UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

CENTRO DE INFORMATIZACIÓN UNIVERSITARIA

FACULTAD 1

Módulo para garantizar la prestación de servicios según el estándar OAI-PMH para el Sistema de Gestión de Documentos Históricos.

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

Autora: Yanet Villar Villar.

Tutores: Ing. Reynier Pernía Rodríguez.

Ing. Leodán De los Ángeles Budúen.

Ciudad de La Habana

Junio, 2012

"Si los jóvenes fallan, todo fallará. Es mi más profunda convicción que la juventud cubana luchará por impedirlo. Creo en ustedes."

Fidel Castro

Declaración de autoría

Por este medio declaro que soy el único autor de este trabajo y autorizo al Centro de Informatización Universitaria de la Universidad de las Ciencias Informáticas para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo el presente a los ____ días del mes de _____ del año _____.

_____ _____

Firma del tutor
Ing. Reynier Pernía Rodríguez

Firma del tutor
Ing. Leodán de los Ángeles Budúen

_____

Firma del autor
Yanet Villar Villar

Dedicatoria

Dedico este trabajo a mi papá porque sé que él más que nadie se sentiría súper orgulloso de verme convertida en una profesional, porque a pesar de no estar, pensar en él me da las fuerzas para conseguir las cosas que quiero.

Agradecimientos

A mi papá que mientras pudo dio todo de sí para verme feliz y hacer todos mis sueños realidad.

A mi mamá por comprenderme, por confiar en mí, por darme tanto amor, por ser la mejor madre del mundo. Te quiero mucho.

A mi novio Javier por darme su ayuda y apoyo incondicional en todo momento, por darme fuerzas para seguir adelante sin importar los obstáculos, por su comprensión, su cariño y su amor, por estar junto mí cuando más lo necesito. Gracias mivi.

A mi hermana por estar siempre a mi lado, por apoyarme en todo y brindarme su cariño.

A mi cuñado Jorgito por todo el apoyo y cariño que me ha brindado durante todos estos años.

A mis abuelos, mis tíos y mis primos por apoyarme durante estos 5 años.

A Lilia por ser como una madre para mí, por darme siempre su cariño, por ser tan especial conmigo.

A Cary, Raulito y todo el familión por ayudarme tanto y ser parte de mi familia.

A Mayda, Raulito, Nito, Miriam y toda la gente de Ciego por apoyarme y ofrecerme un cariño sincero.

A mis amigos del inseparable grupo 4 por todos los momentos lindos que hemos vivido juntos, por estar a mi lado en momentos difíciles, por ser como mi familia durante estos 5 años.

A Ilda, Liuba, Yanet, Lily, Lisbet, Alibech, Inalbis, Maire y Leanet por haberme ofrecido su amistad sincera.

A Mary, Orlando, Alberto, Dayanys y todos los compañeros de trabajo de mi mamá por tenerme presente.

A mis vecinos por estar siempre pendientes de mí.

A mis tutores por su tiempo y dedicación, por su exigencia.

A los profes Yoani y Laritza por sus oportunas sugerencias.

A todas mis amistades, especialmente a Yailín, Yarita, Yuliani, Dairilys, Dairene y Lisbet por apoyarme siempre y ser excelentes amigas, a ellas y a su familia, gracias.

A mis maestras Mayda, Dania, Leanne, Evelyn y Melba por enseñarme que siempre se puede más.

A todas las personas que me aprecian y se preocupan por mí, gracias.

Resumen

El presente trabajo describe las características referentes a una solución que permite al Sistema de Gestión de Documentos Históricos ArchiVenHIS poner a disposición de otros sistemas el patrimonio documental en él descrito. Se analizan algunas de las iniciativas surgidas con el propósito de permitir la interoperabilidad entre los sistemas de información y se selecciona una de ellas para su posterior implementación en ArchiVenHIS. Se realiza un estudio del Protocolo para la Recolección de Metadatos de la Iniciativa de Archivos Abiertos OAI-PMH, señalando sus principales características y deficiencias. Se describen los formatos de metadatos Dublin Core y Descripción Archivística Codificada (EAD), empleados por ArchiVenHIS para transmitir los metadatos que contiene.

Palabras clave: interoperabilidad, metadatos, Protocolo para la recolección de Metadatos (OAI-PMH), Sistema de Gestión de Documentos Históricos (SGDH).

Índice

Introducción

El hombre, desde que comenzó a desarrollar sus pensamientos, sintió la necesidad de dejar constancia de sus memorias y conservar sus conocimientos para que pudieran ser heredados por sucesivas generaciones. Por tal necesidad fue designado un lugar con la finalidad de que las personas almacenaran y coleccionaran sus creaciones, lugar que se conoce hoy con el nombre de archivo o fondo documental.

Un archivo es una institución o parte estructural de ella que reúne, conserva, ordena y difunde conjuntos orgánicos de documentos para la gestión administrativa, la información, la investigación y la cultura (FUSTER RUIZ, 2006). Los archivos remontan sus orígenes al surgimiento de la escritura. Diferentes culturas como la griega, la romana y la egipcia contaron con importantes archivos que sirvieron como una herramienta para el control de su población y sus riquezas. En la actualidad tienen como finalidad la preservación de documentos que posean valor administrativo, legal, fiscal, científico, económico, político, cultural y/o histórico, documentos que obviamente ostentan una importancia vital a la hora de intentar bucear en la identidad y la reconstrucción histórica de una institución o nación, de modo que se garantice su integridad y transmisión a futuras generaciones. La naturaleza de los archivos, los principios de su organización y conservación, así como los medios para su utilización son aspectos estudiados por la ciencia denominada Archivística (DORADO SANTANA, 2009).

Como consecuencia de la creciente presencia de las Tecnologías de la Información y las Comunicaciones (TICs), se hizo necesaria la creación de nuevas formas que permiten a los usuarios interesados adquirir información referente a documentos importantes que por su antigüedad o nivel de deterioro no pueden ser consultados físicamente. Se trata de los Sistemas de Gestión de Documentos Históricos (SGDH), a través de los cuales no solo se garantiza el buen estado de los documentos sino que se agiliza el proceso de búsqueda de los mismos.

Un elemento esencial en el contexto de los SGDH es el intercambio automático de datos. Los sistemas de este tipo pueden establecer una comunicación con el fin de posibilitar la transferencia e intercambio de información entre ellos. Este proceso se conoce con el nombre de interoperabilidad[1]. Los sistemas de información interoperables deben implementar determinada norma o estándar que permita dicha comunicación de manera rápida y transparente.

[1] Habilidad de dos o más sistemas o componentes para intercambiar información y utilizarla.

La Universidad de las Ciencias Informáticas (UCI) es un centro de estudios universitarios que desde su surgimiento ha tenido dos objetivos fundamentales: informatizar el país y desarrollar la industria del software para contribuir al desarrollo económico del mismo. La UCI no ha quedado exenta de los adelantos científicos-técnicos que ocurren en el mundo hoy en día y en aras de dar cumplimiento a sus objetivos, se ha estructurado en diferentes centros productivos entre los que se destaca el Centro de Informatización Universitaria (CENIA) dentro del cual se encuentra el Departamento de Gestión Documental dividido en diferentes líneas de desarrollo donde se encuentra la de Soluciones para la gestión de archivos.

En el año 2007 se creó el Sistema de Gestión de Documentos Históricos ArchiVenHIS, en el marco del proyecto "Uso y aplicación de las TICs para el mejoramiento de la gobernabilidad y aumento de la soberanía tecnológica (Fase 1)", en el que se proporciona al Archivo General de la Nación (AGN) de la República Bolivariana de Venezuela una solución que contribuye a la preservación y difusión del fondo documental bajo su custodia.

ArchiVenHIS cuenta con variadas funcionalidades (Búsqueda sencilla, Búsqueda avanzada, Exploración) que facilitan al usuario la localización de los documentos con base en los metadatos descritos a través del sistema siguiendo la Norma Internacional General de Descripción Archivística ISAD(G). Sin embargo no es capaz de proporcionar a otros sistemas acceso a la información sobre el material custodiado en las instituciones de archivo donde se encuentra instalado. La capacidad de proveer tal servicio constituye un importante valor agregado para los sistemas de gestión de documentos históricos. En la actualidad muchos de ellos brindan soporte para estas funcionalidades contribuyendo a una mejor difusión de la memoria histórica salvaguardada por las instituciones donde se utilizan.

Por lo anteriormente expuesto se plantea como **problema a resolver**: ¿cómo garantizar que el Sistema de Gestión de Documentos Históricos ArchiVenHIS permita la consulta a través de terceros del patrimonio documental resguardado y descrito en las instituciones que lo emplean?

Definiéndose como **objeto de estudio** de esta investigación los protocolos que permiten la comunicación entre aplicaciones web y como **campo de acción** los protocolos que se emplean para la comunicación entre aplicaciones que gestionan documentos históricos.

De acuerdo con la problemática planteada se propone como **objetivo general:** desarrollar un módulo para el Sistema de Gestión de Documentos Históricos ArchiVenHIS que le permita poner a disposición de otros sistemas el patrimonio documental en él descrito.

El presente trabajo queda sustentado por la **idea a defender:** el desarrollo de un módulo capaz de garantizar que ArchiVenHIS permita a otros sistemas la consulta del patrimonio documental resguardado y descrito en él, incrementa las posibilidades de difusión del acervo histórico bajo la custodia de cada una de las instituciones donde se emplea.

Con el propósito de cumplir con el objetivo trazado, se plantean las siguientes **tareas de la investigación:**

- Evaluación de las tendencias actuales de los sistemas de gestión de documentos históricos.
- Estudio de normas y estándares que permitan la interoperabilidad entre sistemas de información.
- Levantamiento de requisitos funcionales y no funcionales para el módulo a desarrollar.
- Diseño de las interfaces de usuario necesarias.
- Diseño de la base de datos.
- Implementación de los elementos de diseño.
- Implementación de pruebas de desarrollador.
- Definición de las configuraciones del entorno de prueba.
- Ejecución de pruebas de desarrollador.
- Integración del sistema.

Para el desarrollo de las tareas de la investigación se hace necesaria la utilización de algunos métodos teóricos como el análisis histórico-lógico que permitirá estudiar el modo en que han evolucionado los estándares y normas existentes para facilitar el intercambio de datos entre sistemas de información. También será de gran utilidad el método analítico-sintético, el cual posibilitará descomponer y distinguir los principales elementos que intervienen en la comunicación que pueda establecerse entre dos sistemas, así como del protocolo que se seleccione para el desarrollo del módulo a construir, con el objetivo de estudiar cada uno de estos elementos por separado y posteriormente integrarlos, de modo que se logre un mayor entendimiento y se pueda arribar a conclusiones que contribuyan a la solución del problema que dio origen a la investigación. El método empírico que se empleará es la observación,

el cual guiará el estudio del estado del arte, permitiendo realizar un análisis sistémico, selectivo y objetivo de los principales sistemas que en la actualidad pueden realizar intercambios de información entre ellos.

Con la implementación de un módulo para garantizar la interoperabilidad entre ArchiVenHIS y otros sistemas se pretende lograr una mayor difusión del patrimonio documental resguardado por las instituciones que emplean dicho sistema.

El presente documento está compuesto por una introducción, tres capítulos de contenido, conclusiones generales, recomendaciones, referencias bibliográficas, bibliografía y anexos. Los capítulos están estructurados de la siguiente manera:

Capítulo 1. **Fundamentos teóricos de la archivística y los protocolos para la interoperabilidad**: se expone un conjunto de conceptos y criterios fundamentales para lograr un mayor entendimiento de lo que es la archivística. Se plasma el resultado de un estudio realizado sobre los estándares que permiten la interoperabilidad entre sistemas de información, específicamente entre sistemas de gestión de documentos históricos. Se analizan los principales sistemas de gestión de documentos históricos que realizan transacciones de información entre ellos. Además, se exponen las distintas tecnologías a utilizar en la implementación del módulo, así como la metodología de desarrollo de software a emplear.

Capítulo 2. Características del módulo para garantizar la prestación de servicios según OAI-PMH para ArchiVenHIS: en este capítulo se realiza una descripción detallada de la solución que se propone para erradicar el problema que dio origen a la presente investigación. Se representan los principales conceptos que se manejan en el contexto del sistema mediante un modelo de dominio. Se plasma un levantamiento de todas las necesidades que debe satisfacer la solución, especificando para ello los requisitos funcionales y no funcionales. A partir de los requisitos obtenidos, se definen los casos de uso y los actores que se relacionan con cada uno de ellos, confeccionando el Diagrama de Casos de Uso. Se realiza el análisis y diseño del sistema, la representación del modelo de datos y se definen los patrones de arquitectura y diseño que serán utilizados.

Capítulo 3. **Implementación y pruebas del módulo para garantizar la prestación de servicios según el estándar OAI-PMH para el SGDH ArchiVenHIS**: se representan los elementos físicos necesarios para un correcto despliegue de la aplicación, empleando para ello un diagrama de despliegue. Se muestran componentes de la implementación. Se efectúa la validación y prueba de la

solución de acuerdo a los requisitos que debe cumplir para garantizar una calidad óptima, todo esto a través de pruebas funcionales.

Capítulo I. Fundamentos teóricos de la archivística y los protocolos para la interoperabilidad.

En el presente capítulo se hace referencia a los principales aspectos teóricos que constituyen la base de la investigación. Se presentan los resultados del estudio del estado del arte, exponiendo la actualidad y antecedentes del tema que se trata en la investigación. Son abordados algunos conceptos fundamentales dentro del ámbito de la gestión documental y la archivística. Se presentan los resultados de un estudio realizado sobre los protocolos que permiten la comunicación entre sistemas de información, especialmente entre SGDH, así como los principales sistemas que implementan dichos protocolos. Quedan descritos algunos elementos como la metodología de desarrollo del software y tecnologías a utilizar en el desarrollo de la aplicación.

1.1. Conceptualización

Aunque no ocurre así con los objetos de los que se ocupa, la ***archivística*** puede considerarse una ciencia relativamente nueva. Algunos autores como Cruz Mundet y Antonia Heredia consideran que el surgimiento de este término tuvo lugar a mediados del siglo XIX. Desde entonces, muchos han sido los teóricos que han tratado de ofrecer una definición de esta ciencia.

La archivera española Antonia Heredia Herrera, una de las teóricas que más ha influido en los últimos tiempos en la capacitación de los archiveros cubanos, la define como “la ciencia de los archivos, no de los documentos, aunque en última instancia estos sean el producto integrante de aquellos. Como tal se ocupará de la creación, historia, organización y el servicio de los mismos a la Administración y a la Historia, en definitiva a la Sociedad” (HEREDIA HERRERA, 1991).

Según el Diccionario de Terminología Archivística es “la disciplina que trata de los aspectos teóricos y prácticos (tipología, organización, funcionamiento, planificación, etcétera) de los archivos y el tratamiento archivístico de sus fondos documentales” (MINISTERIO DE CULTURA, 1995).

Por otra parte, Cruz Mundet considera que “es la ciencia de los archivos, y que como tal ciencia está integrada por un conjunto de conocimientos y de métodos para el tratamiento de los documentos y de los archivos” (CRUZ MUNDET, 2011).

En resumen, la archivística puede ser definida como *la ciencia que estudia la naturaleza de los archivos, los principios de su organización y conservación, así como los medios para su utilización.*

Resulta complicado comprender cualquiera de las definiciones existentes de archivística si no se estudia previamente el significado de su objeto de estudio, los archivos.

Antonia Heredia define un ***archivo*** como “uno o más conjuntos de documentos, sea cual sea su fecha, su forma y soporte material, acumulados en un proceso natural por una persona o institución pública o privada en el transcurso de su gestión, conservados, respetando aquel orden, para servir como testimonio e información para la persona o institución que los produce, para los ciudadanos o para servir de fuentes de historia” (HEREDIA HERRERA, 1991).

Por otra parte, Mayra Mena Mugica considera que “el archivo no es más que el reflejo natural y la plasmación en sus documentos de las actividades y tareas de una entidad determinada” (MENA MUGICA, 2005).

Una definición más actual es ofrecida por Cruz Mundet, planteando que “un archivo es entendido como un sistema corporativo de gestión que contribuye de manera efectiva, mediante una metodología propia, a la definición de los procesos de producción administrativa garantizando la correcta creación de los documentos, su tratamiento, conservación, acceso y comunicación” (CRUZ MUNDET, 2011).

De manera general un archivo puede definirse como *un conjunto de documentos, sin importar su fecha, naturaleza y soporte material, acumulados por una persona o institución en el desarrollo de su actividad, que pueden servir como información o testimonio de sus creadores. Asimismo se entienden también por archivos las instituciones culturales donde se reúnen, conservan, ordenan y difunden conjuntos de documentos utilizables en la investigación, la cultura, la información y la gestión administrativa.*

Sin dudas, un elemento fundamental para los archivos son los ***documentos de archivo***. De acuerdo con los expresado por Mayra Mena Mugica, un documento de archivo “es el testimonio material de un hecho o acto realizado en el ejercicio de sus funciones por personas físicas o jurídicas, públicas o privadas, de acuerdo con las características de tipo material y formal” (MENA MUGICA, 2005).

La definición incorporada en la Ley de Archivos de Andalucía de 1984 plantea que “un documento es toda expresión en lenguaje oral o escrito, natural o codificado, recogida en cualquier tipo de soporte material, así como cualquier otra expresión gráfica que constituya testimonio de funciones y actividades sociales del hombre y de los grupos humanos, con exclusión de las obras de creación y de investigación editadas, y de las que por su índole, forman parte del patrimonio bibliográfico, así como las expresiones aisladas de naturaleza arqueológica o artística” (CHAVEZ GONZÁLEZ, 1999).

En resumen, *son documentos producidos o recibidos por una persona o institución durante el curso de su gestión o actividad para el cumplimiento de sus objetivos y son conservados para que puedan ser utilizados como prueba o fuente de información.*

Existe un conjunto de elementos que diferencian a los documentos de archivo de un documento común (CRUZ MUNDET, 2011):

- El carácter seriado: los documentos se producen uno a uno y con el paso del tiempo constituyen series.
- La génesis: se producen dentro de un proceso natural de actividad, surgen como producto y reflejo de las tareas de su productor, no son algo ajeno a él.
- La exclusividad: la información que contiene rara vez se encuentra en otro documento con idéntica extensión e intensidad, es exclusiva.
- La interrelación: como principio general las piezas aisladas (documentos sueltos) no tienen sentido o tienen muy poco, su razón de ser viene dada por su pertenencia a un conjunto (la unidad archivística o expediente) y por las relaciones establecidas entre sí.

