA	V Herramienta de autor	V PLQL	V candido	V *********	Muestra el mensaje: "¿Está seguro que quiere borrar el elemento seleccionado?".	Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Borrar/Sí
EC 1.2 El autor selecciona de No borrar	Cancela la opción de borrar el repositorio.	NA	NA	NA	NA	NA	NA	NA		Autenticarse/Administración/Autentificación/Interoperabilidad/Repositorios SQI/Borrar/No

Nota:

Las celdas de la tabla contienen V, I, o N/A. V indica válido, I indica inválido, y N/A que no es necesario proporcionar un valor del dato en ese caso, ya que es irrelevante.

Tabla 7. Descripción de variables SQI.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Url Authentication	Campo de texto	No	Url por la que puede acceder el usuario sqi al repositorio. Ejemplo: http://repositorio.uci.cu
2	Url Search	Campo de texto	No	Url del repositorio. Ejemplo: http://biblioteca.uci.cu
3	Name	Campo de texto	No	Nombre del repositorio.
4	Description	Campo de texto	No	Breve descripción del repositorio.
5	Query Language	Campo de texto	No	Lenguaje de consulta.
6	User Sqi	Campo de texto	No	Usuario sqi.
7	Password	Campo de texto	No	Contraseña del usuario.

3.3. Resultado de las pruebas

Durante la aplicación de las pruebas anteriormente descritas, se detectaron un conjunto de deficiencias que fueron recogidas en el expediente del proyecto. Se realizaron 3 iteraciones, donde se detectaron un total de 31 no conformidades, de las cuales 18 fueron no significativas, 9 significativas y 4 recomendaciones. La mayor parte de las no conformidades detectadas se relacionan con la validación de los datos ingresados al módulo, con el objetivo de erradicarlas se utilizaron expresiones regulares. A

continuación se muestra una gráfica que describe cada una de las iteraciones realizadas en el proceso de prueba.

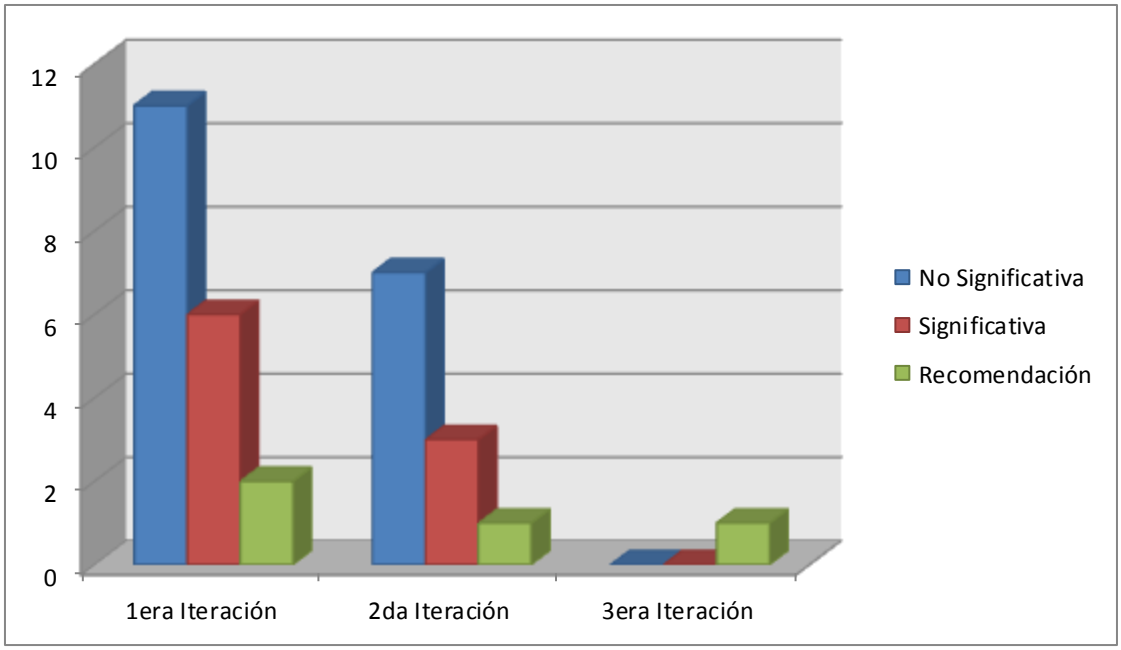


Figura 21. Resultado de las pruebas realizadas en cada iteración.

Conclusiones parciales

Durante el flujo de trabajo de implementación se describieron cómo los elementos del modelo del diseño se implementan en términos de componentes.

Los resultados de las pruebas realizadas lograron su objetivo principal: demostrar la calidad de la solución propuesta a través de un código limpio altamente funcional y un funcionamiento general exitoso.

Conclusiones Generales

Una vez culminada la investigación y después de dar cumplimiento a los objetivos propuestos, se pudo arribar a las siguientes conclusiones:

- ✓ El estudio realizado durante la fundamentación teórica de la investigación, permitió comprender el funcionamiento de los estándares SQI, SPI y OAI-PMH, así como definir qué métodos incorporar para garantizar la interoperabilidad de RHODA.
- ✓ La implementación de los estándares SQI, SPI y OAI-PMH contribuyen al intercambio de los recursos educativos almacenados en RHODA con otros sistemas.
- ✓ Las pruebas de unidad y caja negra aplicadas permitieron identificar y erradicar las deficiencias existentes en la ejecución de las funcionalidades implementadas.

Recomendaciones

Las recomendaciones para la presente investigación que pueden servir de guía para la continuidad de la misma son:

- ✓ Incorporar el lenguaje de consultas ProLearn Query Language (PLQL) integrado al estándar SQI para la realización de las búsquedas sobre el repositorio RHODA.

Bibliografía

1. MUÑOZ, P. G. *La Formación Sin Distancia* de 2013]. Disponible en: http://josebaangulo.files.wordpress.com/2009/12/libro_laformacionsindistancia2006.pdf.

2. BOARD, I. S. *IEEE Standard Glossary of Software Engineering Terminology* (Interoperability). Disponible en: http://www.mit.jyu.fi/ope/kurssit/TIES462/Materiaalit/IEEE_SoftwareEngGlossary.pdf.

3. LEÓN, I. L. M. C. D. *INTEROPERABILIDAD; ESTÁNDARES* de 2013]. Disponible en: http://www.revista.unam.mx/vol.5/num10/art67/nov_art67.pdf.

4. RIQUENE, G. R. *"Análisis de tecnologías para la automatización de catálogos y creación de bibliotecas digitales."*. Universidad de las Ciencias Informáticas, 2011.

5. PANEQUE, Y. F. *Propuesta de repositorio digital para el Sistema de G estión de Documentos Históricos.* Universidad de las Ciencias Informáticas, 2012.

6. GRIFF RICHARDS, R. M., MAREK HATALA, AND NORM FRIESEN. *The Evolution of Learning Object Repository Technologies: Portals for On-line Objects for Learning* de 2014]. Disponible en: http://www.colombiaaprende.edu.co/html/mediateca/1607/articles-172791_archivo.pdf.

7. GONZÁLEZ, I. R. C. *REPOSITORIO DE RECURSOS EDUCATIVOS PARA LAS INSTITUCIONES DE EDUCACIÓN SUPERIOR.* 2012.

8. FERNANDO. *Uso educativo de las TIC* de 2014]. (Diseño de recursos digitales educativos). Disponible en: http://canaltic.com/blog/?p=889#dao3.

9. CHAMORRO, A. E. B. *La interoperabilidad y los estándares abiertos, base del desarrollo de la Sociedad de la Información* de 2013]. Disponible en: http://www.coit.es/publicaciones/bit/bit161/36-39.pdf.

10. GUTIÉRREZ, S. I. Y. P. P. *Estándares e interoperabilidad en salud electrónica: Requisitos para una gestión sanitaria efectiva y eficiente* de 2013]. Disponible en: http://www.eclac.org/publicaciones/xml/4/45524/2011-797_W.440_-_Estandares_e_interoperabilidad_en_salud_electronica_WEB.pdf.

11. SHETH, A. P. *CHANGING FOCUS ON INTEROPERABILITY IN INFORMATION SYSTEMS: FROM SYSTEM, SYNTAX, STRUCTURE TO SEMANTICS* de 2013]. Disponible en: http://lsdis.cs.uga.edu/lib/download/S98-changing.pdf.

12. INDARTE, S. *Interoperabilidad* de 2013]. Disponible en: http://www.seis.es/documentos/informes/secciones/adjunto1/15_Interoperabilidad.pdf.

13. SILVA, E. A. T. *informática en salud estándares e interoperabilidad* de 2013]. Disponible en: http://www.slideshare.net/cheo.torres/4-informtica-en-salud-estndares-e-interoperabilidad.

14. SANDRA AGUIRRE, J. Q., JOAQUÍN SALVACHUA. *Mediadores e Interoperabilidad en Elearning* de 2013]. Disponible en: http://www.dit.upm.es/~saguirre/publications/virtualEduca2004.pdf.

15. CABEZUELO, A. S. *"Diseño de repositorios digitales interoperables"* de 2013]. Disponible en: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/5841/3/sarasaTFC0111memoria.pdf.

16. LAPUENTE, M. J. L. *Servicios Web* de 2013]. Disponible en: http://www.hipertexto.info/documentos/serv_web.htm.

17. GONZÁLEZ, B. *WSDL para la documentación de Servicios Web* de 2014]. Disponible en: http://www.desarrolloweb.com/articulos/1581.php.

18. SOAP. *Protocolo Simple de Acceso a Objetos* de 2013]. Disponible en: http://di002.edv.uniovi.es/~cueva/asignaturas/extension/2003/01-Panorama/ServiciosWeb.pdf.

19. CABRERA, Y. V. *Transferencia de estado representacional (REST): estilo de arquitectura para sistemas distribuidos de hipermedia* de 2014]. Disponible en: http://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0CCsQFjAC&url=http%3A%2F%2Fpublicaciones.uci.cu%2Findex.php%2FSC%2Farticle%2Fdownload%2F1076%2F700&ei=XnOYU8aKHIXlsASh3oHwAw&usg=AFQjCNEJbBVXxwrgxttuimhA-owUSymmBw&cad=rja.