Para posibilitar el acceso a la documentación almacenada en un archivo es imprescindible la existencia de un enlace entre la información contenida en los documentos y las personas que desean consultarla. Para ello, el archivero debe realizar un análisis de todos los documentos y sintetizar la información que poseen, generando descripciones (representaciones de las unidades documentales) para ofrecerlas a los interesados (GÓMEZ DÍAZ y BRINGAS GONZÁLEZ, 2005).

"La ***descripción archivística*** es el medio utilizado por el archivero para obtener información contenida en los documentos y ofrecerla a los interesados en ella" (HEREDIA HERRERA, 1991). Se realiza con dos objetivos fundamentales: dar información a los usuarios y facilitar el trabajo al archivero. La descripción de los documentos debe ser concreta, breve y contener las ideas básicas; tanto los factores internos como los externos deben ser descritos.

Las descripciones son realizadas por los archivistas a través de instrumentos descriptivos como guías, inventarios, índices y catálogos. Un instrumento descriptivo que ha ganado gran aceptación en el campo archivístico es la ***Norma Internacional General de Descripción Archivística ISAD(G)***.

Esta norma constituye una guía general para la elaboración de descripciones archivísticas. Está compuesta por un conjunto de reglas generales encaminadas a asegurar la creación de descripciones coherentes, pertinentes y explícitas, facilitar la recuperación y el intercambio de información sobre los documentos de archivo, compartir los datos de autoridad y hacer posible la integración de descriptores de diferentes archivos en un sistema unificado de información.

Las reglas de la ISAD(G) se encuentran estructuradas en siete áreas de información (THE SOCIETY OF AMERICAN ARCHIVISTS, 2002):

- Área de Identificación.
- Área de Contexto.
- Área de Contenido y Estructura.
- Área de Condiciones de Acceso y Uso.
- Área de Documentación Asociada.
- Área de Notas.
- Área de Control de la Descripción.

Además, establecen veintiséis elementos que conforman la descripción de cualquier unidad, de estos, solo seis son esenciales en cualquier tipo de descripción:

- Código de Referencia.
- Título.
- Productor.
- Fechas.
- Volumen.
- Nivel de Descripción.

La ISAD(G) establece un conjunto de niveles de organización que van de lo general a lo específico, entre los que se encuentran el nivel de Fondo, Subfondo, Serie, Subserie, Unidad documental compuesta y Unidad documental simple.

Con la unión de los elementos anteriormente mencionados surgen las aplicaciones para la gestión de archivos históricos a través de los cuales se proporciona a los usuarios instrumentos para la solicitud y consulta de representaciones digitales de documentos de su interés. Estos sistemas contribuyen considerablemente a la organización, conservación y preservación de los documentos de archivo, además de facilitar a gran escala el trabajo del archivero.

Cualquier desarrollo informático que involucre un procesamiento de datos e información, y que apoye a las organizaciones en la optimización, mejoramiento y desarrollo de sus procesos puede ser considerado un sistema de información (GÓMEZ LAUREANO, 2007). Existen variantes de estos sistemas de acuerdo al tipo de proceso que se desea trabajar, por lo que se puede considerar que los sistemas informáticos que se encargan de gestión de documentos históricos son un caso específico de los sistemas de información.

Los sistemas de información pueden interoperar con otros sistemas de su tipo, es decir, pueden comunicarse y compartir datos, información, documentos y objetos digitales de forma efectiva (con una mínima o nula pérdida de su valor y funcionalidad), con uno o varios sistemas de información (siendo generalmente estos sistemas completamente heterogéneos, distribuidos y geográficamente distantes), mediante una interconexión libre, automática y transparente, sin dejar de utilizar en ningún momento la interfaz del sistema propio (GÓMEZ LAUREANO, 2007).

A partir de la evolución de las TICs y del auge de los sistemas de información, surge la necesidad de establecer entre los sistemas de gestión de documentos históricos la interoperabilidad, con el fin de posibilitar la transferencia y recuperación de información entre ellos. La capacidad de interoperar que pueden poseer estos sistemas está fundamentada en el uso de protocolos que facilitan el intercambio de información de una manera rápida y organizada.

1.2. Protocolos para la transferencia y recuperación de información

Un protocolo puede ser definido como un “conjunto de reglas o estándares diseñados para permitir a las computadoras conectarse con otras e intercambiar información con el mínimo de error posible” (MICROSOFT CORPORATION, 2000).

En el Glosario de Términos Informáticos se define un protocolo como la “descripción formal de mensajes y de reglas que los ordenadores deben seguir para intercambiar dichos mensajes”. Se plantea además que “es un estándar que escribe las funciones de control, configuración y metodología utilizada en, por ejemplo, los sistemas de comunicaciones. Un protocolo asegura la compatibilidad” (GLOSARIOIT, 2012).

Luego del análisis de los conceptos expuestos se puede inferir que, en el ámbito de las ciencias de la información, un protocolo no es más que un *conjunto de reglas escritas con el objetivo de permitir el intercambio de información entre equipos o sistemas informáticos con la menor cantidad de errores posible.*

Muchas han sido las propuestas de protocolos para la interoperabilidad que se han desarrollado, entre ellas se pueden citar las siguientes:

- ***Protocolo Simple para la Interoperabilidad de Bibliotecas Digitales***

El protocolo simple para la interoperabilidad de bibliotecas digitales (SDLIP) es utilizado para la integración de múltiples fuentes de información heterogéneas. Fue desarrollado gracias a la unión de las universidades de California en Berkeley, Stanford y Santa Bárbara, el centro de supercomputadoras de San Diego y el proyecto de Biblioteca Digital de California (STANFORD DIGITAL LIBRARY TECHNOLOGIES, 2004).

A través de un protocolo de transporte y una interfaz de búsqueda SDLIP, un cliente solicita a un servidor realizar una búsqueda sobre determinado recurso de información. Para dar respuesta el servidor accede a los metadatos almacenados en los diferentes repositorios de información los cuales no necesariamente poseen una implementación del SDLIP. Este protocolo ofrece una solución que no se adapta del todo a las necesidades del sistema a implementar. A través de él, un usuario puede realizar búsquedas en diferentes fuentes, sin embargo lo que se quiere lograr es que, mediante otros sistemas, los usuarios tengan acceso a la información salvaguardada por las instituciones de archivo que empleen ArchiVenHIS, además es mayormente usado para la interoperabilidad entre bibliotecas digitales.

- ***Protocolo Guildford***

Proporciona un conjunto de reglas para la publicación e intercambio de documentos y metadatos en la Red y puede ser implementado tanto individualmente como en grupos. Establece dos niveles: el de archivo (de carácter pasivo pues simplemente proporciona información) y el de servicio (activo), ya que extrae la información de los anteriores y construye un "servicio" de utilidad para los usuarios finales (BARRUECO y KRICHEL, 1999).

Su surgimiento se debe, fundamentalmente, a una declaración realizada por William L. Goffe, el 15 de julio de 1995 donde expresó que lo que se necesitaba era un sistema distribuido con cualquier número de sitios que fueran reflejo uno del otro. Su principal objetivo es facilitar el acceso a los resultados de investigaciones en las diferentes áreas de la economía entre comunidades *RePEc*[2], sin embargo

[2] Conjunto de herramientas conceptuales, protocolos, normas y software cuyo objetivo es la distribución electrónica y descripción bibliográfica de los documentos científicos producidos por una disciplina académica, en concreto la economía.

puede ser adaptado a otros campos de la sociedad pero esto requeriría un esfuerzo adicional por lo que resulta más conveniente no utilizarlo en la solución al problema que dio origen a la investigación y buscar una solución más factible.

- ***Protocolo Dienst***

El protocolo Dienst permite la unión de un conjunto de servicios para formar una biblioteca digital distribuida (DAVIS *et al.*, 2000), es decir, los servicios y recursos de una biblioteca digital Dienst pueden estar físicamente ubicados en cualquier lugar. Es utilizado por la Networked Computer Science Technical Reference Library[3] (NCSTRL) para poner sus documentos a disposición de los usuarios. Con el paso de los años este protocolo ha sido reemplazado por otras alternativas más sencillas y genéricas, fundamentalmente por OAI-PMH por lo que en la actualidad es muy poco usado, cuestión que demuestra que su uso para dar solución al problema planteado no es lo más adecuado pues no se garantizaría una exitosa difusión de la memoria histórica de las instituciones que emplean ArchiVenHIS.

- ***Protocolo Z39.50***

El Z39.50 es un protocolo para la recuperación de información basado en la estructura cliente/servidor que facilita la interconexión de sistemas informáticos. Su principal objetivo consiste en permitir al usuario realizar búsquedas en bases de datos que se encuentren en un servidor Z39.50, sin la necesidad de conocer la sintaxis de búsqueda que utilicen los sistemas. Z39.50 especifica un conjunto de reglas para gestionar las formas y procedimientos de interconexión remota de computadoras, con el propósito de buscar y recuperar información, aunque su aplicación actual es más amplia pues incluye la consulta y el intercambio de datos bibliográficos, y la intercomunicación de índices y resúmenes, de información geoespacial, de documentos oficiales, de objetos digitales o de metadatos que describen los documentos de las bibliotecas electrónicas o digitales (GUAJARDO SALINAS, 2010).

Su funcionalidad es bastante completa pues también se preocupa por cuestiones como el manejo de sesiones, gestionar conjuntos de resultados, relacionar bases de datos diferentes y permitir la especificación de predicados para filtrar los resultados obtenidos.

A pesar de las ventajas que implica, el desarrollo de estas funcionalidades trae consigo un considerable incremento en la complejidad y los costos de la implementación, además el protocolo

[3] Biblioteca digital compuesta por informes a texto completo sobre informática.

Z39.50 facilita considerablemente el trabajo a los bibliotecarios por lo que es utilizado mayoritariamente en sistemas de gestión bibliotecaria.

- ***Protocolo para la recolección de metadatos***

El Protocolo para la Recolección de Metadatos de la Iniciativa de Archivos Abiertos (*Open Archives Initiative Protocol for Metadata Harvesting, OAI-PMH*) fue creado gracias al surgimiento de la Iniciativa de Archivos Abiertos (*Open Archives Initiative, OAI)*, la cual se encarga de promover estándares de interoperabilidad para aumentar y facilitar el acceso y transmisión de información.

OAI-PMH proporciona una plataforma sencilla y basada en tecnología existente para acceder y/o difundir cualquier tipo de información, lo cual hace que este protocolo sea preferido por las instituciones que desean transmitir su información a otros sistemas y se haya convertido en el más usado en la actualidad. Establece las reglas necesarias para permitir el intercambio de información de una manera organizada. Está compuesto por dos partes fundamentales, los proveedores de servicios y los proveedores de datos, estos últimos permiten que los primeros realicen recolecciones de metadatos en ellos a través del envío de peticiones. Mediante OAI-PMH solo pueden ser enviados los metadatos que describen un determinado recurso y no el recurso en sí, permitiendo que la transmisión se realice de una manera rápida.

Por las razones anteriormente expuestas se decide emplear OAI-PMH para poner a disposición de otros sistemas de información el patrimonio documental descrito en ArchiVenHIS. Con su uso, no se trata de reemplazar alternativas, sino de buscar la más acertada en el contexto en que se desarrolla el presente trabajo de diploma.

Las bases de este protocolo están fundamentadas en el resultado de un encuentro entre expertos realizado en Santa Fe, Estados Unidos. El encuentro se realizó en octubre del año 1999 y tuvo como objetivos fundamentales debatir las cuestiones relativas a la interoperabilidad, acordar el inicio de los trabajos en un prototipo promocional de un servicio de biblioteca digital basándose en los principales repositorios de *e-prints*[4] existentes, y establecer un foro para la continuación de los trabajos sobre la interoperabilidad de soluciones de autoarchivo (CARPENTER, 2003). Las personas allí reunidas coincidían en la premisa de que la idea de la interoperabilidad de los archivos *e-prints* era clave para aumentar su impacto entre la comunidad académica ya que se podrían federar varios archivos,

[4] Documentos autoarchivados por su autor. Habitualmente se emplea este término en el sentido que el contenido del *e-print* es el resultado de la investigación científica o académica.

intercambiar registros y realizar búsquedas en disciplinas relacionadas al mismo tiempo (BARRUECO y SUBIRATS COLL, 2003).

Seleccionar el camino a seguir fue una decisión clave en este encuentro. Dos de las propuestas de solución fueron la búsqueda simultánea de múltiples archivos a partir de un protocolo como el Z39.50, o la recolección de metadatos en uno o más servicios centrales, moviendo los datos y poniéndolos más cerca de la interfaz de usuario (CARPENTER, 2003). Luego de largas horas de debate entre los participantes, se estableció, como resultado del encuentro, un conjunto de acuerdos técnicos y organizativos conocidos como La Convención de Santa Fe, que proponían la recolección de metadatos como solución para garantizar de interoperabilidad entre los distintos proveedores de *e-prints*. Los aspectos técnicos de la Convención incluían tres puntos fundamentales: un formato para los metadatos, un protocolo de intercambio y un sistema de identificación.

Poco tiempo después, se decidió ampliar el objeto de trabajo de la Convención, ya no se encargaría solamente de la divulgación de *e-prints*, sino que se incluían otras disciplinas que no poseían ese tipo de documentación. Es en este momento que surge OAI-PMH 1.0, cuyo objetivo era facilitar la búsqueda de todos aquellos objetos tipo documento. Con una adopción lenta y progresiva, se crea, en el año 2002 la versión 2. El OAI-PMH 2.0 es una revisión profunda del protocolo, una vez más el objeto se amplía y ahora se trata del intercambio recurrente de metadatos de recursos entre distintos sistemas (GUAJARDO SALINAS, 2010).

Flujo actual de los procesos del OAI-PMH

OAI-PMH provee un marco de interoperabilidad independiente de la aplicación basado en la recolección de metadatos y define reglas que permiten acceder fácilmente a una colección digital y recuperar la información (metadatos) de interés. Utiliza transacciones HTTP (Protocolo de Transferencia de Hipertexto) para emitir preguntas y obtener respuestas entre un proveedor de servicios y un proveedor de datos. El primero puede pedir al segundo que le envíe metadatos según determinados criterios, por ejemplo la fecha de creación de los datos. En respuesta el proveedor de datos le devuelve un conjunto de registros codificados en sintaxis XML, que incluyen identificadores de los objetos descritos en cada registro. Los mensajes de error que se pueden transmitir se basan en HTTP. En el Anexo1 se puede apreciar el funcionamiento básico de OAI-PMH 2.0.

Las peticiones del proveedor de servicios se realizan utilizando los métodos GET o POST del protocolo HTTP y están compuestas por un grupo de opciones con la forma de pares del tipo “clave = valor”.

Para realizar dichas peticiones OAI-PMH consta de seis términos denominados “verbos” (OPEN ARCHIVES INITIATIVE, 2008):

- **GetRecord:** utilizado para recuperar un registro concreto. Necesita dos argumentos: identificador del registro pedido y especificación del formato de metadatos en que se debe devolver.
- **Identify:** utilizado para recuperar información sobre el servidor: nombre, versión del protocolo que utiliza, dirección del administrador, entre otras.
- **ListIdentifiers:** recupera los encabezados de los registros, en lugar de los registros completos.
- **ListRecords:** recupera los registros completos de un repositorio.
- **ListSets:** recupera todos los conjuntos de registros que posee el repositorio. Estos conjuntos son creados opcionalmente por el servidor para facilitar una recuperación selectiva de los registros.
- **ListMetadataFormats:** devuelve la lista de formatos de metadatos que utiliza el servidor.

Además del verbo, la petición puede estar compuesta por algunos argumentos que indican determinadas características que debe tener la respuesta, estos son (OPEN ARCHIVES INITIATIVE, 2008):

- **MetadataPrefix:** con este atributo se especifica el formato de metadatos con que se requiere la respuesta. Es obligatorio para ListIdentifiers, ListRecords y GetRecord.
- **Set:** indica el conjunto del cual deben ser devueltos los metadatos. Se utiliza, solo si el repositorio posee una estructura de conjuntos, para los verbos ListIdentifiers y ListRecords, en ambos casos es opcional.
- **From:** es una fecha que indica que se solicitan solo los registros que han sido modificados o adicionados a partir de ella. Puede ser empleado de conjunto con los verbos ListIdentifiers y ListRecords.
- **Until:** es una fecha que indica que se solicitan solo los registros que han sido modificados o adicionados hasta ella. Puede ser empleado de conjunto con los verbos ListIdentifiers y ListRecords.
- **ResumptionToken:** se utiliza cuando se generan respuestas extensas y es necesario dividirlas en varias partes, indica las características de la búsqueda que se debe realizar y a partir de qué

registro debe comenzar la respuesta. Puede ser empleado de conjunto con los verbos ListSets, ListIdentifiers y ListRecords.

Existen aspectos que no son tratados por el protocolo, siendo las cuestiones de gestión o autorización para el acceso de los clientes un ejemplo de ello. El servidor deberá recurrir a métodos externos si desea limitar a los clientes en el sentido de la utilización que se le puede dar a los datos. Tampoco trata el tema de cómo los clientes pueden localizar aquellos servidores que contengan los datos que necesitan.

Otro aspecto importante que no es tomado en cuenta por el protocolo es la estandarización de la granularidad de las fechas, cada repositorio es libre de emplear el formato deseado, cuestión que provoca problemas de incompatibilidad a la hora de realizar recolecciones por parte de los proveedores de servicios. Además, el protocolo establece que el control del flujo de la transmisión de los metadatos es opcional, sin embargo este aspecto debería considerarse obligatorio para proveedores de datos que almacenan grandes volúmenes de información, de modo que se garantice el éxito de la transmisión.

En este sentido, el protocolo no especifica completamente como manejar la señal de reanudación, el tiempo de expiración de dicha señal varía en cada repositorio, por lo que se da el caso de que proveedores de servicios intentan continuar con una recolección que por algún motivo fue interrumpida durante un determinado tiempo y el proveedor de datos no es capaz de responder correctamente, por lo que se hace necesario comenzar la recolección desde el principio.

Proveedores de datos

En la sección anterior se hizo referencia a los términos proveedores de datos y proveedores de servicios, son estos los participantes en este marco de la Iniciativa de los Archivos Abiertos. Los primeros son los archivos que poseen la información y los segundos son los recolectores que toman los datos con el objetivo de incorporarles algún valor añadido para presentarlos a los usuarios finales. La presente investigación está enfocada mayormente a los proveedores de datos debido a que es esta parte del protocolo OAI-PMH la que permite dar solución al problema existente.

Los proveedores de datos manejan el depósito y la publicación de los recursos en un repositorio y exponen los metadatos de los recursos del repositorio para que puedan ser recolectados. Ellos son los creadores y conservadores de los metadatos y de los repositorios de recursos (CARPENTER, 2003). Son los encargados de recibir peticiones de metadatos provenientes de proveedores de servicios,

validarlas y decidir si son correctas o no. Al recibir una petición, si no es correcta, el proveedor de datos debe ser capaz de responder con un error y en caso contrario debe realizar los procesamientos necesarios para ofrecer una respuesta apropiada a quien realizó la petición.

Un proveedor de datos está compuesto por un intérprete para la validación de las peticiones, un generador de errores, un interfaz que permita el acceso a los datos y un generador de XML para la creación de las respuestas. Además, es recomendado controlar el flujo de la transmisión de los metadatos debido a que en ocasiones se generan respuestas muy extensas y su transmisión podría dificultarse. En el Anexo 2 se puede apreciar un ejemplo de arquitectura para un proveedor de datos, en el cual se evidencia la interacción de sus componentes.

Existe una gran cantidad de instituciones que han aprovechado las ventajas que proporciona la creación archivos abiertos para lograr una mayor difusión de su información. El número de repositorios que se encuentran registrados en el sitio web de la OAI asciende a más de 1500 (OPEN ARCHIVES INITIATIVE, 2008).

OAI-PMH soporta cualquier formato de metadatos codificado en XML, siendo Dublin Core no cualificado el mínimo que se requiere para una interoperabilidad básica. Además de este, cada servidor es libre de ofrecer los registros en otros formatos adicionales. Un cliente puede pedir que los registros se le sirvan en cualquiera de los formatos soportados por el servidor.

1.3. Estándares de metadatos

Las TICs exigen la normalización de los diferentes elementos informativos que hacen posible el acceso e intercambio de información. En aras de suplir esta exigencia surge, en el campo de la ciencia de la computación, el término metadatos.

Su definición más difundida es que son “datos sobre datos” o “informaciones sobre informaciones”. En el ámbito de las ciencias de la información, los metadatos pueden ser identificados como “Información estructurada sobre un recurso de información soportado en cualquier medio o formato. La información estructurada puede o no ser electrónica y los recursos descritos pueden ser publicaciones impresas, electrónicas, objetos digitales, etcétera” (COMITÉ DE METADATOS DE LA BIBLIOTECA NACIONAL DE CHILE, 2006).

Los metadatos permiten tener una descripción estandarizada de los diferentes conjuntos de datos que están presentes en un sistema. Son una herramienta de gran importancia en la gestión de datos e

información porque facilitan la búsqueda, recuperación e integración de datos provenientes de distintas fuentes.

Se han establecido múltiples y diversas clasificaciones de tipos de metadatos atendiendo a distintos aspectos como su forma, funcionalidad, nivel de estructuración de los datos, persona o entidad que los origina, etcétera. En dependencia de esta clasificación han surgido numerosas iniciativas que permiten estandarizar las descripciones que se realizan a los recursos electrónicos mediante metadatos. Para el desarrollo del trabajo de diploma se utilizarán Dublin Core por ser el formato requerido por el protocolo y EAD por ser el formato que se emplea para la transmisión de descripciones archivísticas.

La ***Descripción Archivística Codificada*** (*Encoded Archival Description, EAD*) constituye la primera norma de estructura de datos elaborada para facilitar la distribución por internet de información detallada sobre colecciones y fondos archivísticos.

Su desarrollo comenzó con un proyecto de la Biblioteca de la Universidad de California en Berkeley, en 1993, sobre la conveniencia de desarrollar un estándar de codificación no propietario para instrumentos de descripción legibles por máquina como inventarios, registros, índices y otros documentos creados por archivos, bibliotecas, museos y repositorios de manuscritos para apoyar el uso de sus fondos. El EAD es mantenido por la Library of Congress[5] y está constituido por un conjunto de reglas semánticas y una sintaxis de codificación.

De forma general, la EAD es una estructura de datos normalizada que reproduce en formato digital los instrumentos de descripción archivística (PEIS y RUIZ RODRÍGUEZ, 2004). Determina los tipos de elementos utilizables, los atributos que estos pueden tener asociados y especifica el contenido que estos tipos de elementos pueden incluir. En pocos años se ha convertido en la lengua franca para el intercambio de descripciones archivísticas en la Web, asociando metadatos a imágenes digitalizadas de materiales archivísticos.

EAD supone una Definición de Tipo de Documento (DTD) elaborada según las reglas sintácticas del Estándar de Lenguaje de Marcado Generalizado[6] (SGML) y del Lenguaje de Marcas Extensible (XML) para codificar instrumentos de descripción (MENA MUGICA, 2005).

[5] Biblioteca del Congreso de Estado Unidos. Construida en el año 1800. Es una de las mayores bibliotecas del mundo. Actualmente es la Biblioteca Nacional de los Estados Unidos.

[6] Sistema para la organización y etiquetado de documentos.

Un documento codificado utilizando EAD, consta de tres segmentos: uno que proporciona información sobre el instrumento de descripción en sí mismo, <eadheader>; un segundo componente que incluye las cuestiones preliminares necesarias para la publicación formal del instrumento de descripción, <frontmatter>; y un tercero que proporciona la descripción del material archivístico en sí misma, además de la información contextual y administrativa asociada, <findaid> (PEIS y RUIZ RODRÍGUEZ, 2004).

La EAD es totalmente compatible con todos los principios de descripción de ISAD(G), razón por la cual es empleada para el intercambio de información archivística a nivel internacional. Las descripciones de los documentos que se archivan en ArchiVenHIS son realizadas según la norma de descripción archivística ISAD(G) por lo que se hace necesario realizar una transformación de los elementos de ISAD(G) a EAD para poder proveer los metadatos en este formato. En el Anexo 3 se puede consultar dicha transformación.

El otro formato de metadatos a emplear es ***Dublin Core*** *(DC)*. Este es un estándar mucho más general ya que define un conjunto básico de atributos que pueden ser utilizados para describir los recursos existentes en cualquier tipo de red. En su primera fase fue promovido por la OCLC[7] (*Online Computer Library Center*) y el NCSA[8] (*National Center for Supercomputing Applications*). Actualmente es la iniciativa de catalogación más extendida en el mundo electrónico, al tiempo que es considerada un estándar internacional (ISO-15836-2003). Describe la forma, el contenido y la localización de la información empleando para ello, una semántica perfectamente entendible por cualquier persona (DUBLIN CORE METADATA INITIATIVE, 2012).

La norma DC promueve dos niveles de codificación: simple y cualificado. El Dublin Core simple comprende quince elementos, el Dublin Core cualificado implica el mismo número de elementos más un subgrupo de estos denominados cualificadores, que refinan la semántica de los quince elementos a fin de recuperar y localizar de mejor modo los recursos en Internet. Los elementos comprendidos por Dublin Core pueden ser divididos en tres subgrupos: elementos relacionados con el contenido del recurso, elementos relacionados con el recurso cuando es visto como una propiedad intelectual, elementos relacionados con la instanciación del recurso.

[7] Centro de biblioteca por ordenador en línea y entidad de investigación sin ánimo de lucro.

[8] Centro Nacional de Aplicaciones de Supercomputación. Es un organismo estadounidense relacionado con la investigación en el campo de la informática y las telecomunicaciones.

De igual manera es necesario transformar los elementos de ISAD(G) a Dublin Core para que los metadatos contenidos en ArchiVenHIS puedan ser enviados a los proveedores de servicios en formatos Dublin Core. En el Anexo 4 se puede consultar dicha transformación.

1.4. Sistemas gestores de documentos de archivo que implementan OAI-PMH

En la actualidad existen varios software que permiten a las instituciones gestionar sus fondos archivísticos, algunos de ellos poseen una implementación del protocolo para la recopilación de metadatos OAI-PMH. A continuación se reflejan los resultados de un estudio realizado a algunas de estas aplicaciones.

Archivo 3000: surge en el marco de la OdiloTID[9], una compañía que se encarga del desarrollo de aplicaciones de gestión de centros de documentación. Archivo 3000 Web (A3W) es una aplicación web diseñada y programada siguiendo el modelo líder de desarrollo *Java 2 Enterprise Edition*, empleando el lenguaje de programación Java[10]. Su desarrollo se rige por estándares abiertos, lo que la convierte en una aplicación multiplataforma, apropiada tanto para sistemas libres como propietarios. A3W se adapta fácilmente a cada cliente debido a las características que posee como el multilingüismo, importación de datos desde cualquier sistema productor de documentación, transmisión de contenidos mediante OAI-PMH, configuración de servicios web para comunicación fluida con otras aplicaciones, además de las que a continuación se exponen (ODILOTID, 2012):

- Utiliza la norma ISAD(G) para los niveles previstos: Fondo, Subfondo, Serie, Unidad documental compuesta y Unidad documental simple, además Archivo 3000 incluye Depósito, Grupo o Sección y Subserie o Parte de serie.
- Posibilidad de integrar imágenes y sonidos en los documentos descritos.
- Acceso a los distintos niveles de descripción a partir de múltiples puntos: nombres, fechas, títulos, materias, términos geográficos, fecha de entrada, fecha de producción, búsquedas truncadas, etcétera.
- Importación y exportación de datos.
- Permite realizar préstamos a investigadores en sala, préstamos a oficinas, etcétera.

[9] Compañía que se encarga del desarrollo de aplicaciones de gestión de centros de documentación.

[10] Lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90.

- Programa para Oficinas Productoras con posibilidad de realizar transferencias desde este.
- Posibilidad de búsqueda en línea por los usuarios y de configurar las consultas de estos.

ICA-AtoM: es un software creado para la empresa canadiense *Artefactual Systems* bajo patrocinio del Consejo Internacional de Archivos (ICA) y de Unesco[11]. Utiliza una capa de abstracción de bases de datos que lo hace ser compatible con Postgres, SQLite, SQLServer, Oracle, pero ha sido utilizado MySQL para su desarrollo. Utiliza el lenguaje de programación PHP para gestionar las peticiones y respuestas entre los clientes web y la lógica de la aplicación.

La descripción de materiales se basa en los estándares del ICA ISAD(G), ISAAR(CPF)[12], ISDF[13] e ISDIAH[14], aunque también se pueden crear las descripciones usando otros esquemas como Dublin Core, Mods (*Metadata object description schema*) o las *Rules for Archival Description* (RAD) canadienses (las opciones de configuración permiten elegir el modelo que se quiere utilizar para crear las descripciones). Además, tiene las siguientes características (HERRERA TEJADA, 2011):

- Soporte a los estándares del ICA.
- Capacidad de importar y exportar a EAD y EAC-CPF[15].
- Gestión de las relaciones entre los distintos tipos de registros.
- Gestión de listas de autoridades y de descriptores con las relaciones características de los tesauros.
- Soporte a OAI-PMH para exponer los contenidos de la base de datos en formato DC.
- Módulo de consulta disponible a través de la Web, y funciones para la publicación y gestión de bibliotecas de imágenes.

[11] Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura. Se fundó el 16 de noviembre de 1945 con el objetivo de contribuir a la paz y a la seguridad en el mundo mediante la educación, la ciencia, la cultura y las comunicaciones.

[12] Norma internacional sobre los registros de autoridad de archivos relativos a entidades, personas y familias.

[13] Norma internacional para la descripción de funciones, proporciona una orientación para la elaboración de descripciones de funciones de instituciones asociadas con la producción y la conservación de documentos de archivo.

[14] Norma internacional para describir instituciones que custodian fondos de archivo.

[15] Es un estándar de codificación de la información contextual sobre las personas, entidades corporativas y las familias en relación con los materiales de archivo.

- Interfaz de usuario traducible a distintos idiomas.

1.5. Análisis de las soluciones encontradas

El estudio de las soluciones anteriormente mencionadas permitió recopilar elementos necesarios para la elaboración de una propuesta de solución al problema planteado más completa. De cada uno de estos sistemas fueron tomados los aspectos positivos pues presentan similitudes con el sistema a desarrollar.

Sin embargo, ninguna fue adoptada en su totalidad como solución al problema planteado pues algunas son de carácter propietario, tal es el caso de Archivo 3000 que a pesar de las ventajas que proporciona su uso, su código no puede ser accedido. A pesar de esto su estudio fue útil para la investigación debido a que es un sistema líder en el mundo de la gestión archivística y permitió confirmar que la decisión de seleccionar OAI-PMH como solución es correcta. En el caso específico de ICA-AtoM, no pudo ser adoptado debido a que su desarrollo está basado en el *framework Symfony* y su estudio requeriría un esfuerzo adicional.

Actualmente ArchiVenHIS no es capaz de garantizar que otros sistemas recolecten los metadatos de los documentos en él descritos. Teniendo en cuenta este elemento y los expuestos anteriormente se concluye que es necesaria la implementación de un módulo que le permita a ArchiVenHIS comportarse como un proveedor de datos según lo establecido por el OAI-PMH.

1.6. Herramientas de software para OAI-PMH

El principal objetivo de la OAI es facilitar la distribución de información científica y académica, así como de cualquier contenido electrónico, a través de la Web. Desde los orígenes de OAI-PMH han ido surgiendo una serie de herramientas de software que soportan distintos aspectos, roles y funciones de la arquitectura del protocolo. Tanto en el sitio oficial de la OAI como en el del proyecto europeo Open Archives Forum (OAF), aparece un listado de gran utilidad a la hora de conocer dichas herramientas. Los programas registrados en estas fuentes son caracterizados por ser software libre y haber sido implementados por centros de investigación, bibliotecas, proyectos de investigación de universidades e incluso por programadores individuales, todos ellos pertenecientes a la comunidad OAI.

En el sitio oficial de la OAI se encuentran registradas actualmente un total de 30 herramientas, representadas en un listado por su nombre (con un vínculo hacia a la fuente), el implementador y una breve descripción de sus principales características que incluye la versión del protocolo con la que es

compatible. Este listado no permite tomar decisiones sobre que herramienta utilizar en determinados casos debido a que sus descripciones son muy sintetizadas.

Por otro lado, el listado ofrecido por la Open Archives Forum (OAF), incluye cuatro soluciones más, es decir que presenta un total de 34 herramientas. En este caso, se realiza una descripción mucho más detallada, compuesta por cinco apartados dentro de los cuales se encuentran los atributos descriptivos.

En ambos casos las publicaciones se encuentran desactualizadas pues ninguno de los listados contempla nuevas herramientas ni versiones de las ya existentes que han sido creadas en los últimos años, omitiendo algunas aplicaciones de gran relevancia.

A continuación se ofrece una descripción de un conjunto de plataformas para la creación de repositorios OAI, como resultado de un estudio realizado con el objetivo de determinar si alguna de ellas puede ser empleada en el desarrollo del presente trabajo de diploma.

EPrints: es un software desarrollado en el seno del Open Citation Project dirigido por Stevan Harnad en la Universidad de Southampton (UK). Está diseñado con el objetivo de ser fácil y rápido de instalación. Se distribuye bajo la licencia GNU, lo cual significa que el código fuente es accesible y modificable por cualquier programador, con la condición de que las modificaciones se hagan también accesibles públicamente.

Actualmente en su versión 3.3, utiliza el servidor web Apache y MySQL como gestor de base de datos, pudiendo funcionar sobre los sistemas operativos Linux, GNU o Solaris, así como en Windows gracias a un módulo adicional desarrollado para esta versión. El código de E-prints está escrito en Perl. Aunque por defecto se emplean metadatos Dublin Core, EPrints permite aplicar cualquier esquema de metadatos, y que el administrador decida qué campos se asignan a cada tipología documental. EPrints permite la recuperación de contenido mediante búsqueda textual (simple o avanzada, variando el número de campos de metadatos en los que se puede consultar) y navegación (BUENO-DE-LA-FUENTE y RODRÍGUEZ-MATEOS, 2007).

DSpace: es un software de código abierto que provee herramientas para la administración de colecciones digitales, y comúnmente es usada como solución de repositorio institucional. Soporta una gran variedad de datos organizados como ítems que pertenecen a una colección; cada colección pertenece a una comunidad. Fue liberado en el 2002, como producto de una alianza de Hewlett-Packard (HP) y el Massachussets Institute of Technology (MIT).

Es liberado bajo una licencia BSD[16] que permite a los usuarios personalizar o extender el software según se necesite. Su objetivo es permitir a una organización almacenar, describir y gestionar documentos electrónicos, distribuirlos a través de la Web mediante un sistema de búsqueda y recuperación de la información y finalmente proporcionar un sistema para el almacenamiento a largo tiempo de los documentos. Está programado en Java y corre sobre sistemas tipo Unix, precisa de un servidor web Apache y una base de datos relacional de código abierto PostgreSQL 8.0 o superior, aunque también se permite el uso de Oracle 9.0 o superior.

El esquema de metadatos utilizado para la descripción es Dublin Core cualificado, basado en el perfil de aplicación de Dublin Core para bibliotecas, que puede ser adaptado a través de la interfaz de administración. DSpace utiliza el motor de búsqueda de software libre Lucene, permitiendo realizar búsquedas tanto por campos de metadatos como en el texto completo, así como realizar búsquedas booleanas (BUENO-DE-LA-FUENTE y RODRÍGUEZ-MATEOS, 2007).

Fedora: (Flexible Extensible Digital Object and Repository Architecture) elaborado de manera conjunta por la Universidad de Virginia y la Universidad de Cornell. Proporciona un software libre para repositorios digitales que pueden emplearse como sustento de otros software de gestión como repositorios institucionales y bibliotecas digitales. Fedora está programado en Java, y permite emplear Apache y MySQL. Funciona sobre diversos sistemas operativos como Windows 2000, NT y XP, Solaris y Linux. Proporciona importantes funciones para la gestión de los repositorios, en cuanto al almacenamiento de ficheros, importación y exportación en XML y otros formatos de codificación como Fedora Object XML (FOXML) (BUENO-DE-LA-FUENTE y RODRÍGUEZ-MATEOS, 2007).