20. GONZÁLEZ, B. *UDDI (Universal Description Discovery and Integration)* de 2014]. Disponible en: http://www.desarrolloweb.com/articulos/1589.php.

21. SANCHEZ, M. A. M. *Metodologías De Desarrollo De Software* de 2014]. Disponible en: http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html.

22. ALVAREZ, J. C. *Metodología RUP* de 2014]. Disponible en: http://es.slideshare.net/cortesalvarez/metodologa-rup.

23. SOFTWARE, I. D. *Guión Visual Paradigm for UML* de 2014]. Disponible en: http://www.ie.inf.uc3m.es/grupo/docencia/reglada/ls1y2/PracticaVP.pdf.

24. K., P. S. C. Y. N. H. *Tutorial de UML* de 2014]. Disponible en: http://users.dcc.uchile.cl/~psalinas/uml/introduccion.html.

25. ÁLVAREZ, M. A. *Introducción a CSS 3* de 2014]. Disponible en: http://www.desarrolloweb.com/articulos/introduccion-css3.html.

26. EGUILUZ, J. *Introducción a JavaScript* de 2014]. Disponible en: http://librosweb.es/javascript/.

27. OLSON, P. *Manual de PHP* de 2014]. Disponible en: http://www.php.net/manual/es/index.php.

28. ÁLVAREZ, M. A. *Introducción a PHP 5* de 2014]. Disponible en: http://www.desarrolloweb.com/articulos/1696.php.

29. W3SCHOOLS. *HTML4 y HTML5 Tutorial* de 2014]. Disponible en: http://www.w3schools.com/html/DEFAULT.asp.

30. JOHANSO. *Introducción al HTML5* de 2014]. Disponible en: http://www.tutosytips.com/dia-1-introduccion-a-html5/.

31. MARTINEZ, R. *Sobre PostgreSQL* de 2014]. Disponible en: http://www.postgresql.org.es/sobre_postgresql.

32. *Overview of new features in Apache 2.0* de 2014]. (Apache HTTP Server). Disponible en: https://httpd.apache.org/docs/2.2/new_features_2_0.html.

33. SANCHEZ, M. *Symfony guia definitiva* de 2014]. Disponible en: http://es.slideshare.net/nereosa/symfony-guia-definitiva.

34. NETBEANS. *Bienvenido a NetBeans* Disponible en: https://netbeans.org/index_es.html.

35. SYMFONY. *NetBeans ya incluye soporte para Symfony* de 2014]. Disponible en: http://symfony.es/noticias/2009/10/05/netbeans-ya-incluye-soporte-para-symfony/.

36. MARK OTTO, J. T. *Bootstrap 3, el manual oficial* de 2014]. Disponible en: http://librosweb.es/bootstrap_3/.

37. *Análisis y diseño de sistemas*. de 2014]. Disponible en: http://www.rena.edu.ve/cuartaEtapa/Informatica/Tema11.html.

38. OLIVERA, A. *Requerimientos funcionales y no funcionales* de 2014]. Disponible en: http://es.scribd.com/doc/37187866/Requerimientos-funcionales-y-no-funcionales.

39. LÓPEZ, D. S. *SISTEMA PARA EL CONTROL DEL USO DE LOS SOFTWARES EDUCATIVOS* [Consultado el: 15 de marzo de 2014]. (Modelo de Casos de Uso del Sistema). Disponible en: http://www.eumed.net/libros-gratis/2009c/585/585.zip.

40. ROJAS, C. *Fundamentos de Ingenería de Software* de 2014]. (Modelo de Análisis). Disponible en: http://es.slideshare.net/chiki.carito/modelado-del-anlisis.

41. JEISON PRADA, D. O., MERIBEL BELLO, KELLY ARÉBALO, CARLOS CASTILLO. *Diagrama de Colaboración* de 2014]. Disponible en: http://es.slideshare.net/still01/diagrama-de-colaboracionJeison

42. DAIRA FIGUEROA HIDALGO, Y. V. O. Y. V. M. F. *Propuesta de diseño para proyectos informáticos que utilizan Symfony como Framework Design for software projects that use Symfony as framework proposal* de 2014]. (Modelo del Diseño). Disponible en: http://publicaciones.uci.cu/index.php/SC/article/download/208/228%E2%80%8E.

43. FABIEN POTENCIER, F. Z. *Guía definitiva de Symfony* de 2014]. (Patrón MVC). Disponible en: http://asteinmetz.com.ar/weblog/wp-content/uploads/2008/11/symfony_guia_definitiva.pdf.

44. *Reutilización del Software*. de 2014]. (Patrones de diseño). Disponible en: http://siul02.si.ehu.es/~alfredo/iso/06Patrones.pdf.

45. ASTUDILLO, M. V. Y. H. *Fundamentos de Ingeniería de Software* de 2014]. Disponible en: http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf.

46. HERNANDEZ, L. *MODELO DE IMPLEMENTACIÓN* (UNIDAD 5 : MODELO DE IMPLEMENTACIÓN). Disponible en: http://ithleovi.blogspot.com/2013/06/unidad-5-modelo-deimplementacion-el.html.

47. LOVELLE, M. M. Y. J. M. C. *Pruebas de Calidad de Software* de 2014]. Disponible en: http://www.ecured.cu/index.php/Pruebas_de_Calidad_de_Software.

48. EGUILUZ, J. *Desarrollo web ágil con Symfony2* de 2014]. Disponible en: http://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0CCoQFjAC&url=http%3A%2F%2Fcrcprod.argusws.com%2Fdownload%2F5136201312451612 45.pdf%3Fdir%3D%252F..%252Fweb%252Fuploads%252Fdocumento%252F&ei=RWaiU92KPNPLsQTesoGQAg&usg=AFQjCNEpJoTedjEWw4LUP3HpX3nhkAwoiA&bvm=bv.69411363,d.cWc&cad=rja.

49. *Técnicas de pruebas*. de 2010]. (Caja Negra). Disponible en: http://indalog.ual.es/mtorres/LP/Prueba.pdf.

50. *Asegurando la calidad de software*. de 2014]. (Casos de uso vs. Casos de Prueba). Disponible en: http://testeandosoftware.com/casos-de-uso-vs-casos-de-prueba/.

Anexos

Anexo I. Explicación de cada método de SQI.

- Métodos para el manejo de sesiones:

- **Create Session():** Permite crear una sesión asociada a un usuario, para lo cual requiere del usuario y contraseña como parámetros, y retorna un identificador de sesión si el usuario es autenticado. Al usar este método se pueden producir algunos errores:
  - ✓ "WRONG_CREDENTIALS" si se envía un userID o una contraseña inválida.
  - ✓ "METHOD_FAILURE" es lanzado en caso de hay algo que no funciona correctamente en el uso del método.
- **Create Anonymous Session():** Permite crear una sesión no asociada a un usuario, y puede ser usado en un sistema que permite a todo las personas crear consultas. Al usar este método se pueden producir algunos errores:
  - ✓ "METHOD_FAILURE" es lanzado en caso de hay algo que no funciona correctamente en el uso del método.
- **Destroy Session():** Permite a la fuente que inició la sesión finalizar la sesión, para lo que es necesario eliminar el identificador de sesión sessionID. Al usar este método se pueden producir algunos errores:
  - ✓ “NO_SUCH_SESSION" si la sesión a eliminar no existe.
  - ✓ "METHOD_FAILURE" es lanzado en caso de hay algo que no funciona correctamente en el uso del método.

- Métodos para el manejo de consultas:

- **Set Query Language():** Permite a la fuente controlar la sintaxis usada en las instrucciones de la consulta por identificación del lenguaje de consulta. Los valores para el parámetro queryLanguageID con case-sensitivos. Al usar este método se pueden producir algunos errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID sea inválido.
  - ✓ QUERY_LANGUAGE_NOT_SUPPORTED si el lenguaje de consultas usado en la petición no está soportado por el destino.
  - ✓ "METHOD_FAILURE" si la operación falla por otras razones.

- **Set Maximum Number of Query Results():** Define el número máximo que una consulta puede producir, especificándose como un porcentaje. El parámetro maxQueryResults debe ser cero o más grande, de manera que si se configura a cero, se entiende que la fuente no limita el número de resultados devueltos. Al usar este método se pueden producir algunos errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID sea inválido.
  - ✓ INVALID_MAX_QUERY_RESULTS si un número invalido es proporcionado como valor del parámetro maxQueryResults. Esto puede suceder con números negativos o bien con valores que superan el límite de resultados que es capaz de devolver una fuente.
  - ✓ "METHOD_FAILURE" si la operación falla por otras razones.
- **Set Maximum Duration():** Permite a la fuente configurar un tiempo de espera en caso de que la consulta opera en una interface de consultas asíncronas. El parámetro que controla el tiempo de espera se expresa en milisegundos. Cuando un destino recibe una consulta asíncrona, puede retornar los resultados por un máximo de tiempo expresado en los milisegundos del parámetro maxDuration. Después de pasado este tiempo, un destino no puede usar el método queryRestultsListener. El valor del parámetro maxDuration debe ser cero o más grande. Si toma el valor de cero, se entiende que la fuente delega la gestión del tiempo de espera en el destino. El valor por defecto es cero. Al usar este método se pueden producir algunos errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID sea inválido.
  - ✓ INVALID_MAX_DURATION si un número invalid es proporcionado para el parámetro maxDuration.
  - ✓ "METHOD_FAILURE" si la operación falla por otras razones.
- **Set Results Format():** Permite a la fuente controlar el formato de los resultados retornados por el destino. El formato es especificado en el parámetro resultsFormat en forma de URI. Al usar este método se pueden producir algunos errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID sea inválido.
  - ✓ RESULTS_FORMAT_NOT_SUPPORTED cuando el format proporcionado en el parámetro resultsFormat no es soportador por el destino.
  - ✓ "METHOD_FAILURE" si la operación falla por otras razones.