El empleo de una de estas herramientas facilitaría considerablemente el trabajo, sin embargo el sistema para el cual se construirá el módulo presenta características muy particulares y estas herramientas están desarrolladas bajo las restricciones y exigencias propuestas por sus creadores, por lo que el uso de una de ellas implicaría un profundo análisis, interpretación y modificación de su código fuente, lo que provocaría un mayor costo en tiempo. Por tal motivo se descarta el empleo de una de ellas para dar solución al problema que da origen a la investigación.

[16] Licencia de software libre que permite a los programadores utilizar, modificar y distribuir a terceros el código fuente y el código binario de un programa original con o sin modificaciones.

1.7. Tecnologías necesarias para la implementación del módulo Proveedor de datos para AchiVenHIS

En el presente epígrafe quedan plasmadas las tecnologías que se emplearán en la implementación de la solución propuesta. La selección de las tecnologías no fue del todo libre debido a que algunas como el *Framework* CodeIgniter 1.6.3, el gestor de base de datos MySQL, el servidor web Apache y el lenguaje de programación PHP fueron seleccionadas para el desarrollo del sistema en el año 2007 y el módulo en cuestión debe adaptarse a dicha selección. Sin embargo un estudio sobre dichas tecnologías permitió actualizarlas a nuevas versiones como MySQL 5.1 y PHP5, además se determinó emplear un nuevo entorno de desarrollo integrado.

Servidor web

Puede definirse como un "programa que escucha las peticiones de los usuarios o navegantes y las atiende o satisface. Por medio de la especificación de la búsqueda el servidor web buscará una página específica o ejecutará un programa, pero, necesariamente, enviará algún resultado sobre la búsqueda recibida. Los sistemas operativos más utilizados por los servidores son Windows y Linux, siendo este último más estable y por lo tanto de uso más frecuente" (F. ROMERO, 2007).

Apache 2.2

Es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos (HTTP 1.1). Entre sus características se destacan:

- Multiplataforma.
- Conforme al protocolo HTTP/1.1.
- Puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo y las API[17] de programación que proporciona.
- Permite emplear diversos lenguajes en el lado del servidor.
- Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos.
- Se desarrolla de forma abierta.
- Permite la compresión de datos, las conexiones seguras y la utilización de URLs[18] amigables.

[17] Siglas de Application Programming Interface (Interfaz de Programación de Aplicaciones). Son funciones externas que se encuentran compiladas y almacenadas en librerías DLL.
[18] Siglas de Localizador Uniforme de Recurso. Se refiere a la dirección única que identifica a una página web en Internet.

Framework

Un *framework* es una estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un *framework* se puede considerar como una aplicación genérica incompleta y configurable a la que se le pueden añadir las últimas piezas para construir una aplicación concreta, en dependencia de las necesidades de los programadores.

CodeIgniter 1.6.3

Fue desarrollado por *Rick Ellis* para EllisLab[19]*.* Es un programa desarrollado en PHP para la creación de cualquier tipo de aplicación web bajo PHP. Está basado en el patrón arquitectónico Modelo-Vista-Controlador y tiene soporte para bases de datos MySQL, SQL Server, PostgreSQL y Oracle. Algunas de las características más relevantes de este *framework* son:

- Versatilidad: es capaz de trabajar en la mayoría de los entornos o servidores, incluso en sistemas de alojamiento compartido, donde sólo se tiene un acceso por FTP para enviar los archivos al servidor y donde no se tiene acceso a su configuración.
- Compatibilidad: es compatible con la versión PHP 4, lo que hace que se pueda utilizar en cualquier servidor, incluso en algunos antiguos. Funciona correctamente también en PHP 5.
- Flexibilidad: CodeIgniter es menos rígido que otros *frameworks.* Define una manera de trabajar específica, pero en muchos de los casos puede seguirse o no.
- Ligereza: el núcleo de CodeIgniter es bastante ligero, lo que permite que el servidor no se sobrecargue interpretando o ejecutando grandes porciones de código. La mayoría de los módulos o clases que ofrece se pueden cargar de manera opcional, sólo cuando se van a utilizar realmente.
- Documentación inteligible: la documentación de CodeIgniter es fácil de seguir y de asimilar, porque está escrita en modo de tutorial.

En la actualidad existen versiones más recientes de CodeIgniter que ofrecen numerosas ventajas, sin embargo no pueden ser empleadas en el desarrollo del módulo debido al tiempo que requiere la

[19] Empresa localizada en Bend (Oregón, EE. UU.) que desarrolla aplicaciones software en lenguaje PHP. La empresa es de propiedad privada y no tiene socios financieros de ningún tipo.

migración. Por tanto, se recomienda la migración del sistema a una versión del *framework* más actual que permita explotar sus mejoras.

Sistema gestor de base de datos (SGBD)

Compendiando una serie de conceptos de diferentes autores, se puede concluir que una base de datos es una recopilación de información interrelacionada y organizada, relativa a un asunto o propósito particular. Las bases de datos facilitan el almacenamiento de grandes cantidades de información, permitiendo su recuperación rápida y flexible. Posibilitan organizar y reorganizar la información, así como su distribución de diversas formas. Los sistemas encargados de proporcionar un entorno que permita almacenar, extraer y manipular los datos de las bases de datos son denominados Sistemas Gestores de Bases de Datos (SGBD). Estos pueden ser definidos como un conjunto de programas, procedimientos, lenguajes, etcétera, que suministran a los usuarios los medios necesarios para manejar los datos almacenados en la base de datos, manteniendo su integridad, confidencialidad y seguridad.

MySQL 5.1

Es un sistema de gestión de bases de datos relacional, multihilo y multiusuario cuya idea originaria surge de la *empresa Open Source MySQL AB* establecida inicialmente en Suecia en 1995. El objetivo de esta empresa era cumplir con el estándar SQL sin sacrificar velocidad, fiabilidad y usabilidad. Algunas de sus características son (MYSQL, 2012):

- Escrito en C[20] y C++[21].
- Combinaciones de registros de dos o más tablas (joins).
- Sistema de reserva de memoria muy rápido basado en hilos.
- Proporciona sistemas de almacenamiento transaccional y no transaccional.
- Soporte a grandes bases de datos.

Ventajas de MySQL

[20] Lenguaje que nació en la década del 70. Es de propósito general y, fue creado para desarrollar aplicaciones en el entorno UNIX. Sus creadores fueron Dennis Ritchie y Brian Kernighan. El lenguaje C está basado en otro más antiguo llamado, originalmente B. Pero, a diferencia de este, el C introducía el concepto de tipos de datos y creaba código ejecutable, mientras que el B creaba código interpretable.

[21] Lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos.

- Escalabilidad: es posible manipular bases de datos enormes, del orden de seis mil tablas y alrededor de cincuenta millones de registros, y hasta 64 índices por tabla.
- Conectividad: permite conexiones entre diferentes máquinas con distintos sistemas operativos.
- Es multihilo, con lo que puede beneficiarse de sistemas multiprocesador, lo que repercute directamente en una de sus principales ventajas: mayor rendimiento.
- Permite manejar multitud de tipos para columnas.
- Permite manejar registros de longitud fija o variable.

Lenguaje de Programación

En el Diccionario de Informática e Internet de Microsoft, un lenguaje de programación se define como "cualquier lenguaje artificial utilizado para definir una secuencia de instrucciones que la computadora podrá, finalmente, procesar y ejecutar" (MICROSOFT CORPORATION, 2000).

Los lenguajes de programación orientados a la Web se dividen en dos grupos, los del lado del cliente y los del lado del servidor. Los lenguajes de programación del lado del cliente se ejecutan en el navegador del usuario. Las páginas del cliente son muy dependientes del sistema donde se están ejecutando y esa es su principal desventaja, ya que cada navegador tiene sus propias características, incluso cada versión, y lo que puede funcionar en un navegador puede no funcionar en otro. Como ventaja se puede decir que ofrecen respuestas inmediatas a las acciones del usuario y permiten la utilización de algunos recursos de la máquina local (MATOS PADILLA, 2008).

Los lenguajes del lado del servidor son reconocidos, ejecutados e interpretados por el propio servidor. Son independientes del cliente por lo que son mucho menos rígidos respecto al cambio de un navegador a otro o respecto a las versiones de este (QUERO CATALINAS *et al.*, 2007).

Lenguajes del lado del cliente.

JavaScript

Es un lenguaje script[22] u orientado a documento desarrollado por la empresa Netscape Communications. Un programa en JavaScript se integra en una página web (entre el código HTML) y es el navegador el que lo interpreta, o sea, es un lenguaje interpretado y no compilado. Se utiliza en páginas web HTML para realizar tareas y operaciones en el marco de la aplicación cliente. El núcleo

[22] Fichero o secciones de código escritas en algún lenguaje de programación, como Visual Basic Script (VBScript), JavaScript, etc. Conjunto de instrucciones que permiten la automatización de tareas o conjunto de comandos que se utilizan con frecuencia.

de JavaScript incluye los elementos típicos de un lenguaje de programación: variables, sentencias, estructuras, operadores, etcétera. La extensión para cliente permite acceder a los objetos que utiliza el navegador y al DOM[23]. Añade soporte para el control de eventos, de tal forma que el programa puede interactuar con el usuario (QUERO CATALINAS *et al.*, 2007).

jQuery

Es una librería de JavaScript rápida y concisa que simplifica el trabajo con el documento HTML, el manejo de eventos, la animación, y las interacciones Ajax para un rápido desarrollo web. jQuery está diseñado para cambiar la forma en que se escribe JavaScript (THE JQUERY FOUNDATION, 2012).

Lenguaje del lado del servidor.

PHP

Siglas de Hypertext Preprocessor es un lenguaje script para el desarrollo de páginas web dinámicas del lado del servidor, cuyos fragmentos de código se intercalan fácilmente en páginas HTML.

PHP es capaz de realizar determinadas acciones de una forma fácil y eficaz sin tener que generar programas escritos en un lenguaje distinto al HTML. Esto se debe a que PHP ofrece un extenso conjunto de funciones para la explotación de bases de datos sin complicaciones (CIBERAULA, 2010).

Ventajas de PHP

- Lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, entre ellos MySQL y PostgreSQL.
- Posee una amplia documentación en su página oficial, entre la cual se destaca que 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.
- PHP generalmente es utilizado como módulo de Apache, lo que lo hace extremadamente veloz. Está completamente escrito en C, así que se ejecuta rápidamente utilizando poca memoria.
- Posee manejo de excepciones a partir de la versión 5.
- No requiere definición de tipos de variables.

[23] Document Object Model (Modelo de Objetos del Documento). Define cómo los objetos de una página Web son representados, qué atributos tienen y cómo se manipulan. DOM es una interfaz para que aplicaciones y scripts puedan acceder a un documento pudiendo modificar su estructura y su contenido.

Entorno de desarrollo integrado (IDE)

Es un conjunto de herramientas de programación tales como un editor de código, un compilador, un depurador y un constructor de interfaz gráfica, unidas en un programa para servir de gran utilidad al programador. Para el desarrollo de la aplicación se empleará el IDE NetBeans 7.0.

NetBeans 7.0

Es una aplicación de código abierto sin restricciones de uso escrita en Java. Posee las herramientas necesarias para crear aplicaciones web, de escritorio y para teléfonos celulares. Soporta variados lenguajes tales como Java, C, C++, PHP y JavaScript, entre otros. Es fácil de instalar, usar y corre sobre plataformas como Linux, Windows, Mac y Solaris. Entre sus principales características se pueden citar:

- Posee una integración completa en términos de administración básica y avanzada de MySQL.
- Integración con Sistemas de Control de Versiones
- El depurado de las aplicaciones es sencillo.
- Autocompleta código de JavaScript, CSS, HTML y PHP.
- Las aplicaciones basadas en la plataforma NetBeans generan instaladores para los sistemas operativos más usados.
- Soporte mejorado para consumir servicios web y conectarse a bases de datos.

eXtensible Markup Language (XML)

Formato simple de texto muy flexible derivado de SGML (ISO 8879). Originalmente diseñado para afrontar los retos de las grandes publicaciones electrónicas. XML es un lenguaje que permite jerarquizar y estructurar la información y describir los contenidos dentro del propio documento, así como la reutilización de partes del mismo. La información estructurada presenta varios contenidos (texto, imágenes, audio, etcétera) (LAMARCA LAPUENTE, 2011).

Ofrece un formato para la descripción de datos estructurados, lo que facilita que las declaraciones de contenido sean más precisas y los resultados de búsquedas sean más significativos. Proporciona interoperabilidad mediante un formato basado en estándares flexibles y abiertos, con formas nuevas de acceso a las bases de datos existentes y de entregar datos a clientes de la Web. Las aplicaciones

se pueden generar más rápidamente, su mantenimiento es más sencillo y pueden ofrecer fácilmente varias vistas de los datos estructurados.

Un documento XML tiene dos estructuras, una lógica y otra física. Físicamente, el documento está compuesto por unidades llamadas entidades. Una entidad puede hacer referencia a otra entidad, causando que esta se incluya en el documento. Cada documento comienza con una entidad documento, también llamada raíz. Lógicamente, el documento está compuesto de declaraciones, elementos, comentarios, referencias a caracteres e instrucciones de procesamiento, todos los cuales están indicados por una marca explícita (WORLD WIDE WEB CONSORTIUM, 1998).

XML es empleado por OAI-PMH para definir formatos de registro. Cualquier tipo de metadatos puede ser intercambiado vía OAI-PMH siempre que este pueda ser codificado como XML, por lo que se hace necesario su uso para emitir las respuestas a los proveedores de servicios que realicen solicitudes al sistema.

Herramientas CASE

Acrónimo de Computer Aided Software Engineering (Ingeniería de Software Asistida por Computadoras). Las herramientas CASE son un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software (INSTITUTO NACIONAL DE ESTADÍSTICA E INFORMÁTICA, 1999).

Visual Paradigm 8.0

Visual Paradigm para UML (VP-UML) es una herramienta de diseño UML diseñada para ayudar al desarrollo de software. VP-UML soporta los principales estándares de la industria tales como Lenguaje de Modelado Unificado (UML), SysML[24], BPMN[25], etcétera. Ofrece un completo conjunto de herramientas utilizado por los equipos de desarrollo de software en la captura de requisitos, planificación de software, planificación de controles, el modelado de clases, el modelado de datos, etcétera (VISUAL PARADIGM, 2012).

[24] Por sus siglas en inglés (Systems Modeling Language) es una notación de modelado para la ingeniería de sistemas. Tiene como objetivo proporcionar técnicas de modelado de una gran variedad de sistemas entre los que se incluyen equipos físicos, software, datos, personas, procedimientos e instalaciones.

[25] Acrónimo de Notación para el Modelado de Procesos de Negocio. Es una notación gráfica estandarizada que permite el modelado de procesos de negocio, en un formato de flujo de trabajo.

Es muy estable en cuanto a la compatibilidad con sistemas operativos, ayuda a una más rápida construcción de aplicaciones de calidad y a un menor costo. Permite generar código desde diagramas y documentación.

Lenguaje de modelado

Un lenguaje de modelado es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar un sistema o parte de él.

Lenguaje Unificado de Modelado (UML)

UML es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Su primera versión UML 1.0, sale a la luz en al año 1997, como resultado del intercambio y trabajo colaborativo entre varias personas que se habían dedicado a la confección de documentos relacionados con el análisis y diseño de sistemas informáticos como Grady Booch, James Rumbaugh, e Ivar Jacobson. Se le suponen características comunes en muchos productos de dibujo, como edición, dar formato, hacer zoom, dar color y diseño automático. Define reglas sintácticas que especifican cómo combinar elementos del lenguaje (JACOBSON *et al.*, 2000).

Actualmente está respaldado por el OMG (Object Management Group). Se utiliza para definir un sistema, para detallar los artefactos, documentar y construir. Desde el año 1995, UML es un estándar aprobado por la ISO como ISO/IEC 19501:2005 Information technology - Open Distributed Processing - Unified Modeling Language (UML) Versión 1.4.2. Cuenta con varios tipos de diagramas:

- Diagramas de estructura (enfatizan en los elementos que deben existir en el sistema modelado).
- Diagramas de comportamiento (enfatizan en lo que debe suceder en el sistema modelado).
- Diagramas de Interacción (un subtipo de diagramas de comportamiento, que enfatiza sobre el flujo de control y de datos entre los elementos del sistema modelado).

1.8. Metodología de desarrollo de software

El proceso de desarrollo de software implica cierto grado de riesgo, por lo que se requiere de una metodología de desarrollo capaz de conducir este proceso de manera que desarrolladores y clientes queden satisfechos con el producto en cuestión.

Una Metodología de Desarrollo de Software es un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos de software, que indican paso a paso todas las

actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben tener. Además detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla (MENÉNDEZ-BARZANALLANA ASENSIO, 2011).

Proceso Unificado de Desarrollo de Software (RUP)

El Proceso Unificado de Desarrollo (RUP), es un ejemplo de estas metodologías y es la que se adoptará para el desarrollo del sistema. Es el resultado de una convergencia entre Rational Approach y Objectory, su primera versión recibió el nombre de Rational Objectory Process y fue puesta en el mercado en el año 1998.

RUP unifica los mejores procesos de metodologías anteriores, es orientado a objetos y está preparado para desarrollar grandes y complejos proyectos. Se caracteriza por ser iterativo e incremental, dirigido por casos de uso y centrado en la arquitectura. Las actividades se agrupan en grupos lógicos, definiendo 9 flujos de trabajo principales que guían el desarrollo del software a través de las 4 fases: Inicio, Elaboración, Construcción y Transición, que componen el ciclo de vida del proyecto. En su modelación define un grupo de elementos que determinan quién, qué, cómo y cuándo se realizarán las actividades (JACOBSON *et al.*, 2000):

- Trabajadores (“quién”): definen el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos.
- Actividades (“cómo”): son tareas que tienen un propósito claro, son realizadas por los trabajadores.
- Artefactos (“qué”): productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.
- Flujo de actividades (“cuándo”): secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable.