➢ Métodos para las consultas síncronas:

- **Set Results Set Size():** Define el máximo número de resultados, que por defecto son 25. Si toma el valor de cero, entonces se entiende que quiere recuperar todos los resultados. Al usar este método se pueden producir algunos errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID sea inválido.
  - ✓ INVALID_RESULTS_SET_SIZE si un número invalido es provisto para el parámetro resultsSetSize.
  - ✓ QUERY_MODE_NOT_SUPPORTED en caso de que el destino no soporte consultas síncronas.
  - ✓ "METHOD_FAILURE" si la operación falla por otras razones.
- **Synchronous Query():** Envía una consulta al destino a través del parámetro queryStatement. Dentro de una sesión identificada por el targetSessionID, múltiples consultas pueden ser enviadas simultáneamente por invocación repetida de este método. El método retorna un conjunto de instancias de metadatos acordes con la consulta realizada. El parámetro startResults identifica la instancia de comienzo del conjunto de resultados, cuyo valor puede variar entre 1 y el número total de resultados. El índice del conjunto de resultados comienza en 1, y el número de resultados devueltos está controlado por setResultsSetSize y no puede ser mayor que el valor de setMaxQueryResults.Al usar este método se pueden producir algunos errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID sea inválido.
  - ✓ QUERY_MODE_NOT_SUPPORTED en caso de que el destino no soporte consultas síncronas.
  - ✓ INVALID_QUERY_STATEMENT si las instrucciones de la consulta no cumplen con la sintaxis del lenguaje de consultas.
  - ✓ INVALID_START_RESULT si un número invalid es proporcionado por startResult.
  - ✓ NO_MORE_RESULTS si startResult está configurado a cero y ningún resultado más es proporcionado.
  - ✓ "METHOD_FAILURE" si la operación falla por otras razones.
- **Get Total Results Count():** Retorna el número total de resultados existentes de una consulta. Al usar este método se pueden producir algunos errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID sea inválido.

- ✓ QUERY_MODE_NOT_SUPPORTED en caso de que el destino no soporte consultas síncronas.
- ✓ INVALID_QUERY_STATEMENT si las instrucciones de la consulta no cumplen con la sintaxis del lenguaje de consultas.
- ✓ METHOD_FAILURE si la operación falla por otras razones.

➢ Métodos para las consultas asíncronas:

- **Set Source Location():** Este método es llamado antes de una consulta sea enviada en modo asíncrono. El parámetro sourceLocation especifica la localización de fuente que espera los resultados, de manera que el destino pueda enviar los resultados. El sourceLocation es una cadena que puede resolver la localización de la queryResultsListener.Al usar este método se pueden producir algunos errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID sea inválido.
  - ✓ QUERY_MODE_NOT_SUPPORTED en caso de que el destino no soporte consultas asíncronas.
  - ✓ INVALID_SOURCE_LOCATION si la localización no puede ser resuelta.
  - ✓ METHOD_FAILURE si la operación falla por otras razones.
- **Asynchronous Query():** Permite a la fuente enviar una consulta al destino, mientras los resultados son retornados de manera asíncrona. La consulta es proporcionada a través del parámetro queryStatement. Un queryID proporcionado por la fuente es requerido con el fin de enlazar los resultados de la consulta con la consulta cuando son retornados. Usando queryIDs únicos es posible enviar un número arbitrario de consultas en cada sesión activa. La localización de la fuente que espera los resultados es necesaria y debe haber sido proporcionado antes de usar el método setSourceLocation. Debido a la naturaleza asíncrona del método, los resultados de la consulta podrían llegar desde consultas previas. La consulta es procesada y los resultados son enviados dentro de la franja de tiempo especificada en el método setMaxDuration. Al usar este método se pueden producir algunos errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID sea inválido.
  - ✓ QUERY_MODE_NOT_SUPPORTED en caso de que el destino no soporte consultas asíncronas.

- ✓ NO_SOURCE_LOCATION en caso de que ninguna localización haya sido especificada ante de enviar la consulta en la sesión.
- ✓ INVALID_QUERY_STATEMENT si las instrucciones de la consulta no cumplen con la sintaxis del lenguaje de consultas.
- ✓ METHOD_FAILURE si la operación falla por otras razones.

- **Query Results Listener():** Es un método iniciado por el destino que tiene como objetivo dirigir los resultados a la fuente. El parámetro queryID es usado para enlazar los resultados de la consulta a consulta previamente enviada. El parámetro queryResults genera un conjunto de resultados consistentes en una lista de instancias de metadatos, los cuales están formateados de acuerdo al esquema especificado por el método setResultsFormat. Al usar este método se pueden producir algunos errores:
  - ✓ INVALID_QUERY_RESULTS en caso de que el conjunto de resultados no puedan ser interpretados por la fuente.
  - ✓ NO_SUCH_QUERY en caso de que el queryID sea invalid.
  - ✓ METHOD_FAILURE si la operación falla por otras razones.

Anexo II. Explicación de cada método de SPI.

- ➢ Orientados a la publicación de recursos educativos

- **Método set Data Format():** Permite indicar el tipo de objetos que se va a almacenar en un repositorio. Dispone del parámetro setDataFormatID, que establece el formato de datos para objetos compuestos como los paquetes SCORM, AICC o IMS-QTI. Para objetos no compuesto, el parámetro indica el tipo de archivado y el formato de comprensión usado en el objeto. Cuando se usa éste método se pueden producir 3 tipos de fallos:
  - ✓ NO_SUCH_SESSION si el TargetSessionID es inválido.
  - ✓ DATA_FORMAT_NOT_SUPPORTED si el formato usado en la petición no es soportado.
  - ✓ METHOD_FAILURE si la operación falla por otras razones.
- **Método Set Source Location():** Se usa antes de que un objeto de aprendizaje sea enviado en modo “por referencia”. Dispone del parámetro sourceLocation que permite resolver la localización

de la fuente del método notifyRetrievalStatus , y así enviar un mensaje de reconocimiento. Cuándo se usa este método se pueden producir 4 tipos de errores:

- ✓ NO_SUCH_SESSION en caso de que el targetSessionID sea inválido.
- ✓ SUBMISSION_MODE_NOT_SUPPORTED si no se soportan envíos en modo por referencia.
- ✓ INVALID_SOURCE_LOCATION si la localización es incorrecta o no puede ser parseado.
- ✓ METHOD_FAILURE si la operación falla por otra razón.

- **Método Submit Resource():** Este método permite publicar un objeto de aprendizaje en el repositorio mediante dos modos diferentes, bien por referencia o bien por valor. El identificador que se asocia al objeto que se publica puede ser generado por la fuente o por el destino, existiendo así dos métodos diferentes por cada tipo de modo de publicación, uno en el que se especifica como parámetro el identificador y otro que no. Cuándo se usa este método se pueden producir algunos de los siguientes tipos de errores:
  - ✓ NO_SUCH_SESSION en caso de que el TargetSessionID es inválido.
  - ✓ METHOD_NOT_SUPPORTED si el método de publicación no es soportado por el destino.
  - ✓ INSUFFICIENT_CREDENTIALS si el usuario no es autorizado a ejecutar este método.
  - ✓ INVALID_RESOURCE_IDENTIFIER cuando el identificador provisto por la fuente ya existe o tiene una estructura no soportada.
  - ✓ NO_SOURCE_LOCATION en caso de que ninguna localización para la fuente haya sido especificada antes de publicar el recurso.
  - ✓ METHOD_FAILURE si la operación falla por alguna otra razón.
- **Método Notify Retrieval Status():** Después de invocar un método submitResourceByReference, la fuente espera asíncronamente una notificación de transmisión completada. Así el destino usa el método notifyRetrievalStatus para informar a la fuente del estado de la transferencia. Después de completarse la transferencia, se debe enviar una notificación con estado “COMPLETO” a la fuente. Si el mensaje no puede ser entregado, entonces el destino debería borrar el recurso para el cual envió la notificación. Cuándo se usa este método se pueden producir 2 tipos de errores:
  - ✓ NO_SUCH_IDENTIFIER si un identificador invalido es enviado.
  - ✓ METHOD_FAILURE si la operación falla por cualquier otra razón.

- **Método Delete Resource():** Este método permite borrar un objeto de aprendizaje del destino. El borrado de un recurso no tiene un efecto cascada sobre las instancias de metadatos que están asociadas al recurso. Por tanto la fuente debe explicitar el borrado de los metadatos asociados. Cuándo se usa este método se pueden producir 7 tipos de errores:
  - ✓NO_SUCH_SESSION en caso de que el targetSessionID sea inválido.
  - ✓INVALID_RESOURCE_IDENTIFIER si el identificador tiene una estructura inválida.
  - ✓RESOURCE_DOES_NOT_EXIST si el objeto de aprendizaje asociado con el identificador no existe.
  - ✓INSUFFICIENT_CREDENTIALS si el usuario no autoriza el borrado del objeto de aprendizaje.
  - ✓DELETION_NOT_ALLOWED. El destino no permite borrar objetos de aprendizaje después de que ellos son publicados.
  - ✓METADATA_ASSOCIATED. El objeto de aprendizaje está asociado a instancias de metadatos. Disocia primero metadatos y objetos de aprendizaje.
  - ✓METHOD_FAILURE si la operación falla por otras razones.

➢ Orientados a la publicación de metadatos

- **Método Associate():** En un repositorio de objetos de aprendizaje puede contener tanto un almacén para los objetos y otro para los metadatos de manera independiente, de manera que si la instancia de metadatos se publica primero, entonces no puede referenciar al objeto de aprendizaje, hasta que este se publique. Así el método de asociación, permite añadir esta referencia. Sin embargo, el método no chequea la consistencia entre la instancia de metadatos y el objeto de aprendizaje al que se referencia. Cuando se usa éste método se pueden producir 6 tipos de fallos:
  - ✓ NO_SUCH_SESSION si el TargetSessionID es invalido.
  - ✓ INVALID_RESOURCE_IDENTIFIER si ningún recurso es identificado por el parámetro resourceIdentifier.
  - ✓ INVALID_METADATA_IDENTIFIER si ninguna instancia de metadatos es identificada por el parámetro metadataIdentifier.
  - ✓ RESOURCE_NOT_RETRIEVED en caso de recuperación por referencia, puede ocurrir que la recuperación de un recurso no se haya completado.
  - ✓ INSUFFICIENT_CREDENTIALS si el usuario no está autorizado a ejecutar este método.