Para el desarrollo del módulo será empleada la metodología de desarrollo RUP según lo que establece el Programa de mejora que se está llevando a cabo en la universidad con el objetivo de alcanzar una certificación internacional del nivel 2 del modelo CMMI[26].

1.9. Conclusiones parciales

- En la actualidad gran parte de los sistemas de gestión de documentos históricos tienen la capacidad de interoperar con otros sistemas.
- El protocolo para la interoperabilidad que más se adapta a las características de un SGDH es OAI-PMH.
- Se descarta la posibilidad de utilizar alguna de las aplicaciones encontradas en la literatura por no ajustarse a las características de la solución que se plantea.
- Se decide no emplear ninguna de las herramientas de software para OAI-PMH estudiadas debido a que resulta más conveniente desarrollar una propia para el sistema.
- Las tecnologías seleccionadas para el desarrollo de la solución están acordes a los requisitos y a las políticas del centro.

[26] Modelo de referencia para el crecimiento de capacidades y madurez, que se enfoca tanto en procesos de administración como de ingeniería de sistemas y software.

Capítulo II. Características del módulo para garantizar la prestación de servicios según OAI-PMH para ArchiVenHIS

Actualmente el SGDH ArchiVenHIS no ofrece a otros sistemas la posibilidad de recolectar los metadatos descritos en él. Los documentos custodiados por las instituciones que emplean dicho software pueden ser consultados únicamente desde el propio sistema a través de las funcionalidades de búsqueda y exploración que proporciona, es decir, no existe la posibilidad de que los documentos sean vistos desde otros sistemas. Esta capacidad de interoperar constituye, en la actualidad, un aspecto de gran importancia para los SGDH debido a que se contribuye considerablemente en la difusión de la memoria histórica de las instituciones que los utilizan. En el presente capítulo se realiza una propuesta para dar solución al problema anteriormente expuesto.

2.1. Descripción de la propuesta de solución

Con el desarrollo de un módulo que garantice que ArchiVenHIS se comporte como un proveedor de datos según OAI-PMH, se le proporciona al sistema una nueva funcionalidad que permite que otros recuperen la información en él descrita.

ArchiVenHIS podrá recibir peticiones provenientes de un proveedor de servicios a través los métodos GET y POST del protocolo para la transferencia de hipertexto HTTP. Al recibir una solicitud, deberá analizarla sintácticamente y luego decidir de qué tipo es dentro de los seis válidos posibles, o si el tipo de petición es incorrecta debe generar un error. Si la petición está correctamente elaborada, se procede a ensamblar una consulta SQL con los parámetros extraídos de la petición y enviarla a la base de datos, en caso de que la petición lo requiera. Finalmente se elabora una respuesta con los datos requeridos codificados en sintaxis XML y se envía a través de uno de los métodos anteriormente mencionados a quién realizó la petición.

Adicionalmente, contará con un administrador que se encargará de gestionar los recolectores que podrán realizar peticiones de metadatos. Si un recolector no está autorizado por el administrador, no podrá recolectar los metadatos descritos a través del sistema. Una vez insertado un recolector, el administrador podrá modificarlo, consultarlo y eliminarlo. Esta persona tendrá acceso también a los datos que se registran de las peticiones que son recibidas por el sistema y en caso de que sea necesario puede eliminarlas, tiene la posibilidad de conocer algunos datos estadísticos, entre ellos, la cantidad de peticiones recibidas por cada uno de los verbos, por estado, tipo de respuesta y recolector, y la cantidad de metadatos que ha recolectado cada proveedor de servicios. Además puede eliminar conjuntos de peticiones dependiendo de las características de las mismas.

2.2. Modelo de dominio

Un modelo de dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que trabaja el sistema (JACOBSON *et al.*, 2000). El modelo de dominio es una de las alternativas que ofrece RUP para la comprensión del contexto cuando existe poca estructuración de los procesos del negocio. A través de él se definen los principales conceptos que se manejan en el dominio del sistema, sus partes y sus relaciones. Su elaboración posibilita a los involucrados manejar un vocabulario común lo que permite un mejor entendimiento del contexto en que se sitúa el sistema. Se representa mediante de un diagrama de clases de UML.

Descripción de los principales conceptos

A continuación se proporciona una descripción de los conceptos fundamentales que intervienen en el dominio del sistema, que permitirán una mejor comprensión del diagrama del modelo de dominio.

Administrador: es la persona encargada de controlar el acceso a los datos a través de configuraciones que determinarán de qué proveedores de servicios se aceptarán peticiones. Además puede solicitar que se generen reportes que le posibiliten mantener un control estadístico del comportamiento del sistema.

Reporte estadístico: representa las estadísticas del sistema en cuestiones referentes al proveedor de datos, específicamente con los recolectores y las peticiones que son recibidas.

Proveedor de servicios: es un sistema informático que se encarga de realizar peticiones al sistema con el objetivo de recolectar metadatos que sean de su interés.

Proveedor de datos: recibe las peticiones enviadas por el proveedor de servicios, las analiza y en dependencia del resultado obtenido en dicho análisis, genera un XML como respuesta a la solicitud recibida.

Petición: solicitud que realiza el proveedor de servicios al proveedor de datos. Puede ser de seis tipos diferentes: GetRecord, ListRecords, ListIdentifiers, ListSets, ListMetadataFormats e Identify. Además, en depencia de su tipo puede o no contener argumentos, estos últimos pueden ser una fecha inicial, una fecha final o un conjunto determinado.

Respuesta: es lo que se genera cuando un proveedor de servicios envía una petición a un proveedor de datos. Se codifica en sintaxis XML y puede ser un error o una respuesta normal.

Documento: se trata de las descripciones archivísticas contenidas por el sistema.

Diagrama de clases del dominio

El diagrama del modelo de dominio permite representar de manera visual los principales conceptos que conforman el dominio del problema, así como las relaciones entre estos.

Diagrama de clases del dominio

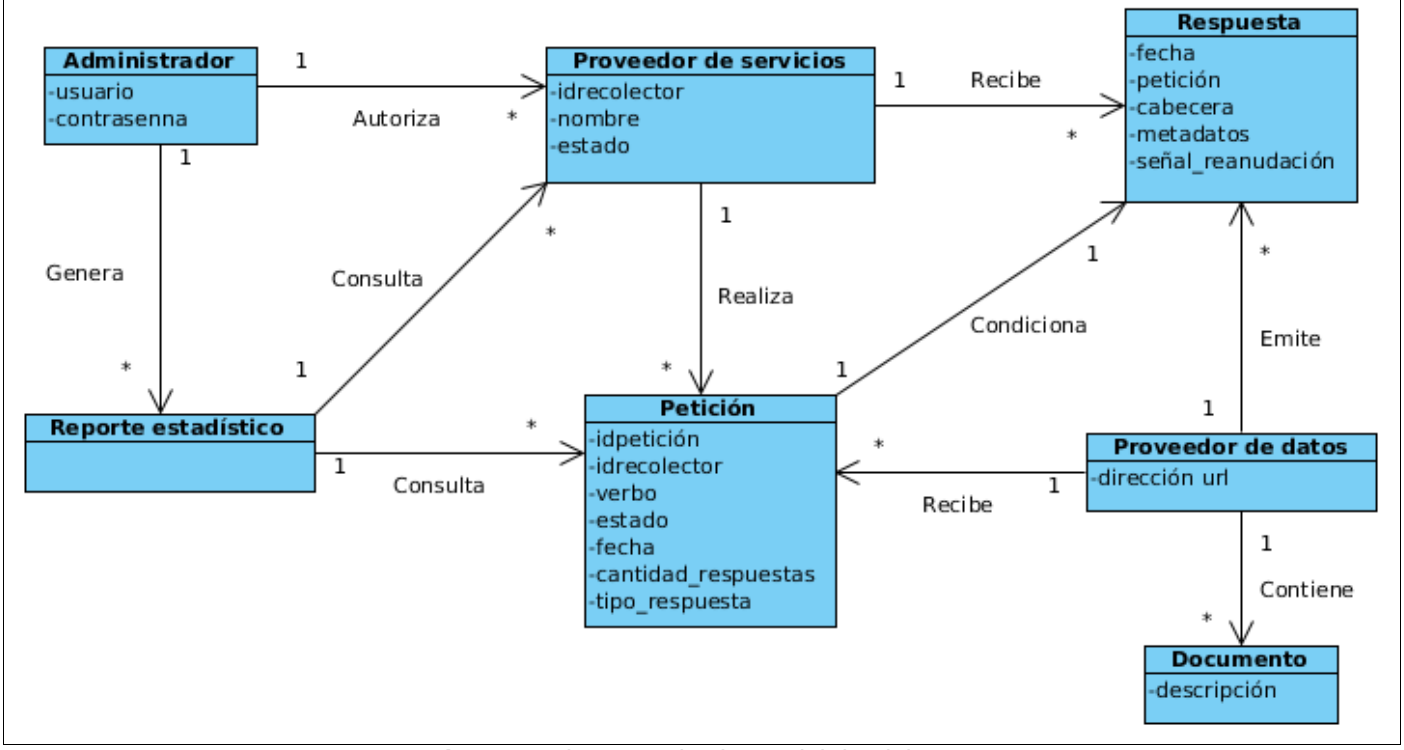


Figura 1. Diagrama de clases del dominio.

2.3. Requisitos del sistema

Requisitos funcionales

Los requisitos funcionales son condiciones o capacidades que el sistema debe cumplir. Con el desarrollo del módulo el SGDH ArchiVenHIS debe ser capaz de:

RF1- Permitir recolección de metadatos.

El sistema debe permitir que cualquier proveedor de servicios establezca una comunicación con él con el objetivo de recolectar sus metadatos.

RF2- Proveer metadatos en formato Dublin Core.

Utilizar Dublin Core como uno de los formatos de metadatos para la transmisión de las descripciones de los documentos.

RF3- Proveer metadatos en formato EAD.

Utilizar EAD como uno de los formatos de metadatos para la transmisión de las descripciones de los documentos.

RF4- Validar pedidos.

Realizar una estricta validación de las peticiones pues de ellas depende el éxito de la respuesta.

RF5- Agrupar las descripciones mediante una estructura de conjuntos (sets)

Cada fondo constituirá un set.

RF6- Controlar el flujo de la transmisión de metadatos.

Controlar el flujo de transmisión de los metadatos en caso de que se originen respuestas muy extensas.

RF7- Adicionar recolector.

Permitir que el administrador del sistema adicione los datos de los recolectores que podrán realizar recolecciones de metadatos.

RF8- Modificar recolector.

Permitir que se modifiquen los datos de los recolectores que se encuentran registrados.

RF9- Consultar recolector.

Permitir que se consulten los datos de los recolectores que se encuentran registrados en el sistema, atendiendo a determinados criterios introducidos por el usuario, por ejemplo un ip o un estado.

RF10- Eliminar recolector.

Permitir que los recolectores sean eliminados.

RF11- Registrar solicitud.

Almacenar los datos de todas las solicitudes que sean recibidas por el sistema.

RF12- Consultar solicitud.

Obtener los datos referentes a las peticiones que han sido recibidas por el sistema.

RF13- Eliminar solicitud.

Eliminar las peticiones que se han registrado en el sistema. Esta eliminación debe poder realizarse petición por petición o por pequeños grupos de peticiones.

RF14- Controlar el acceso a los datos.

Verificar que quien solicita los metadatos sea un proveedor de servicios autorizado por el administrador antes de suministrarle una respuesta.

RF15- Determinar cantidad de peticiones recibidas atendiendo al tipo de respuesta.

Permitir que se conozca la cantidad de peticiones que generaron un error o una respuesta exitosa, esto debe poder ser en un período de tiempo y atendiendo al estado, verbo y/o recolector.

RF16- Determinar cantidad de peticiones recibidas atendiendo al estado.

Permitir que se conozca la cantidad de peticiones que fueron aceptadas o rechazadas en un determinado período de tiempo, atendiendo al verbo, tipo de respuesta y/o recolector.

RF17- Determinar cantidad de peticiones recibidas atendiendo al verbo.

Permitir que se consulte la cantidad de peticiones recibidas por el sistema atendiendo al verbo que la compone en un período de tiempo determinado, en dependencia de su estado, tipo de respuesta y recolector que la realizó.

RF18- Determinar la cantidad de metadatos que ha recolectado cada recolector.

Informar la cantidad de metadatos que cada proveedor de servicios ha recolectado a través del sistema.

RF19- Eliminar conjuntos de peticiones.

Se debe permitir que el usuario elimine grandes grupos de peticiones, por ejemplo todas las peticiones suministradas en determinado período de tiempo.

Requisitos no funcionales

Los requisitos no funcionales definen propiedades y restricciones del sistema.

- **Usabilidad**

RNF1- El sistema podrá ser utilizado con el fin de cosechar metadatos por cualquier proveedor de servicios que cumpla con las especificaciones que establece el protocolo OAI-PMH y que posea permisos de acceso otorgados por el administrador.

- **Portabilidad**

RNF2- El sistema debe ser independiente de la plataforma.

- **Soporte**

RNF3- Se debe implementar el sistema siguiendo el estándar de codificación.

RNF4- Los componentes de software que integran la solución se organizarán de forma modular.

- **Diseño**

RNF5- Lenguaje de programación: PHP.

RNF6- *Framework* de desarrollo: CodeIgniter 1.6.3.

RNF7- Entorno de desarrollo integrado: NetBeans.

- **Software**

RNF8- Instalar en el servidor de bases de datos el SGBD MySQL 5.1 o superior.

RNF9- Instalar en el servidor web, Apache 2.2 o superior configurado adecuadamente para trabajar con CodeIgniter.

RNF10- Navegador web en los clientes: Mozilla Firefox 11.0 o superior, Google Chrome 17.0 o superior.

- **Documentación y ayuda del sistema**

RNF11- El sistema notifica a los usuarios acerca del resultado, exitoso o no, de las acciones realizadas.

RNF12- El usuario autenticado con permisos para realizar las funciones que se añaden al sistema debe poder contar con una ayuda incluida en la aplicación.

- **Hardware**

RNF13- Servidor de base de datos y servidor web con las siguientes prestaciones:

RAM: 1 GB.

Procesador: dos núcleos a 2.4 GHZ +.

2.4. Modelo de casos de uso

El modelo de casos de uso permite que los desarrolladores del software y los clientes lleguen a un acuerdo sobre los requisitos, es decir, sobre las condiciones y posibilidades que debe cumplir el sistema. Describe lo que hace el sistema para cada tipo de usuario (BRITO ACUÑA, 2009).

Actores

Los actores del sistema representan el rol que juega una o varias personas, un equipo o un sistema automatizado, son parte del sistema y pueden intercambiar información con él o ser recipientes

pasivos de información. En este caso, los actores que interactúan con el sistema quedan definidos en la siguiente tabla.

Actor	Descripción
Proveedor de servicios	Es el software encargado de iniciar la comunicación entre él y el proveedor de datos, enviando una petición a este último.
Administrador	Es la persona que se encarga de la gestión de los proveedores de servicios, es decir, es quien debe insertar, modificar, consultar y eliminar los recolectores. Además podrá consultar determinados datos de las peticiones que se han recibido.

Tabla 1. Actores del sistema.

Patrón de casos de uso

En la confección del diagrama de casos de uso se aplicará el patrón CRUD (*Create, Read, Update and Delete*) Completo ya que permite, de manera simple, modelar los casos de uso que poseen las operaciones crear, leer, actualizar y eliminar.

El patrón CRUD Completo consiste en un caso de uso para administrar la información, que permite modelar las diferentes operaciones para administrar una entidad de información, tales como crear, leer, cambiar y eliminar o dar de baja. Este patrón deberá ser usado cuando todas las operaciones contribuyen al mismo valor de negocio y todas son cortas y simples (ECURED, 2012).

En el caso de la solución que se propone es empleado para modelar las acciones adicionar, consultar, modificar y eliminar recolector. Todas se engloban en el caso de uso denominado Gestionar recolector que se describirá más adelante.

Además es utilizada una variante de este patrón llamada CRUD Parcial, mediante la cual pueden modelarse solo algunos de los casos de CRUD Completo. Para la confección del diagrama de casos de uso es empleada con el objetivo de unificar las funciones listar y eliminar peticiones en un único caso de uso denominado Gestionar petición.

Diagrama de casos de uso

Diagrama de casos de uso del sistema

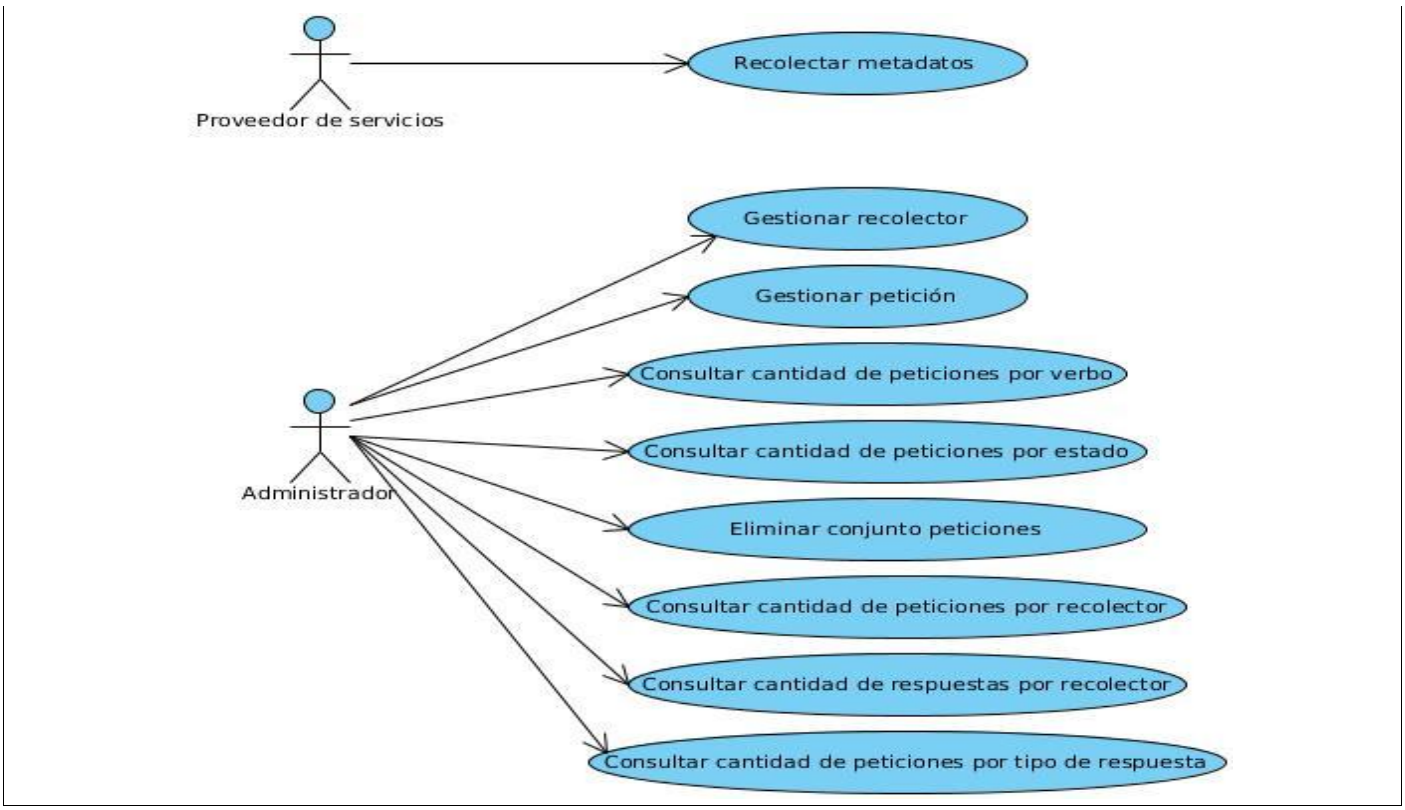


Figura 2. Diagrama de casos de uso.

Descripción de los casos de uso

A continuación se realiza la descripción de dos casos de uso que se consideran imprescindibles para el funcionamiento del sistema. La descripción del resto de los casos de uso puede consultarse en el expediente de proyecto.

Caso de uso Recolectar metadatos

Objetivo	Recolectar metadatos de interés.
Actores	Proveedor de servicios: (Inicia) Envía peticiones al proveedor de datos.
Resumen	El caso de uso inicia cuando el proveedor de servicios envía una petición al proveedor de datos, este último debe realizar las operaciones pertinentes, elaborar una respuesta con los datos solicitados y ponerla a disposición de quien realizó la petición, de este modo finaliza el caso de uso.
Complejidad	Media
Prioridad	Alta

Precondiciones	Se ha enviado una solicitud desde un proveedor de servicios.
Postcondiciones	Se ofrece una respuesta al proveedor de servicios que realizó la petición.
Flujo de eventos	
Flujo básico Recolectar metadatos	
Actor	**Sistema**
1- Envía una petición.	2- Recibe la petición.
	3- Verifica que el proveedor que realizó la petición se encuentre activo en la base de datos.
	4- Valida la petición.
	5- Inserta los datos de la petición en la base de datos.
	6- Realiza la búsqueda en la base de datos.
	7- Confecciona respuesta codificada en sintaxis XML y la muestra.
	8- Termina el caso de uso.
Flujos alternos	
3 El proveedor que realizó la petición no se encuentra en la base de datos	
Actor	**Sistema**
	4- Inserta el recolector como inactivo.
	5-Inserta los datos de la petición en la base de datos.
	6- Confecciona respuesta con el mensaje de error codificado en sintaxis XML y la muestra.
	7- Termina el caso de uso.
3 El proveedor que realizó la petición no se encuentra activo en la base de datos	
Actor	**Sistema**
	4- Inserta los datos de la petición en la base de datos.
	5- Confecciona respuesta con el mensaje de error codificado en sintaxis XML y la muestra.
	6- Termina el caso de uso.
4 Se envía una solicitud inválida.	
Actor	**Sistema**
	5- Inserta los datos de la petición en la base de datos.
	6- Confecciona respuesta con el mensaje de error codificado en sintaxis XML y la muestra.
	7- Termina el caso de uso.

Tabla 2. Descripción del caso de uso Recolectar metadatos.

Caso de uso Gestionar recolector

Objetivo	Mantener un control de los proveedores de servicios a los que se le suministra datos.
Actores	Administrador: (Inicia) Accede a la interfaz de usuario que le permite realizar las acciones de las que consta el caso de uso.
Resumen	El caso de uso inicia cuando el administrador necesita insertar un nuevo recolector, modificar o eliminar sus datos, se realiza la operación solicitada y muestra un mensaje indicando el resultado de la acción, finalizando así el caso de uso.
Complejidad	Media
Prioridad	Alta
Precondiciones	El usuario debe estar autenticado en el sistema como administrador.
Postcondiciones	Se garantizó que los datos fueran accedidos solo por quien tenía autorización.

Flujo de eventos	
Sección Adicionar recolector	
Actor	**Sistema**
1- Se autentica en el sistema.	
2- Selecciona la opción Insertar recolector.	3- Muestra un formulario con el campo ip donde se debe introducir la ip del recolector que se desee adicionar.
4- Introduce los datos requeridos por el sistema.	5- Verifica que se introdujeron datos.
	6- Verifica que la ip introducida sea válida.
	7-Verifica que el recolector no se encuentre en la base de datos.
	8- Guarda los datos en la base de datos y muestra el mensaje de confirmación “El recolector se insertó satisfactoriamente”.
	9- Finaliza el caso de uso.
Flujos Alternos	
7 El recolector ya existe.	
Actor	**Sistema**
	8- Muestra un mensaje el mensaje de error “El recolector especificado ya existe”.
	9- Finaliza el caso de uso.
6 IP no válida.	
Actor	**Sistema**

	7- Muestra el mensaje de error “La ip especificada no es válida”.
	8- Finaliza el caso de uso.
5 Campos vacíos.	
Actor	**Sistema**
	6- Muestra el mensaje de error “Debe especificar un recolector”.
	7- Termina el caso de uso.
Sección Listar recolector	
Actor	**Sistema**
1- Se autentica en el sistema.	
2- Selecciona la opción Administrar del bloque Recolector OAI.	3- Muestra un formulario para introducir los datos de los recolectores que se desean listar.
4- Introduce los datos.	
5- Presiona el botón Buscar.	6- Muestra un listado con los datos de los recolectores solicitados.
	7- Finaliza el caso de uso.
Flujos alternos	
6 No existen recolectores con las características especificadas	
Actor	**Sistema**
	6- Muestra el mensaje “La búsqueda no arrojó ningún resultado”.
	7- Finaliza el caso de uso.
Sección Modificar recolector	
Actor	**Sistema**
1- Se autentica en el sistema.	
2- Selecciona la opción Listar recolectores.	
3- Presiona el vínculo para modificar el estado de un recolector determinado.	4- Modifica el estado del recolector y muestra el mensaje “El recolector ha sido modificado”.
	5- Finaliza el caso de uso.
Sección Eliminar recolector	
Actor	**Sistema**
1- Se autentica en el sistema.	
2- Selecciona la opción Listar recolectores.	
3- Presiona el vínculo eliminar.	4- Elimina el recolector seleccionado y muestra el mensaje “Se ha eliminado el recolector seleccionado”.
	5- Finaliza el caso de uso.
Flujos alternos.	
3 Se seleccionan varios recolectores para ser eliminados	

Actor	Sistema
3- Selecciona los recolectores que desea eliminar. 4- Presiona el botón Eliminar.	5- Elimina los recolectores seleccionados y muestra el mensaje “Los recolectores seleccionados fueron eliminados”.
	6- Finaliza el caso de uso.

Tabla 3. Descripción del caso de uso Gestionar recolector.

2.5. Análisis del sistema

Durante el análisis, se analizan los requisitos que se obtuvieron en la captura de requisitos, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que ayude a estructurar el sistema entero, incluyendo la arquitectura (BRITO ACUÑA, 2009). Para un mejor entendimiento de los modelos se emplean los siguientes prototipos.

CI_ [Nombre de la Clase]: representan las formas de interacción entre los actores y el sistema, ventanas, formularios, comunicación con otros sistemas.

CC_[Nombre de la Clase]: constituyen las clases que coordinan y controlan los procesos para satisfacer las necesidades surgidas a partir del levantamiento de requisitos. Se encargan del procesamiento más complejo dentro del sistema.

CE_ [Nombre de la Clase]: modelan la información almacenada en el sistema.

Diagramas de interacción

Los diagramas de colaboración muestran las interacciones entre los objetos que componen el diagrama de clases del análisis, creando enlaces entre ellos y añadiendo mensajes a estos enlaces. Seguidamente se muestran los diagramas de colaboración correspondientes a los casos de uso que se considera son indispensables para el sistema, el resto puede ser consultado en el Anexo 5.

CU Recolectar metadatos

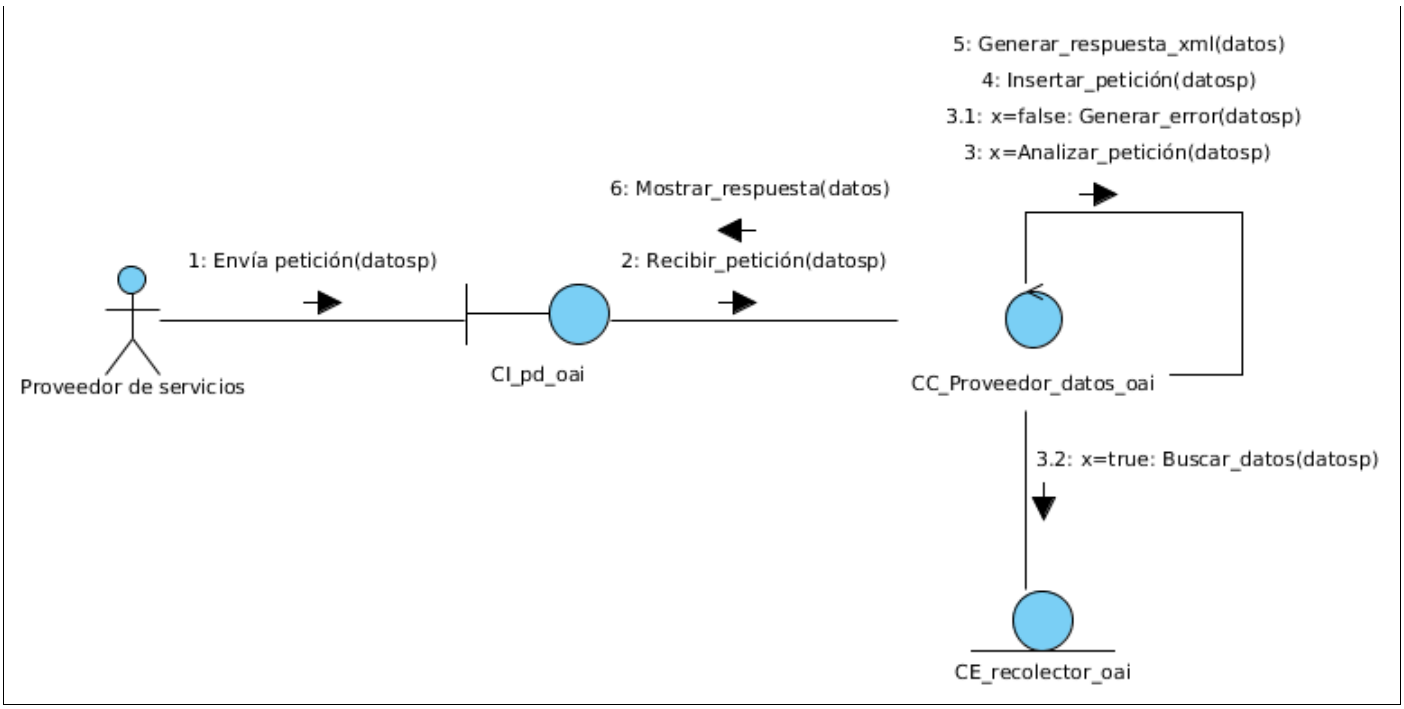
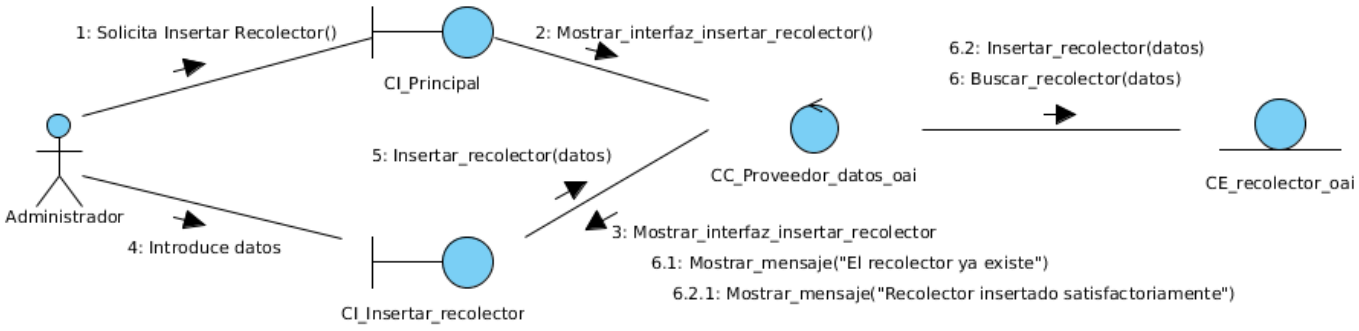


Figura 3. Diagrama de interacción del caso de uso Recolectar metadatos.

CU Gestionar recolector

SC1 Insertar recolector



SC2 Listar recolector

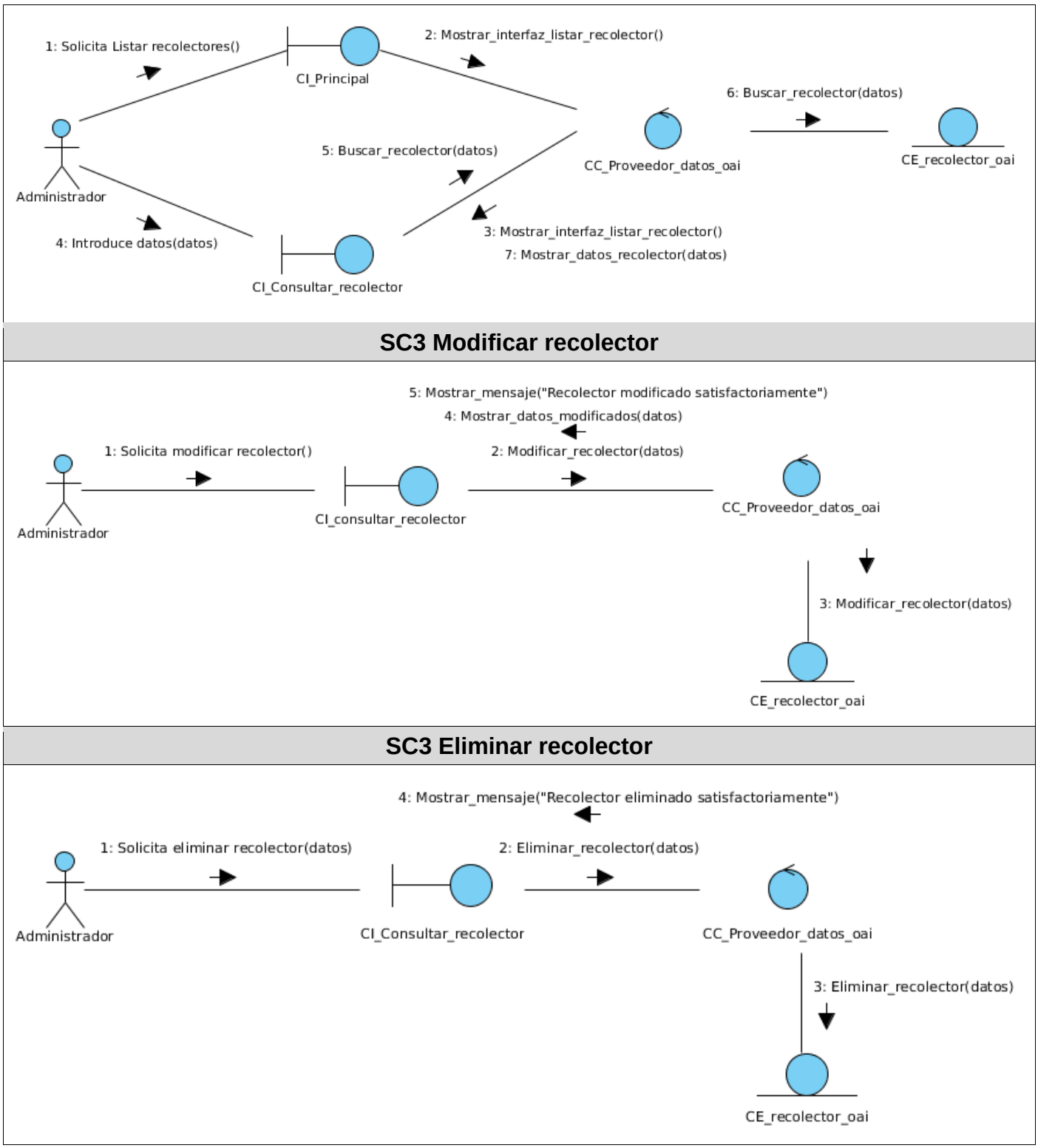


Figura 4. Diagrama de interacción del caso de uso Gestionar recolector.

2.6. Diseño del sistema

El modelo de diseño es una abstracción de la implementación del sistema. Se utiliza para concebir y para documentar el diseño del sistema de software. Es un producto de trabajo integral y compuesto que abarca todas las clases de diseño, subsistemas, paquetes, colaboraciones y las relaciones entre ellos (IBM CORPORATION, 2006).

Diagramas de clases del diseño

Un diagrama de clases representa las clases que serán utilizadas dentro del sistema y las relaciones que existen entre ellas. Sirve para visualizar las relaciones entre las clases involucradas en el sistema, las cuales pueden ser asociativas, de herencia, de uso y de convencimiento.