- ✓ METHOD_FAILURE si la operación falla por otras razones.

- **Método Dissociate():** Permite que una asociación pueda ser deshecha. Es necesario cuando una instancia de metadatos o un objeto de aprendizaje va a ser borrado. Cuando se usa éste método se pueden producir 6 tipos de fallos:
  - ✓ NO_SUCH_SESSION si el TargetSessionID es invalido.
  - ✓ INVALID_RESOURCE_IDENTIFIER si ningún recurso es identificado por el parámetro resourceIdentifier.
  - ✓ INVALID_METADATA_IDENTIFIER si ninguna instancia de metadatos es identificada por el parámetro metadataIdentifier.
  - ✓ INSUFFICIENT_CREDENTIALS si el usuario no está autorizado a ejecutar este método.
  - ✓ NOT_ASSOCIATED cuando se intenta disociar un recurso y una instancia de metadatos que no estabas asociadas.
  - ✓ METHOD_FAILURE si la operación falla por otras razones.

Anexo III. Descripción de los CU.

Tabla 8. Descripción del CU: Gestionar repositorio para OAI_PMH.

Objetivo	Administrar los repositorios para colectar haciendo uso del protocolo OAI-PMH.
Actores	Administrador
Resumen	El caso de uso inicia cuando el administrador selecciona en el menú la opción Repositorio OAI-PMH. El sistema muestra un listado de los repositorios y las opciones de agregar nuevo, editar y borrar un repositorio. El administrador realiza las operaciones que desee y de esta forma se gestionan los repositorios. El caso de uso termina.
Complejidad	Media
Prioridad	Alta
Precondiciones	El usuario debe estar autenticado como administrador y tener el rol de administrador.
Postcondiciones	Ha quedado adicionado, editado o eliminado uno o varios repositorios.

Flujo de eventos		
Flujo básico <Nombre del flujo básico>		
	Actor	**Sistema**
1	Selecciona la opción: *Repositorios OAI-PMH.*	1.1. Muestra una interfaz con los repositorios existentes mostrando de cada uno los siguientes datos: - URL Base (Url del repositorio). - Name (Nombre del repositorio). - Description (Breve descripción del repositorio). - Update At (Fecha y hora en la que se realiza la consulta al repositorio). - Catalog Language (Lenguaje de catalogación). - Type (Tipo de repositorio). - Version (Versión del repositorio). 1.2. Permite por cada repositorio realizar las acciones: - Agregar un nuevo repositorio (Ver sección 1: Agregar nuevo repositorio). - Editarlo (Ver sección 2: Editar repositorio). - Borrarlo (Ver sección 3: Borrar repositorio). 1.3. El botón “Cancelar” para realizar las acciones correspondientes.
2	Presiona el botón ¨Cancelar¨.	2.1. Termina el caso de uso.
Sección 1: “Agregar nuevo repositorio”		
Flujo básico “Agregar nuevo”		
	Actor	**Sistema**
2	Selecciona la opción: *Agregar nuevo*.	2.1. Muestra un formulario para que se introduzca los datos del nuevo repositorios, con OAI-PMH para ello debe ingresar los siguientes datos:

		- URL Base (Tipo Dato: String). - Name (Tipo Dato: String). - Description (Tipo Dato: String). - Update At (Tipo Dato: Datatime). - Catalog Language (Tipo Dato: String). - Type (Tipo Dato: String). - Version (Tipo Dato: String). - Los botones “Adicionar” y “Cancelar” para realizar las acciones correspondientes.
3	Ingresa los datos y hace clic en el botón *Aceptar*.	3.1. Muestra el mensaje de notificación: *“Elemento creado satisfactoriamente”*. 3.2. Guarda el repositorio en la base de datos y regresa al listado de los repositorios existentes. 3.3. Termina el caso de uso.
Flujos alternos		
Evento 3.a Introducir datos incorrectos		
	Actor	**Sistema**
3	Ingresa los datos y hace clic en el botón Aceptar” con dato incorrecto.	3.1. Muestra el mensaje de error: *“Se ha producido un error durante la creación del elemento”*. 3.2. Diferencia los campos en rojo, donde están los datos incorrectos. 3.3. Permite que sean modificados los datos introducidos incorrectamente. 3.4. Se mantiene en ejecución el flujo básico de eventos para el paso 2.
Evento 3.b Campos vacíos		
3	Ingresa todos los datos y hace clic en el botón *Aceptar* con campos vacíos.	3.1. Muestra el mensaje de error*: “Existen campos en blanco”*. 3.2. Diferencia en rojo los campos donde están los datos faltantes.

		3.3. Permite que sean entrados los datos faltantes.
Evento 3.c Cancelación de acción		
3	Hace clic en el botón *Cancelar*.	3.1 Termina la acción de adicionar repositorio OAI-PMH.
Sección 2: "Editar repositorio"		
Flujo básico "Editar"		
	Actor	**Sistema**
2	Selecciona la opción: ¨Editar¨	2.1 Muestra una interfaz con los datos que pueden modificarse del repositorio: - URL Base (Tipo Dato: String). - Name (Tipo Dato: String). - Description (Tipo Dato: String). - Update At (Tipo Dato: Datatime). - Catalog Language (Tipo Dato: String). - Type (Tipo Dato: String). - Version (Tipo Dato: String). - Los botones "Adicionar" y "Cancelar" para realizar las acciones correspondientes.
3	Modifica los datos necesarios y hace clic en el botón *Aceptar*.	3.1. Se muestra el mensaje de notificación: "*Elemento actualizado satisfactoriamente"*. 3.2. Termina el caso de uso
Flujos alternos		
Evento 3.a Introducir datos incorrectos		
	Actor	**Sistema**
3.	Modifica los datos necesarios y hace clic en el botón *Aceptar* con datos incorrecto.	3.1. Muestra mensaje de error: *"Se ha producido un error durante la actualización del elemento".* 3.2. Diferencia los campos en rojo, donde están los datos incorrectos. 3.3. Permite que sean modificados los datos introducidos incorrectamente.

		3.4. Se mantiene en ejecución el flujo básico de eventos para el paso 2.
Evento 3.b Campos Vacíos		
3.	Modifica los datos y hace clic en el botón *Aceptar* con campos vacíos.	3.1. Muestra el mensaje de error*: "Existen campos en blanco"*. 3.2. Diferencia en rojo los campos donde están los datos faltantes. 3.3. Permite que sean entrados los datos faltantes.
Evento 3.c Cancelación de acción		
3.	Hace clic en el botón *Cancelar*.	3.1 Termina la acción de editar repositorio OAI-PMH.
Sección 3: "Borrar repositorio"		
Flujo básico "Borrar"		
	Actor	**Sistema**
2	Selecciona la opción: *Borrar*	2.1 Muestra el mensaje*: "¿Está seguro que quiere borrar el elemento seleccionado?"*. 2.2. Muestra los botones "Si" y "No" para las acciones correspondientes.
3	Selecciona la opción: *Si*	3.1. Muestra el mensaje: "*Elemento eliminado satisfactoriamente".* 3.2. Termina el caso de uso.
Flujos alternos		
Evento 3.a Editar		
	Actor	**Sistema**
3	Hace clic en el botón: ¨No¨	3.1 Regresa a la interfaz del listado de repositorios para buscar con OAI-PMH. 3.2 Termina el caso de uso.
Relaciones	**CU Incluidos**	
	CU Extendidos	

Requisitos no funcionales	
Asuntos pendientes	

Tabla 9. Descripción del CU: Actualizar catálogo.

Objetivo	Actualizar todos los registros de cada repositorio.	
Actores	CRON(tarea programada)	
Resumen	Este CU inicia cuando el administrador solicita los metadatos de los registros de otros repositorios, eliminando las irregularidades de los registros y finaliza cuando se guarda el catálogo.	
Complejidad	Media	
Prioridad	Alta	
Precondiciones	Tiene que estar configurado el CRON, registrada la información del repositorio que se quiere actualizar.	
Postcondiciones	Queda actualizado el catálogo de un repositorio.	
Flujo de eventos		
Flujo básico <Nombre del flujo básico>		
	Actor	**Sistema**
1	Selecciona la opción *Recolectar*.	1. Obtiene los catálogos de metadatos que se encuentran en otros repositorios. 3. Guarda la información recopilada. 4. Muestra un listado con los catálogos recolectados. 5. Termina el caso de uso.
Flujos alternos		
Evento 1.a No existen catálogos		
	Actor	**Sistema**
1	Selecciona la opción *Recolectar* y no existen catálogos.	2. Muestra el mensaje: *"No existen datos para mostrar"*.

Relaciones	**CU Incluidos**	
	CU Extendidos	
Requisitos no funcionales		
Asuntos pendientes		

Tabla 10. Descripción del CU: Ofrecer catálogo.

Objetivo	Brindar el catálogo, a través del protocolo OAI-PMH.	
Actores	Repositorio externo.	
Resumen	Este CU inicia cuando el administrador le solicita el catálogo. Finaliza cuando se visualiza el catálogo solicitado.	
Complejidad	Media	
Prioridad	Alta	
Precondiciones	Deben existir recursos educativos en el repositorio.	
Postcondiciones	Otros repositorios pueden obtener los catálogos brindados a través del protocolo OAI-PMH.	
Flujo de eventos		
Flujo básico <Nombre del flujo básico>		
	Actor	**Sistema**
1	Selecciona la opción *Metadatos Recolectados*.	1.1. Ofrece un listado de los catálogos. 1.2. Termina el caso de uso.
Flujos alternos		
Evento 1.a No existen catálogos		
	Actor	**Sistema**
1	Selecciona la opción *Metadatos Recolectados* y no hay catálogos recolectados.	1.1. Muestra el mensaje: *"No existen datos para mostrar"*.
	CU	

	Extendidos	
Requisitos no funcionales		
Asuntos pendientes		

Tabla 11. Descripción del CU: Gestionar catálogo.