CU Recolectar metadatos

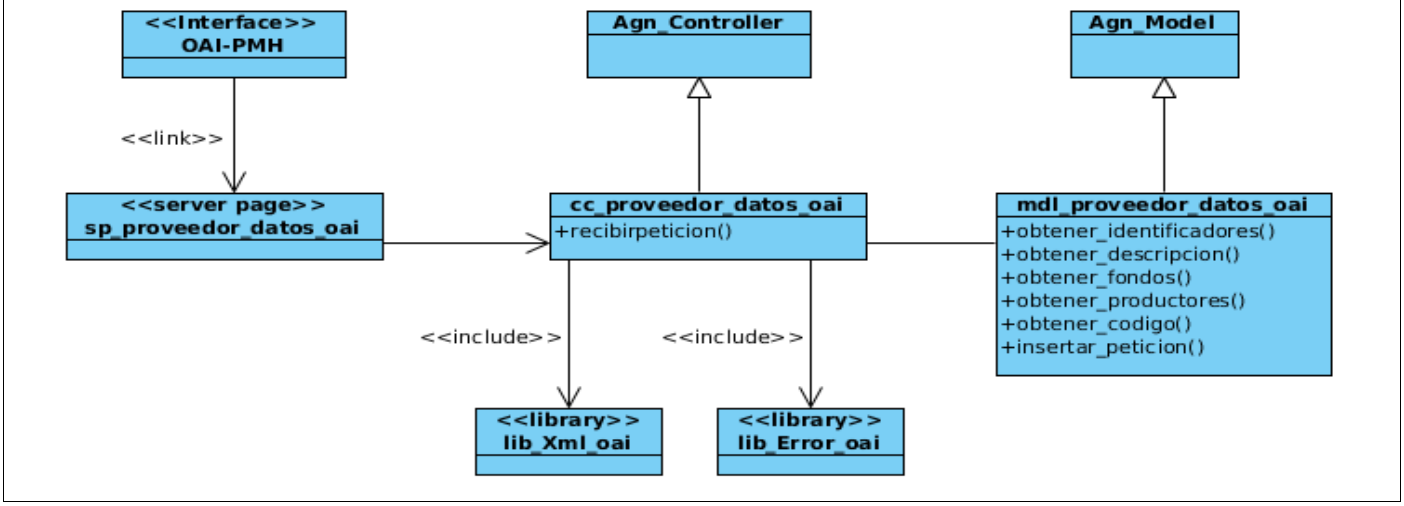
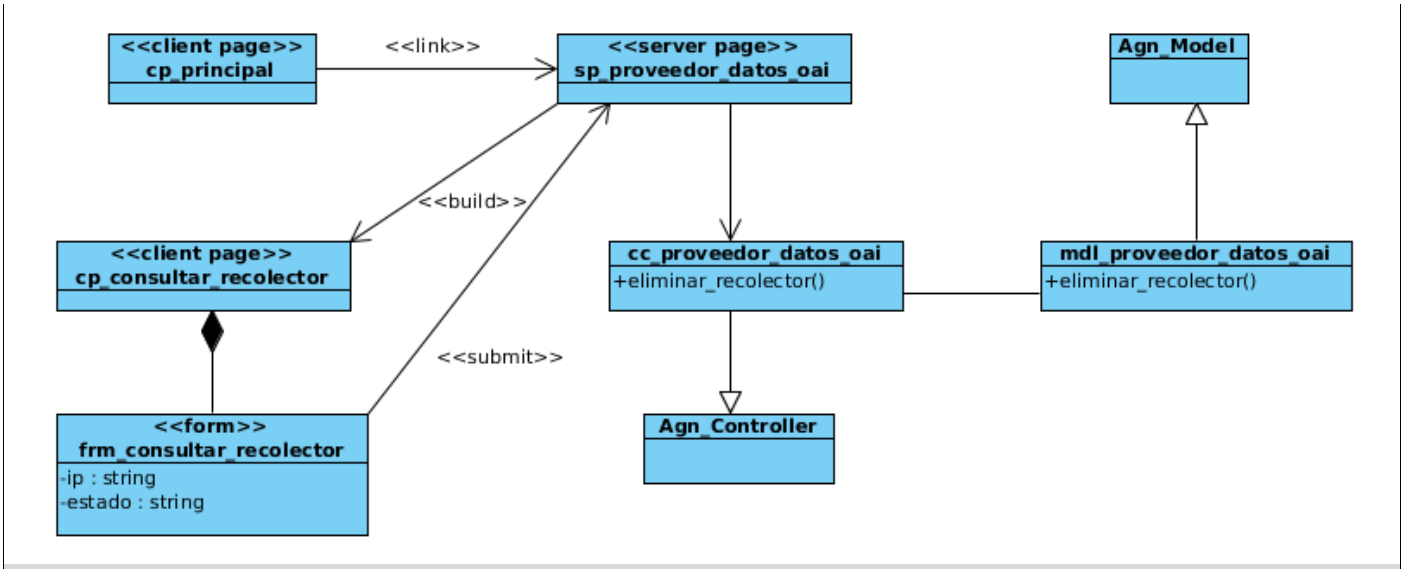


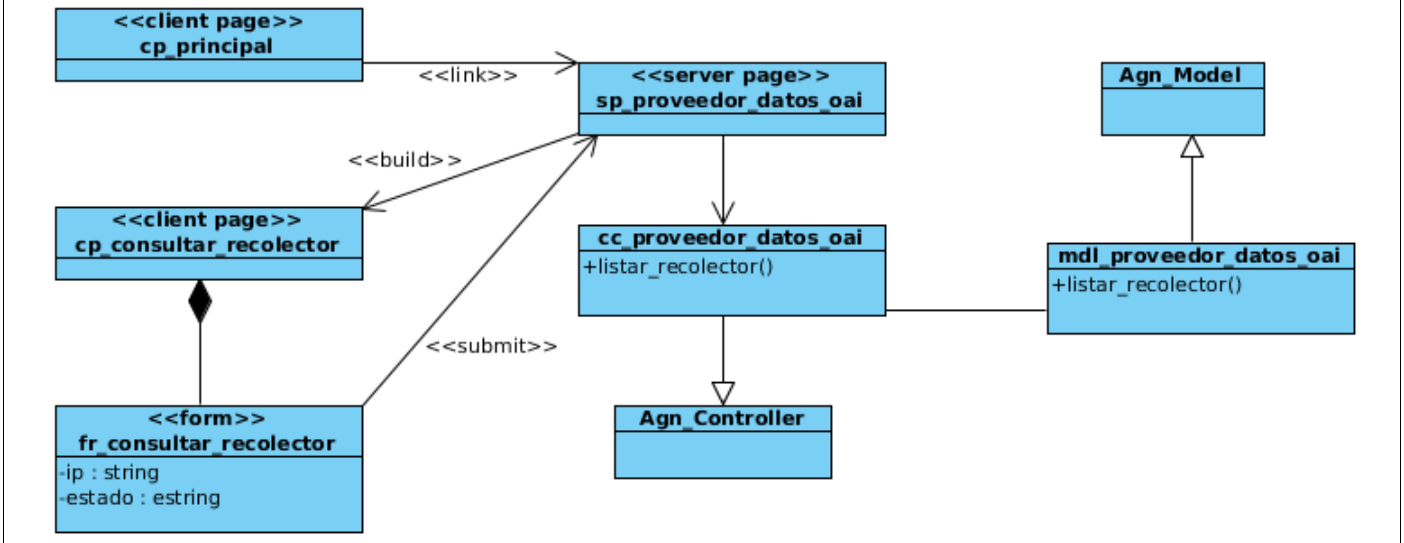
Figura 5. Diagrama de clases del caso de uso Recolectar metadatos.

CU Gestionar recolector

SC Insertar recolector



SC Listar recolector



SC Modificar recolector

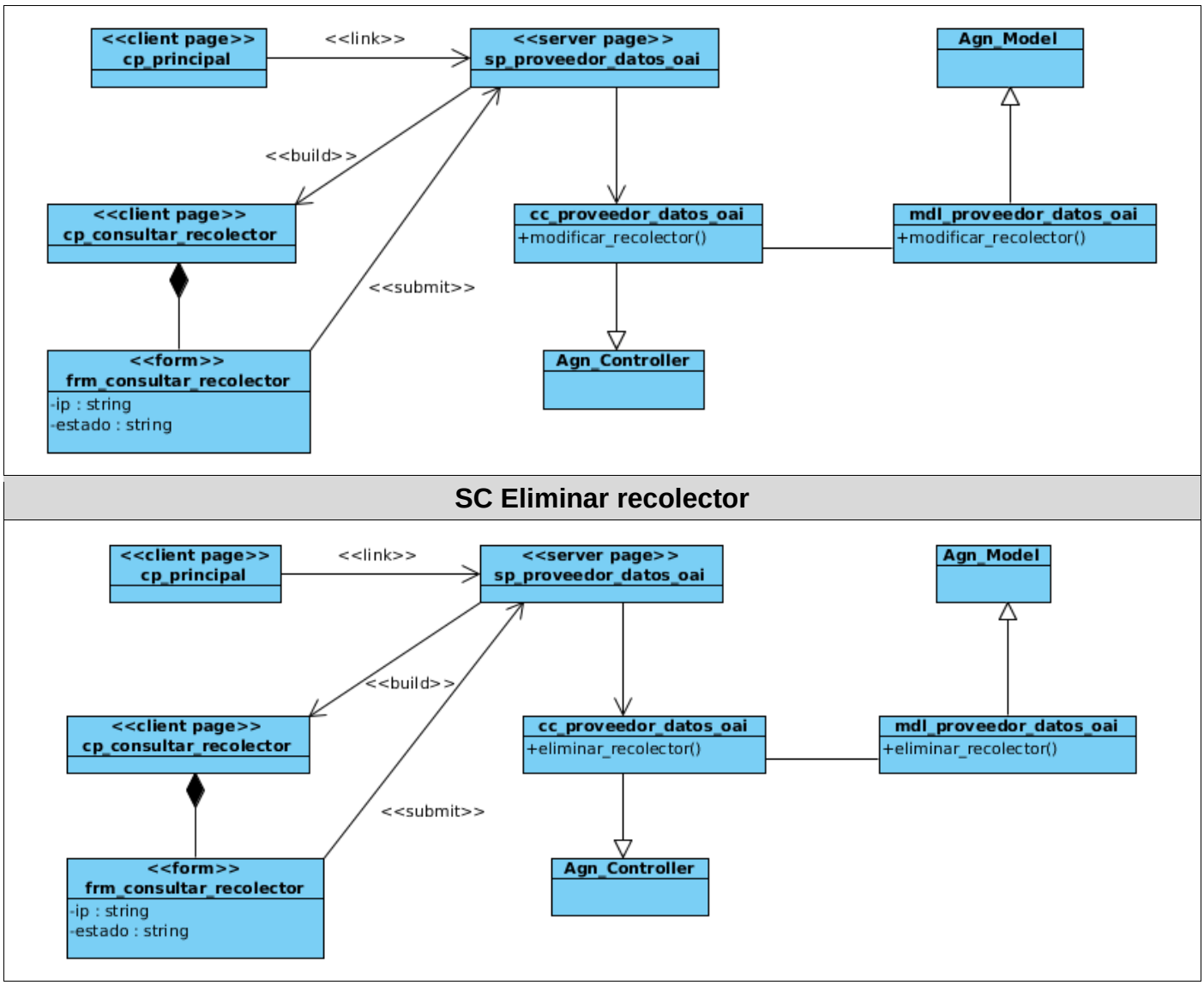


Figura 6. Diagrama de clases del caso de uso Gestionar recolector.

2.7. Diseño de la base de datos

El módulo trabaja con la base de datos que posee ArchiVenHIS. Los datos de los documentos que son enviados al proveedor de servicios que los solicite se obtienen directamente de dicha base datos, por lo que no será necesario construir una nueva, solamente se realizarán las trasformaciones pertinentes para enviar los metadatos en formato Dublin Core o EAD, según lo especificado en la petición.

Fue necesario adicionar dos nuevas tablas a dicha base de datos, una que permite almacenar los datos referentes a cada petición que se recibe y otra donde se guarda la información de los proveedores de servicios que pueden recolectar metadatos en el sistema.

Modelo de datos

Un modelo de datos no es más que la representación de un fenómeno de la realidad objetiva a través de los objetos, sus propiedades y las relaciones que se establecen entre ellos (MATO GARCÍA, 1999).

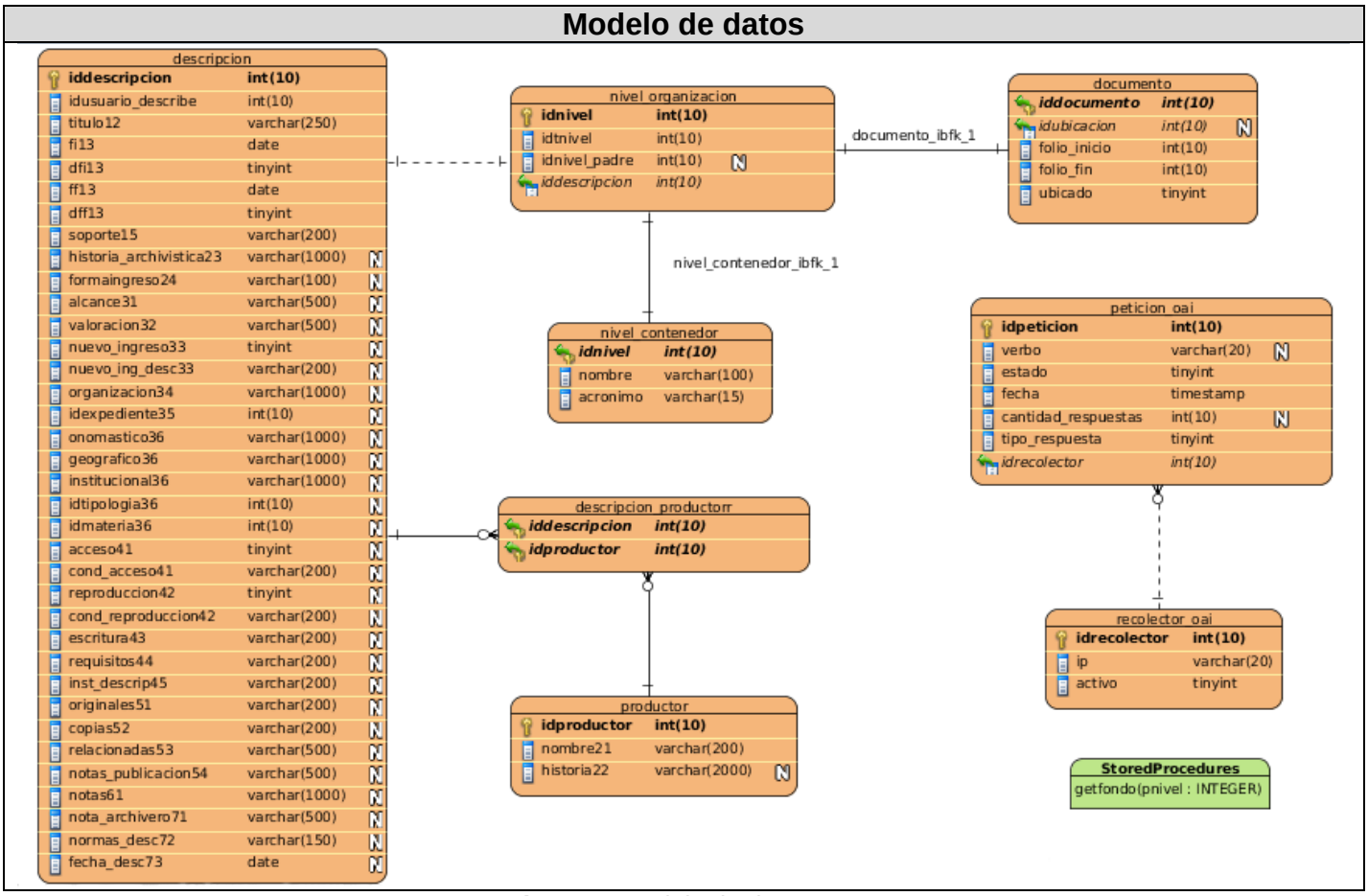


Figura 7. Modelo de datos.

Descripción de las entidades del modelo

Las siguientes tablas contienen una descripción de las relaciones que fue necesario añadir para el funcionamiento del módulo. Las descripciones del resto de las tablas pueden ser consultadas en el expediente de proyecto.

Nombre de la entidad	recolector_oai		
Descripción de la entidad	Almacena los datos de los recolectores que pueden realizar peticiones al sistema.		
Nombre del atributo	**Descripción**	**Tipo**	**Puede ser nulo**

idrecolector	Identifica las tuplas en la tabla, valor auto incremental generado por el gestor.	int(10)	No
ip	Representa la dirección ip del recolector.	varchar(20)	No
activo	Representa el estado del recolector. Toma valor 1 si se aceptarán sus peticiones y 0 en caso contrario.	tinyint	No

Tabla 4. Descripción de la entidad recolector_oai.

Nombre de la entidad	peticion_oai		
Descripción de la entidad	Almacena los datos de las peticiones que se realizan al sistema.		
Nombre del atributo	**Descripción**	**Tipo**	**Puede ser nulo**
idpeticion	Identifica las tuplas en la tabla, valor auto incremental generado por el gestor.	int(10)	No
idrecolector	Representa al recolector que realizó la petición.	int(10)	No
verbo	Representa el verbo por el cual se realizó la petición.	varchar(20)	Si
estado	Representa el estado de la solicitud, si fue aceptada toma valor 1 y si fue denegada toma valor 0.	tinyint	No
fecha	Fecha en que se realizó la petición.	timestamp	No
cantidad_respuestas	Cantidad de metadatos recolectados.	int(10)	Si
tipo_respuesta	Representa el tipo de respuesta que generó la solicitud. Si fue un error toma valor 0 y en caso contrario toma valor 1.	tinyint	No

Tabla 4. Descripción de la entidad peticion_oai.

2.8. Arquitectura

La arquitectura de software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución (ALONSO MARTÍNEZ y SOLER ARCIA, 2010).

Modelo-Vista-Controlador (MVC)

MVC es un patrón de arquitectura de software encargado de separar la lógica de negocio de la interfaz del usuario y es más utilizado en aplicaciones web, ya que facilita la funcionalidad, el mantenimiento y la escalabilidad del sistema, de forma simple y sencilla (BAHIT, 2011).