Objetivo	Administrar los formatos de catalogación soportados por el repositorio.	
Actores	Administrador	
Resumen	El caso de uso inicia cuando el administrador selecciona en el menú la opción Catálogo. El sistema muestra las opciones de agregar nuevo, editar, borrar y un listado de los catálogos. El administrador realiza las operaciones que desee y de esta forma se gestionan los catálogos. El caso de uso termina.	
Complejidad	Media	
Prioridad	Alta	
Precondiciones	El usuario debe estar autenticado como administrador y tener el rol de administrador.	
Postcondiciones	Ha quedado adicionado, editado o eliminado uno o varios catálogos.	
Flujo de eventos		
Flujo básico "Gestionar catálogo"		
	Actor	**Sistema**
1	Selecciona la opción: *Catálogos.*	1.1. Muestra una interfaz con los catálogos existentes mostrando de cada uno los siguientes datos: - metadataPrefix (Prefijo del catálogo). - metadataNamespace (Nombre del catálogo). - schema (El esquema del catálogo). 1.2. Permite por cada repositorio realizar las acciones:

		- Agregar un nuevo catálogo (Ver sección 1: Agregar nuevo catálogo). - Editarlo (Ver sección 2: Editar catálogo). - Borrarlo (Ver sección 3: Borrar catálogo). 1.3. El botón "Cancelar" para realizar las acciones correspondientes.
2	Hace clic en el botón ¨Cancelar¨.	2.1. Termina el caso de uso.
Sección 1: "Agregar nuevo catálogo"		
Flujo básico "Gestionar catálogo"		
	Actor	**Sistema**
2	Selecciona la opción: *Agregar nuevo*.	2.1. Muestra un formulario para que se introduzca los datos del nuevo catálogo, para ello debe ingresar los siguientes datos: - metadataPrefix (Tipo Dato: String). - metadataNamespace (Tipo Dato: String). - schema (Tipo Dato: String). - Los botones "Aceptar" y "Cancelar" para realizar las acciones correspondientes.
3	Ingresa todos los datos y hace clic en el botón *Aceptar.*	3.1. Muestra el mensaje de notificación: *"Elemento creado satisfactoriamente"*. 3.2. Guarda el catálogo en la base de datos y regresa al listado de los catálogos existentes. 3.3. Termina el caso de uso.
Flujos alternos		
Evento 3.a Introducir datos incorrectos		
	Actor	**Sistema**
3.	Ingresa todos los datos y hace clic en el botón *Aceptar* con datos incorrectos.	3.1. Muestra el mensaje de error: *"Se ha producido un error durante la creación del elemento"*. 3.2. Diferencia los campos en rojo, donde están los

		datos incorrectos. 3.3. Permite que sean modificados los datos introducidos incorrectamente. 3.4. Se mantiene en ejecución el flujo básico de eventos para el paso 2.
Evento 3.b Campos vacíos		
3	Ingresa todos los datos y hace clic en el botón *Aceptar* con campos vacíos.	3.1. Muestra el mensaje de error*: "Existen campos en blanco"*. 3.2. Diferencia en rojo los campos donde están los datos faltantes. 3.3. Permite que sean entrados los datos faltantes.
Evento 3.c Cancelación de acción		
3	Hace clic en el botón *Cancelar*.	3.1. Termina la acción de adicionar formato de catálogo.
Sección 2: "Editar catálogo"		
Flujo básico "Gestionar catálogo"		
	Actor	**Sistema**
2	Selecciona la opción: *Editar*.	2.1 Muestra una interfaz con los datos que pueden modificarse del catálogo: - Url Authentication (Tipo Dato: String). - Url Search (Tipo Dato: String). - metadataPrefix (Tipo Dato: String). - metadataNamespace (Tipo Dato: String). - schema (Tipo Dato: String). - Los botones "Aceptar" y "Cancelar" para realizar las acciones correspondientes.
3	Modifica los datos necesarios y hace clic en el botón *Aceptar*.	3.1. Se muestra el mensaje de notificación: "*Elemento actualizado satisfactoriamente"*. 3.2. Termina el caso de uso
Flujos alternos		

Evento 3.a Introducir datos incorrectos		
	Actor	**Sistema**
3.	Modifica los datos necesarios y hace clic en el botón *Aceptar* con datos incorrecto.	3.1. Muestra mensaje de error: *"Se ha producido un error durante la actualización del elemento".* 3.2. Diferencia los campos en rojo, donde están los datos incorrectos. 3.3. Permite que sean modificados los datos introducidos incorrectamente. 3.4. Se mantiene en ejecución el flujo básico de eventos para el paso 2.
Evento 3.b Campos vacíos		
3	Modifica los datos y hace clic en el botón *Aceptar* con campos vacíos.	3.1. Muestra el mensaje de error*: "Existen campos en blanco"*. 3.2. Diferencia en rojo los campos donde están los datos faltantes. 3.3. Permite que sean entrados los datos faltantes.
Evento 3.c Cancelación de acción		
3	Hace clic en el botón *Cancelar*.	3.1. Termina la acción de modificar formato de catálogo.
Sección 3: "Borrar catálogo"		
Flujo básico "Gestionar catálogo"		
	Actor	**Sistema**
2	Selecciona la opción: *Borrar*	2.1 Muestra el mensaje*: "¿Está seguro que quiere borrar el elemento seleccionado?"*. 2.2. Muestra los botones "Si" y "No" para las acciones correspondientes.
3	Selecciona la opción: *Sí*	3.1. Muestra el mensaje: "*Elemento eliminado satisfactoriamente".* 3.2. Termina el caso de uso.
Flujos alternos		

Evento 3.a Cancelación de acción			
	Actor	**Sistema**	
3	Hace clic en el botón: ¨No¨	3.1 Regresa a la interfaz del listado de catálogo. 3.2 Termina el caso de uso.	
Relaciones	**CU Incluidos**		
	CU Extendidos		
Requisitos no funcionales			
Asuntos pendientes			

Anexo IV. Diagrama de clases del análisis.

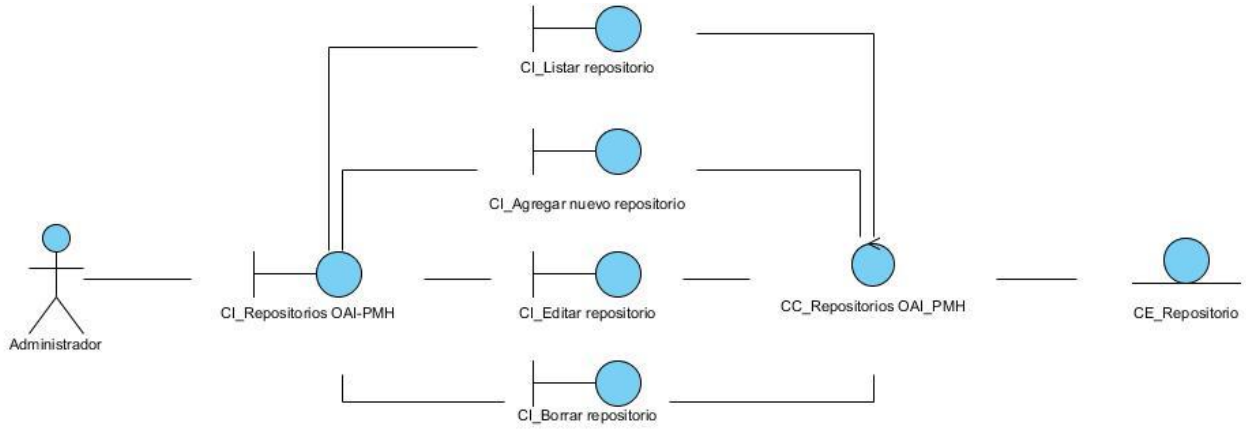


Figura 22. Diagrama de clase del CU: Gestionar repositorio para OAI-PMH.

Figura 23. Diagrama de clase del CU: Actualizar catálogo.

Figura 24. Diagrama de clase del CU: Ofrecer catálogo.

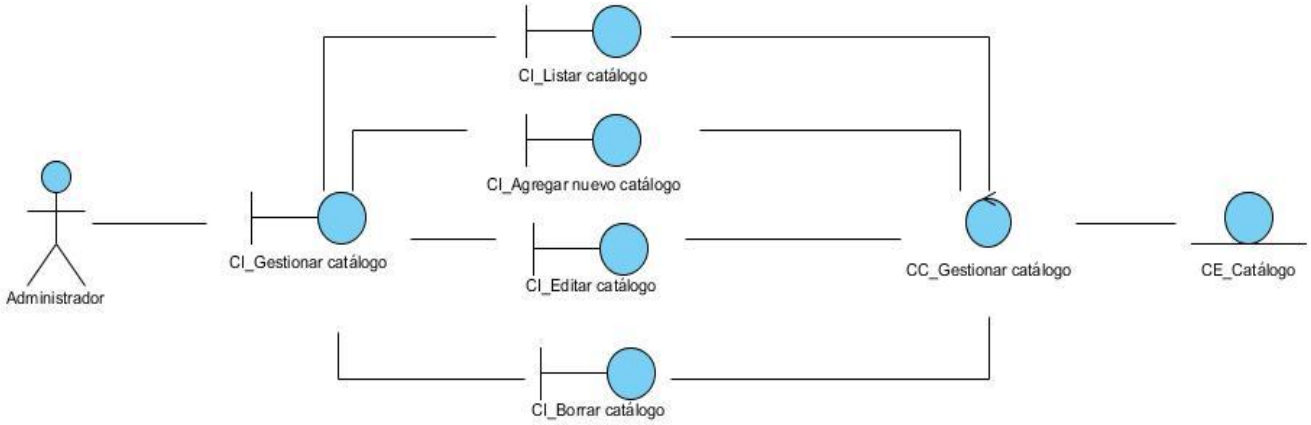


Figura 25. Diagrama de clase del CU: Gestionar catálogo.

Anexo V. Diagramas de Colaboración.

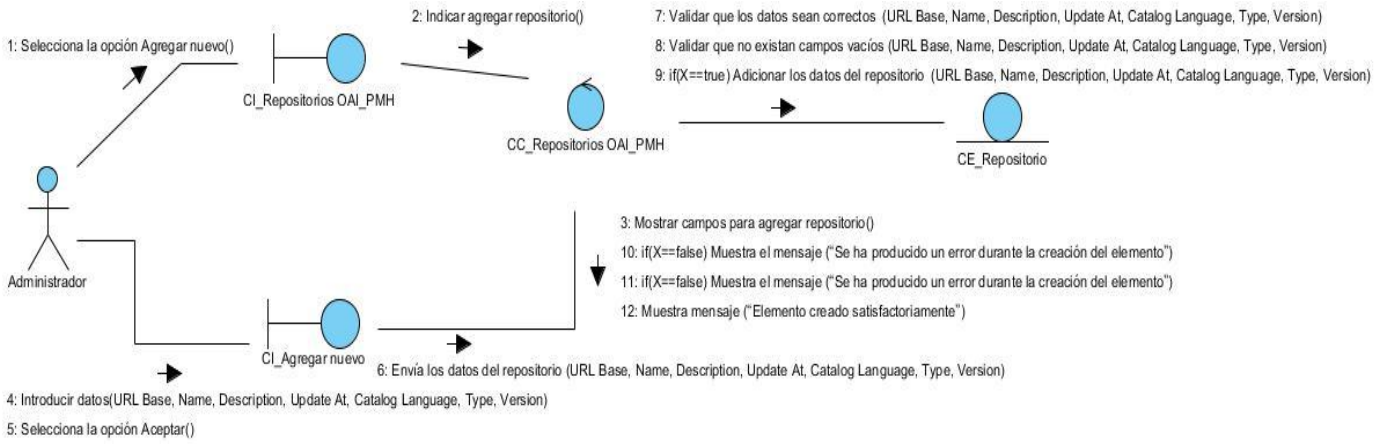


Figura 26. DC del CU Gestionar repositorio para OAI-PMH. Sección agregar repositorio.

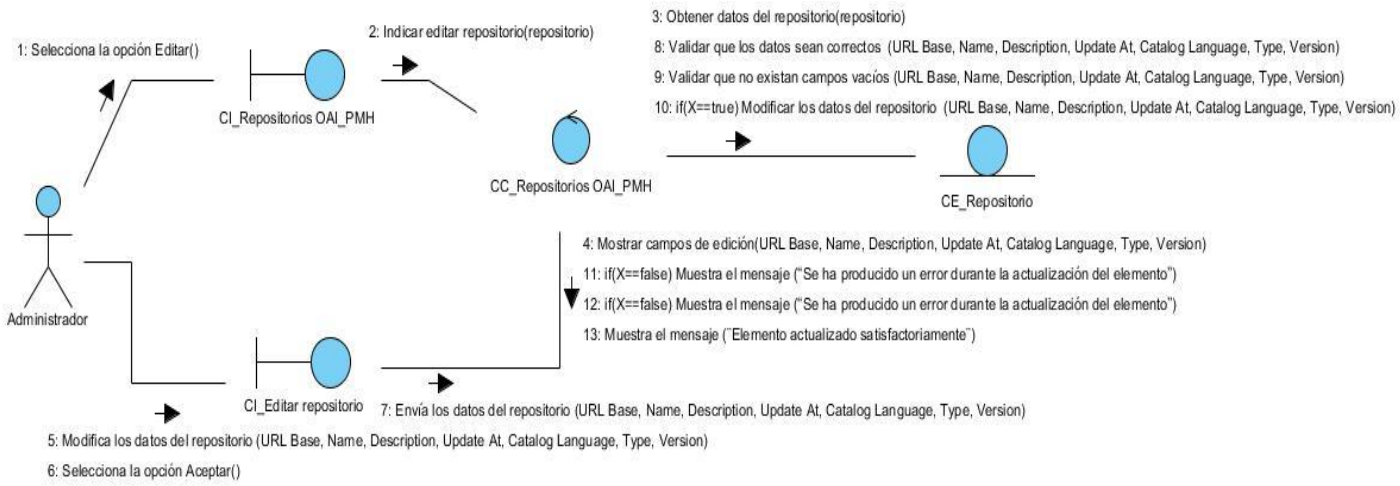


Figura 27. DC del CU Gestionar repositorio para OAI-PMH. Sección editar repositorio.

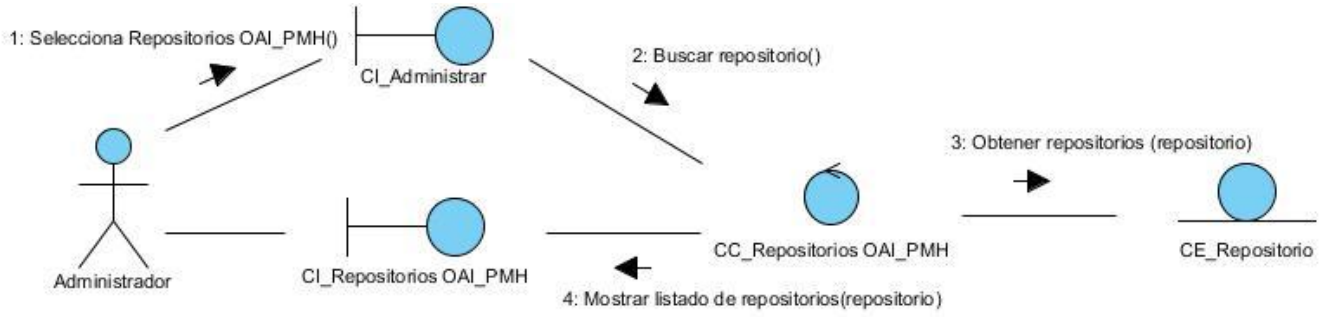


Figura 28. DC del CU Gestionar repositorio. Sección listar repositorio.

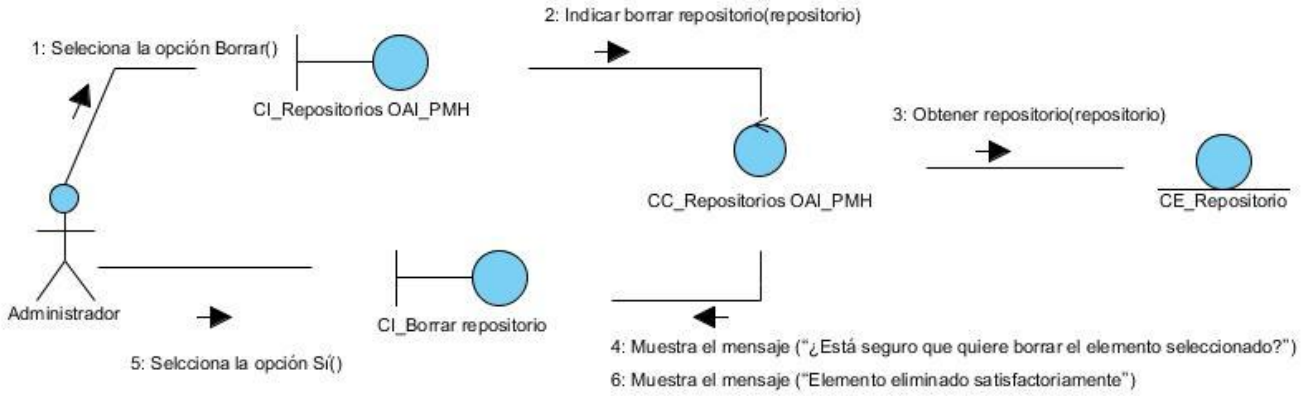


Figura 29. DC del CU Gestionar repositorio para OAI-PMH. Sección borrar repositorio.

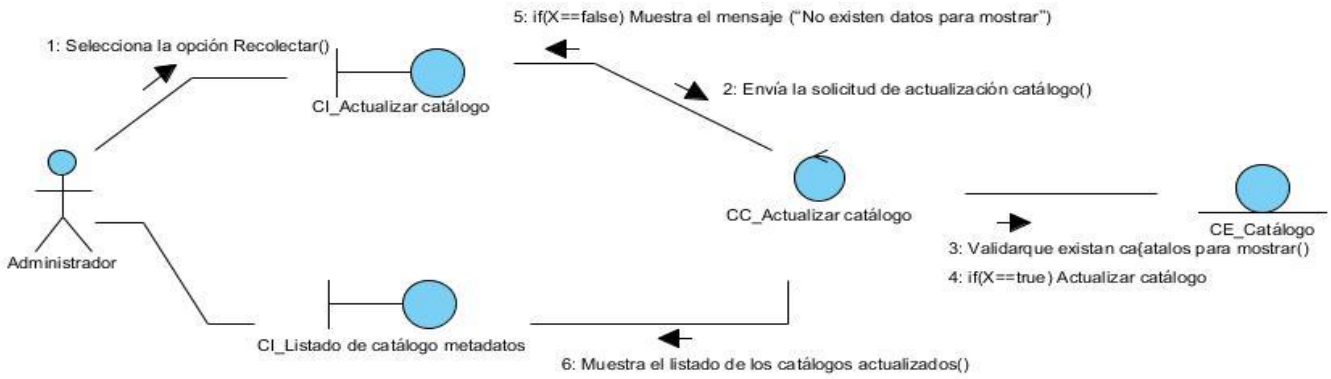


Figura 30. DC del CU Actualizar catálogo.

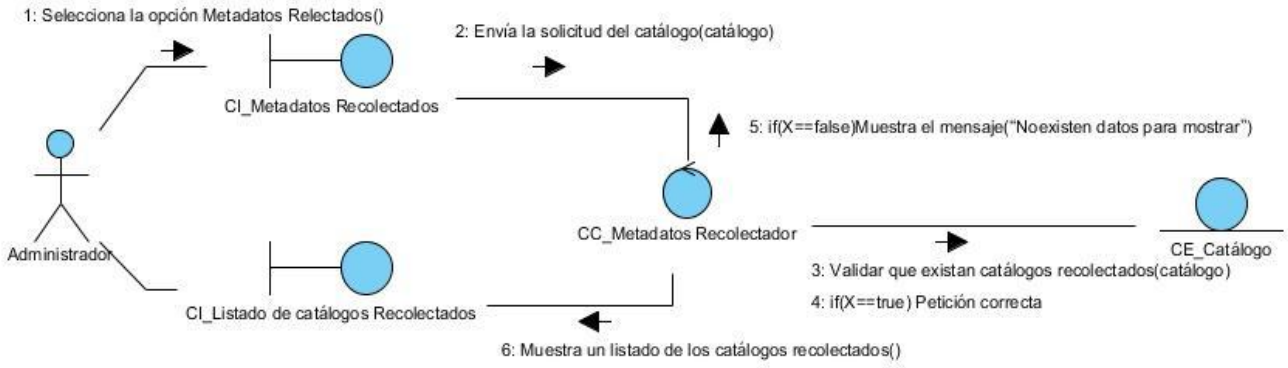


Figura 31. DC del CU Ofrecer catálogo.

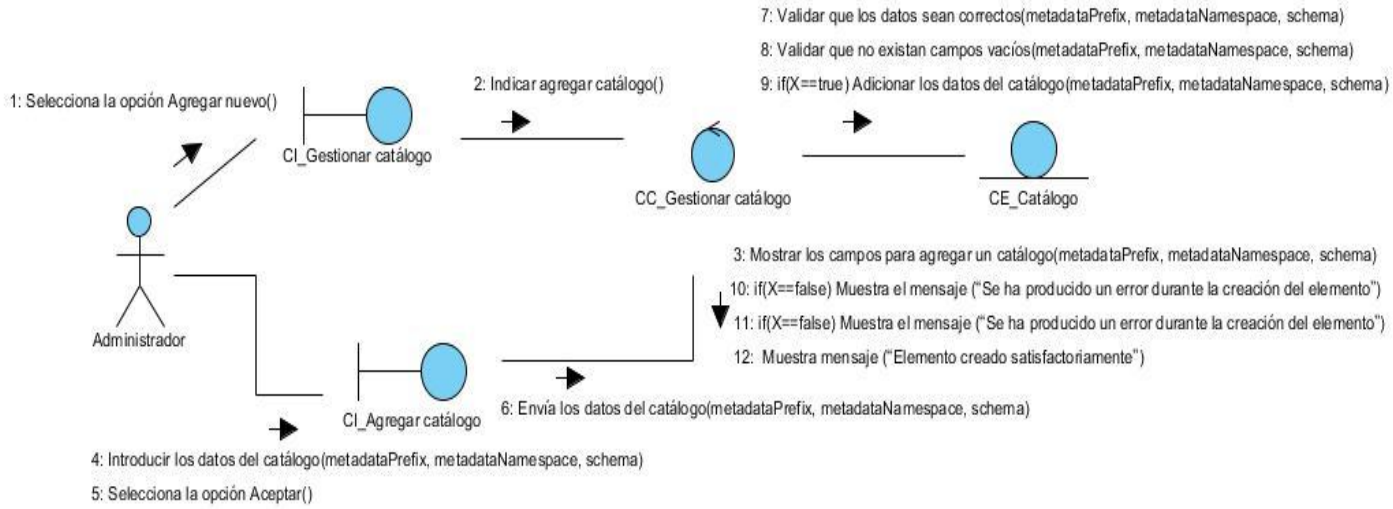


Figura 32. DC del CU Gestionar catálogo. Sección agregar catálogo.

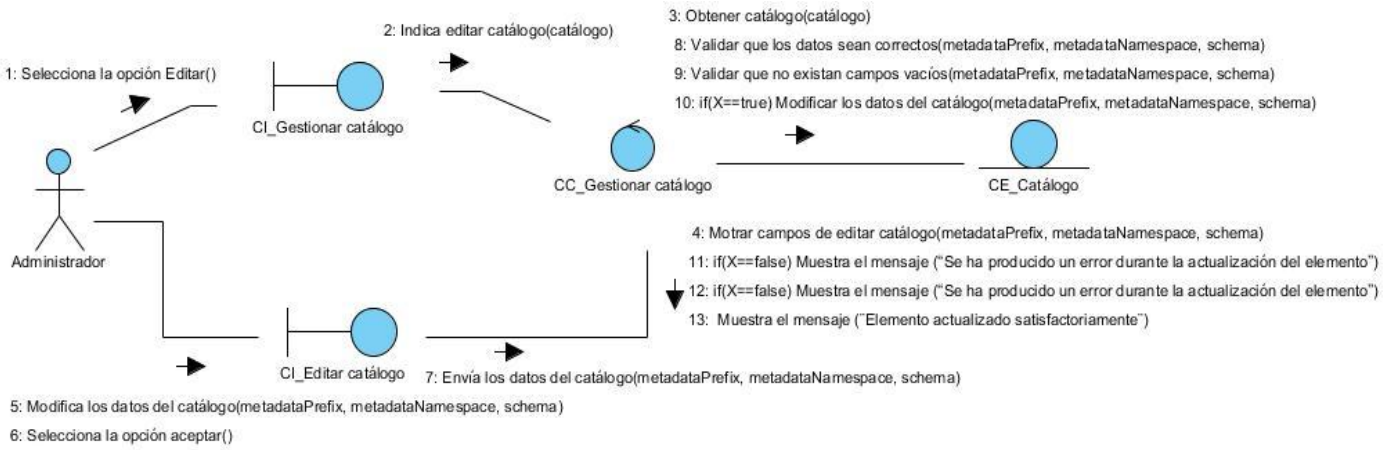


Figura 33. DC del CU Gestionar catálogo. Sección editar catálogo.

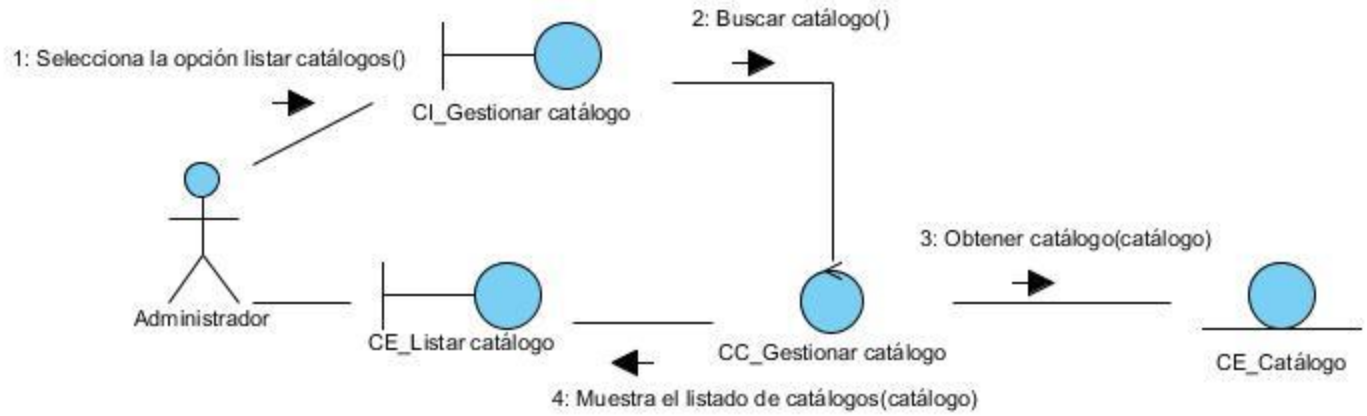


Figura 34. DC del CU Gestionar catálogo. Sección listar catálogo.

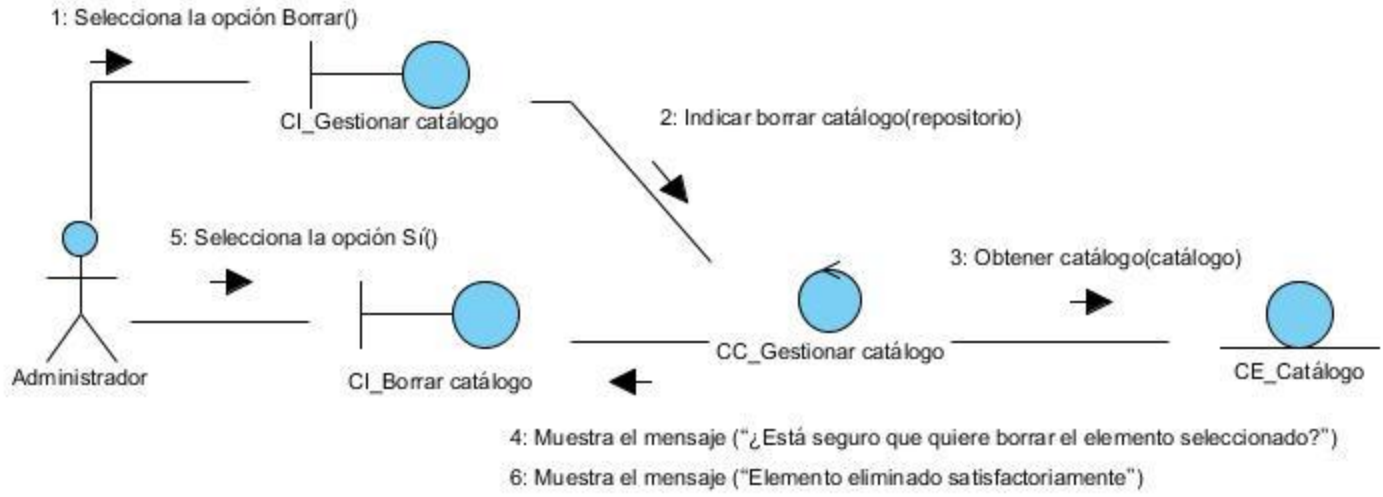


Figura 35. DC del CU Gestionar catálogo. Sección borrar catálogo.

Anexo VI. Diagramas de clase del diseño

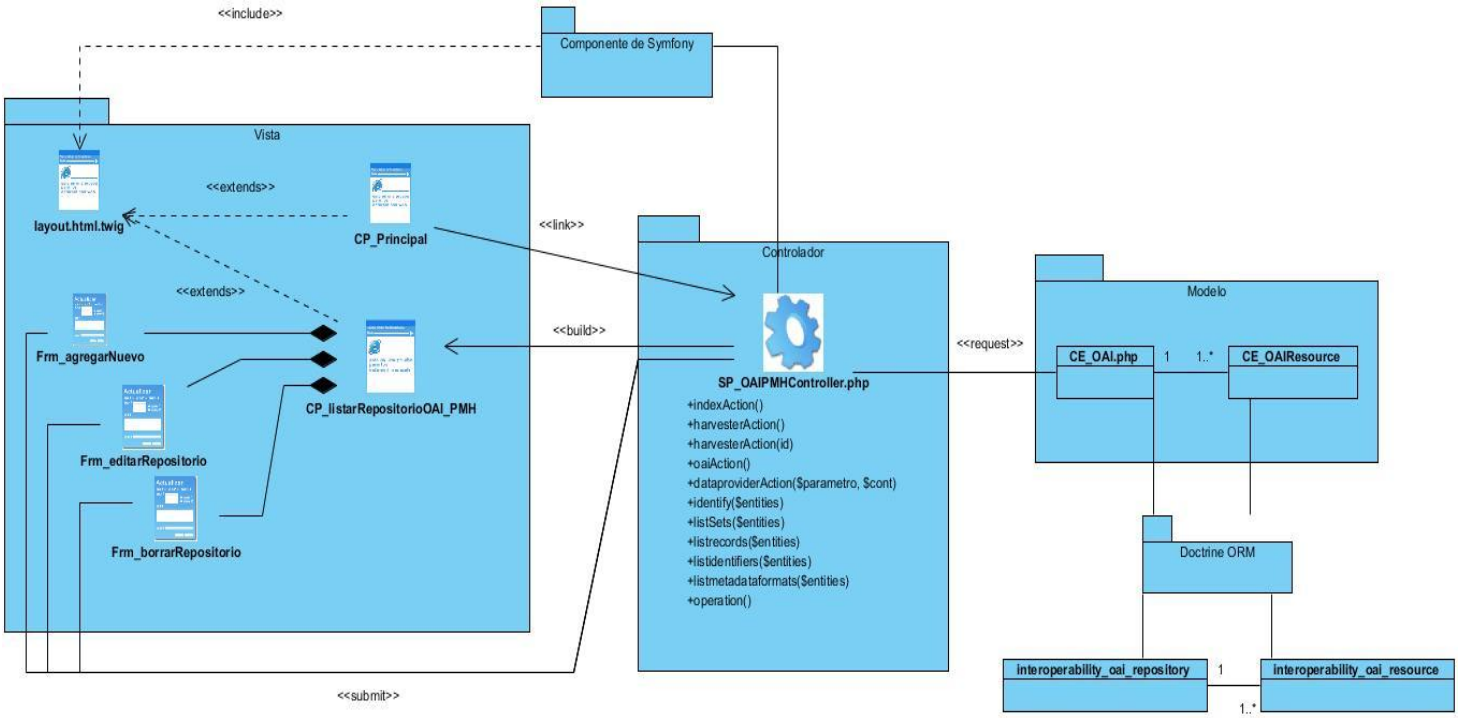


Figura 36. DCD del CU: Gestionar repositorio para OAI-PMH.

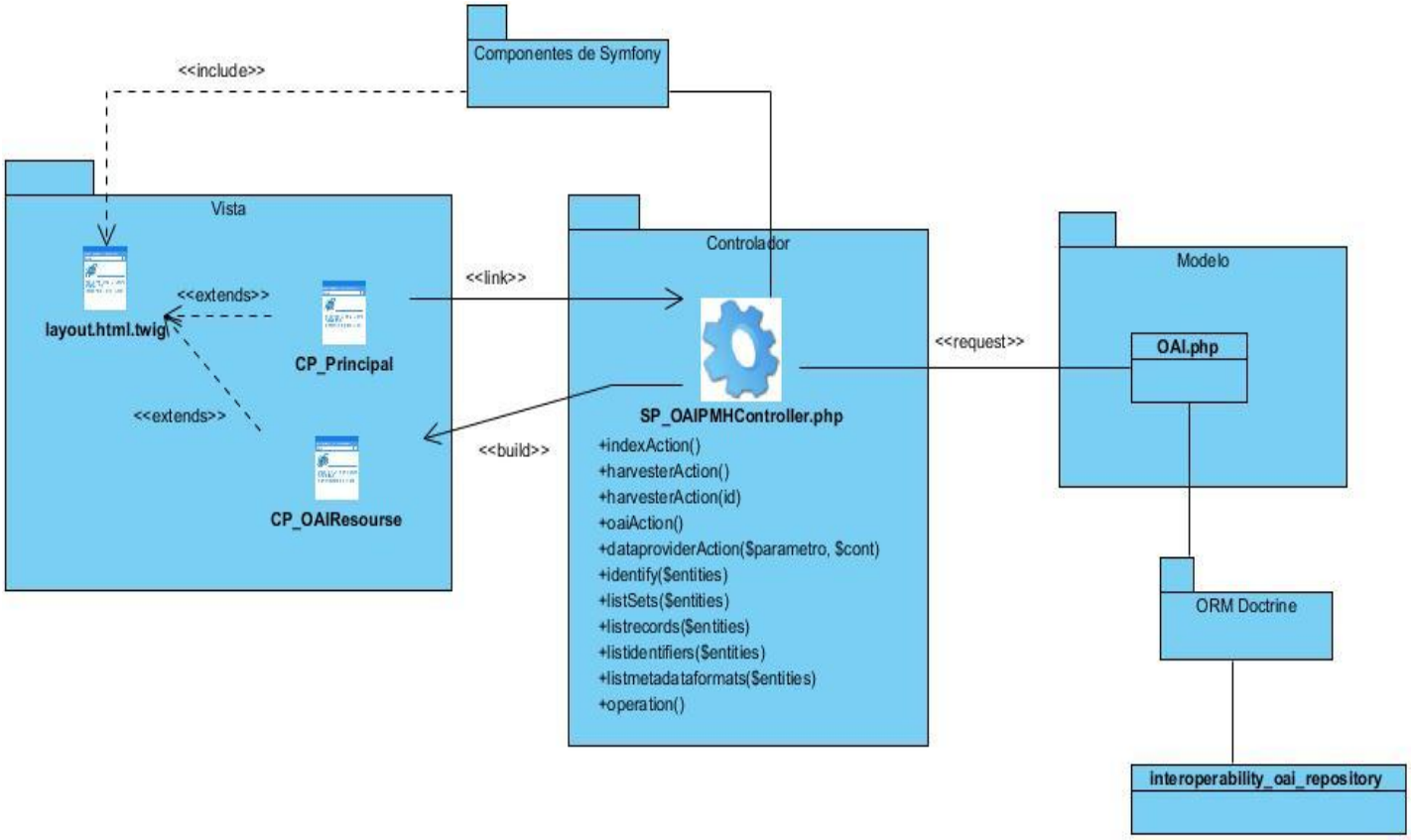


Figura 37. DCD del CU: Actualizar catálogo

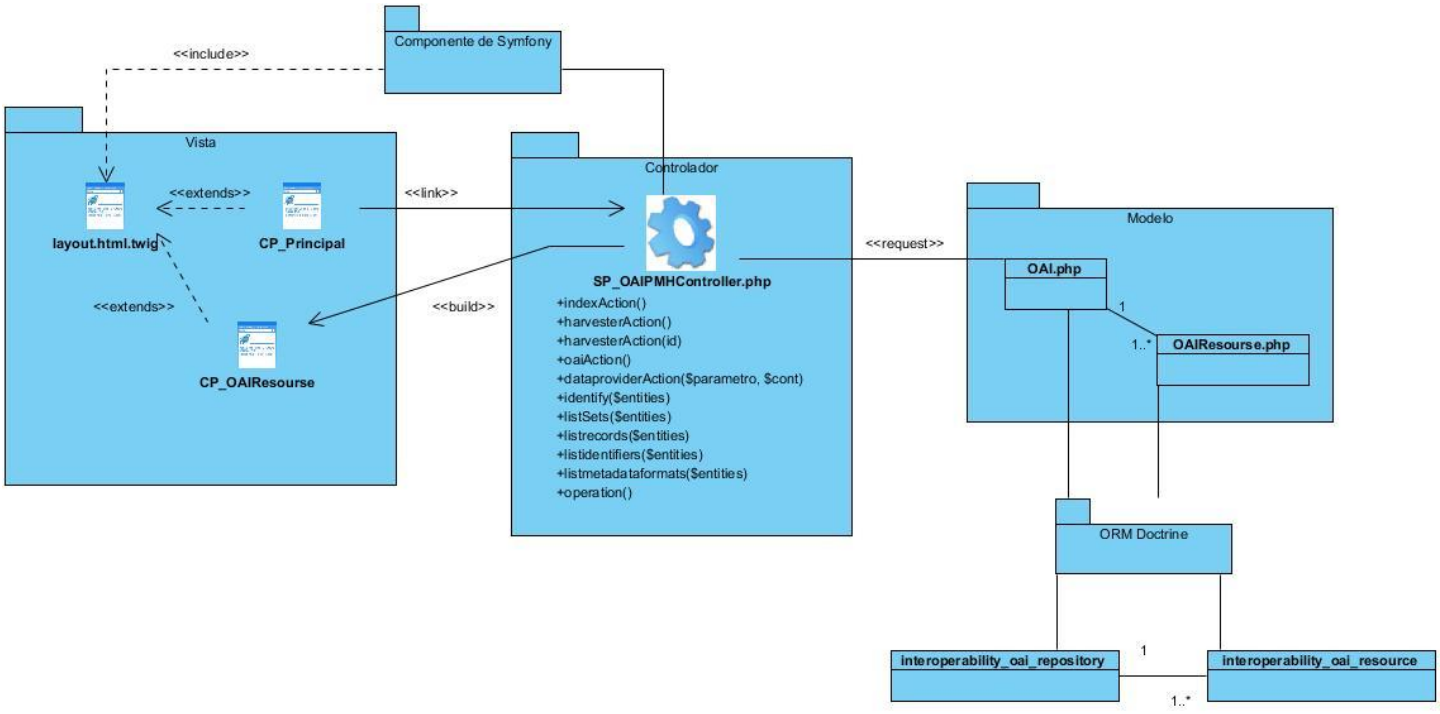


Figura 38. DCD del CU: Ofrecer catálogo.