MVC divide las aplicaciones en tres niveles de abstracción:

Modelo: representa la lógica de negocios. Es el encargado de acceder de forma directa a los datos actuando como “intermediario” con la base de datos.

Vista: es la encargada de mostrar la información al usuario de forma gráfica y “humanamente legible”.

Controlador: es el intermediario entre la vista y el modelo. Es quien controla las interacciones del usuario solicitando los datos al modelo y entregándolos a la vista para que esta los presente al usuario de forma “humanamente legible”.

Funcionamiento del patrón Modelo-Vista-Controlador

Desde la vista se envían peticiones que son recibidas por el controlador y este último decide quién las lleva a cabo en el modelo. Una vez terminadas las operaciones en el modelo, se transfiere el flujo al controlador que devuelve los resultados a una vista determinada.

Modelo-Vista-Controlador

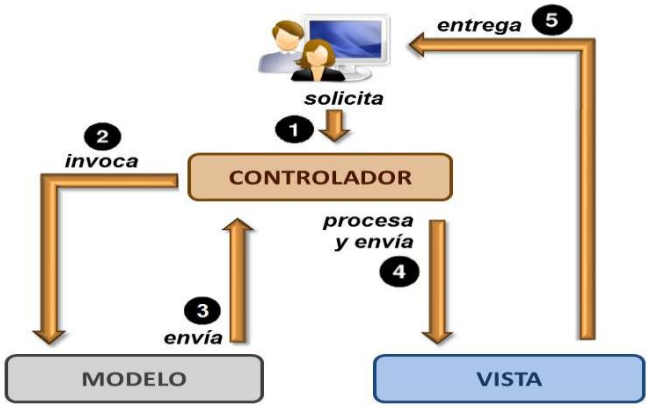


Figura 8. Funcionamiento del patrón Modelo-Vista-Controlador.

2.9. Patrones de diseño

Un Patrón de diseño define un esquema de refinamiento de los subsistemas o componentes dentro de un sistema, o las relaciones entre estos. Describe una estructura común y recurrente de componentes interrelacionados, que resuelve un problema general de diseño dentro de un contexto particular (MARQUINA y PARRA, 2008).

En la solución propuesta se evidencian varios patrones de diseño pertenecientes al grupo GRASP. Estos patrones describen los principios fundamentales de la asignación de responsabilidades a objetos (LARMAN, 1999).

Experto: asignar una determinada responsabilidad a la clase que tiene la información necesaria para cumplirla. Se observa en las librerías que cuentan con la información necesaria para realizar las operaciones invocadas por la controladora.

Creador: asignar a una clase la responsabilidad de crear una instancia de otra clase. Se evidencia en las clases controladoras, el objeto *load* es el encargado de cargar los elementos del marco de trabajo (librerías, modelos, vistas).

Controlador: asignar la responsabilidad del manejo de los eventos de un sistema a una clase. Su utilización se evidencia en las clases controladoras que se encargan de la obtención y procesamiento de los datos para enviarlos a las modelos, librerías y vistas.

Bajo acoplamiento: asignar una responsabilidad para mantener bajo acoplamiento[27].

Alta cohesión: asignar una responsabilidad de modo que la cohesión[28] siga siendo alta.

La propia implementación de CodeIgniter contiene estos dos últimos patrones nivelados pues permite el uso de los componentes de forma individual.

2.10. Estándar de codificación

Con el propósito de hacer el código uniforme y mejorar el rendimiento de la aplicación, se determinó cumplir con los parámetros que establece el estándar de codificación propuesto por el proyecto. Dichos estándares no son más que un conjunto de reglas específicas a un lenguaje, que reducen perceptiblemente el riesgo de que los desarrolladores introduzcan errores. Es notable destacar que los estándares de codificación no destapan problemas existentes, más bien evitan que los errores ocurran.

Algunas de las restricciones que establece el estándar de codificación a emplear son:

- Se debe programar en un solo idioma, preferentemente el español.
- Los inicios y cierres de ámbito deben ir en una nueva línea y estar alineados debajo de la declaración a la que pertenecen.
- El cuerpo de declaraciones compuestas (declaraciones como if, while, for, etcétera) no debe anidarse a una profundidad mayor de cinco.

[27] Es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo acoplamiento no depende de muchas otras clases.

[28] Es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.

- En caso de realizar comentarios, estos deben ser escritos de manera completa y exacta, se deben evitar los comentarios obvios, así como el uso de abreviaturas en los nombres de clases o funciones.

2.11. Conclusiones parciales

- La confección del Modelo de dominio permite obtener una idea más definida de los conceptos que intervienen en el contexto del módulo proveedor de datos para ArchiVenHIS.
- Las características de la solución propuesta están descritas a través de los artefactos que genera la metodología RUP.
- Para el desarrollo del sistema se emplea el patrón arquitectónico Modelo-Vista-Controlador.

Capítulo III. Implementación y pruebas del módulo para garantizar la prestación de servicios según el estándar OAI-PMH para el SGDH

En el presente capítulo quedan reflejados los principales elementos referentes a la implementación del módulo, los diagramas de componentes de los dos casos de uso seleccionados y el diagrama de despliegue. Se incluye, además, todo lo referente a las pruebas que se realizaron al sistema.

3.1. Diagrama de componentes

Los diagramas de componentes ilustran las piezas del software que conforman un sistema. Muestran los elementos de diseño de un sistema de software, permitiendo visualizar con más facilidad su estructura general y el comportamiento del servicio que estos componentes proporcionan y utilizan a través de las interfaces.

Los componentes representan todos los tipos de elementos software que se incluyen en la fabricación de aplicaciones informáticas, por tanto pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, etcétera. A continuación quedan plasmados los diagramas de componentes de los dos casos de uso que se consideran fundamentales para el desarrollo del sistema, el resto se puede encontrar en el Anexo 6.

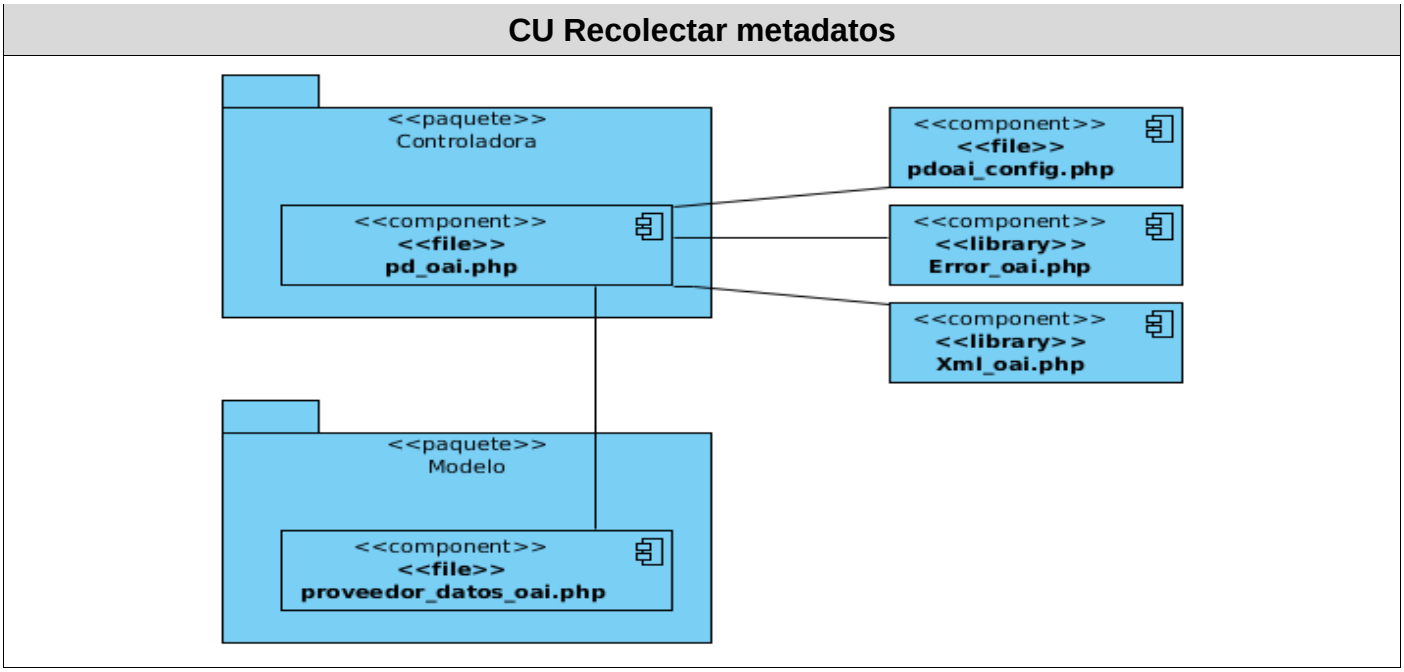


Figura 9. Diagrama de componentes del caso de uso Recolectar metadatos.

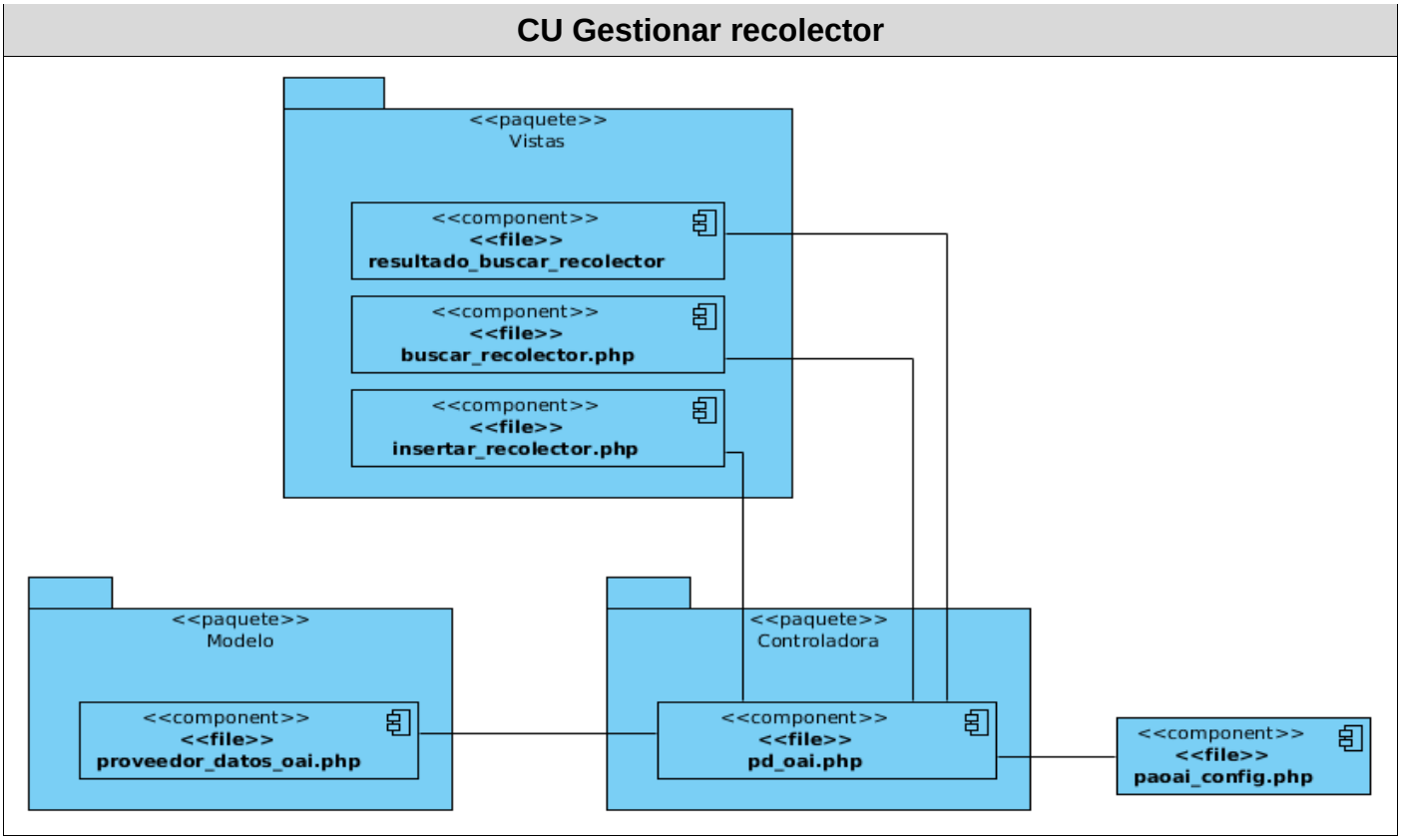


Figura 10. Diagrama de componentes del caso de uso Gestionar recolector.

3.2. Modelo de despliegue

Los diagramas de despliegue muestran la disposición física de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria. A continuación se describen los elementos que componen el diagrama de despliegue correspondiente al módulo proveedor de datos para el SGDH.

Nodo PC cliente: se refiere a las computadoras que utilizarán los usuarios para interactuar con la aplicación, en este caso se trata del proveedor de servicios. Se comunica con el Servidor de Aplicación a través del protocolo HTTP.

Nodo Servidor web: representa al servidor Apache donde se encontrará instalado el sistema.

Nodo Servidor de base de datos: servidor MySQL donde se almacenarán los datos relacionados con el sistema.

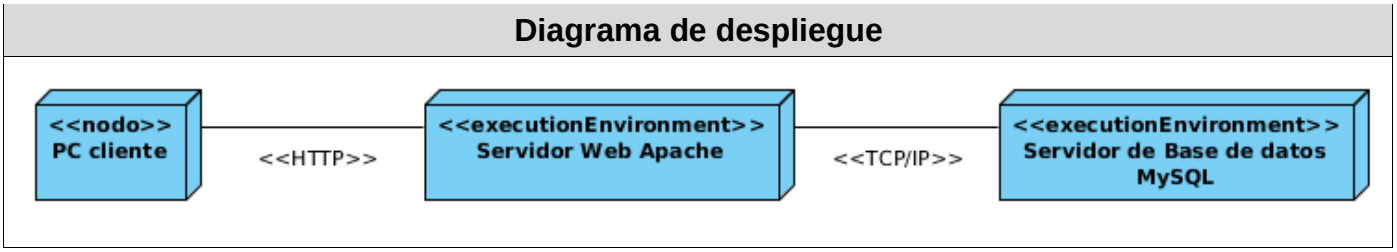


Figura 11. Diagrama de despliegue.

3.3. Pruebas

La calidad de los sistemas informáticos se ha convertido en uno de los principales objetivos estratégicos de las organizaciones debido a que su éxito depende en gran medida de esta. Para hablar del aseguramiento de la calidad se debe también hablar de control de calidad, que son procedimientos que se establecen para poder verificar y medir la aptitud de un producto, servicio o proceso. Se puede definir entonces que el aseguramiento de la calidad es la labor continua de garantizar que el control se aplica y el resultado es un producto de calidad (LUNA HERNÁNDEZ, 2010).

Para garantizar la calidad del sistema se aplicaron pruebas de caja de negra[29], diseñando nueve casos de prueba basados en los casos de uso definidos, cada uno de ellos con sus respectivas secciones y escenarios. Por la extensión de los mismos no fue posible incluirlos en el presente documento, pero todos pueden consultarse en el expediente de proyecto.

Resultado de las pruebas aplicadas al sistema

Los nueve casos de prueba diseñados fueron aplicados en una iteración por parte del propio equipo de desarrollo, obteniendo un total de 9 no conformidades, de ellas tres significativas, cinco no significativas y una recomendación. Todas las no conformidades encontradas fueron corregidas y posteriormente se realizó una segunda iteración que no arrojó no conformidades. La siguiente imagen representa los resultados obtenidos en la primera iteración.

[29] Pruebas que se llevan a cabo sobre la interfaz del software.

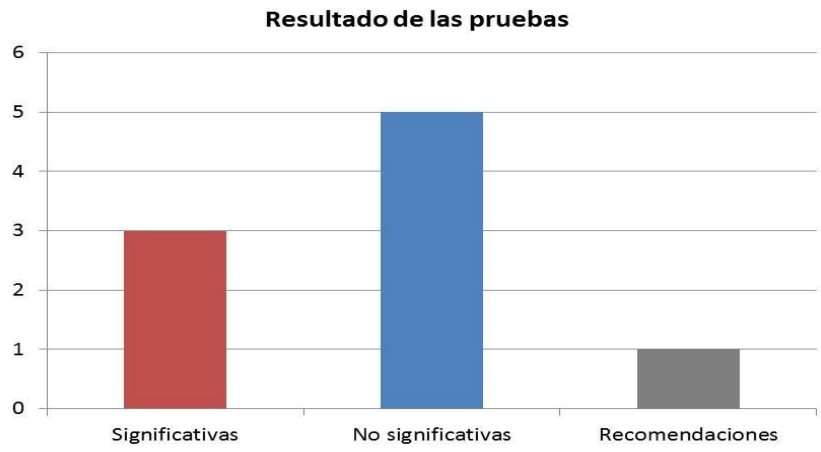


Figura 12. Resultado de las pruebas.

3.4. Conclusiones parciales

- El diagrama de componentes muestra el modo en que se encuentran estructurados los componentes del sistema.
- El diagrama de despliegue refleja los elementos físicos necesarios para el despliegue del sistema y la distribución que deben tener los mismos.
- Las pruebas funcionales aplicadas permitieron detectar un conjunto de no conformidades que afectaban la calidad del sistema.

Conclusiones generales

- El protocolo para la interoperabilidad que mejor se adapta a las características de un SGDH es OAI-PMH.
- Se descarta la utilización de las soluciones encontradas por no ajustarse a las particularidades del sistema que se necesita.
- El proceso de desarrollo estuvo guiado por la metodología RUP, permitiendo describir las características de la solución propuesta.
- Las pruebas funcionales aplicadas permitieron recopilar un conjunto de no conformidades, las cuales fueron eliminadas con el objetivo de garantizar la calidad del sistema.
- El módulo para garantizar que ArchiVenHIS se comporte como un proveedor de datos según el estándar OAI-PMH cumple con los requisitos establecidos.

Recomendaciones

Teniendo en cuenta las experiencias obtenidas con el desarrollo de la investigación se recomienda:

- Migrar el sistema a una versión de CodeIgniter más actual.
- Internacionalizar el módulo.
- Publicar la URL a través de la cual se podrán realizar peticiones OAI-PMH al sistema.

Referencias bibliográficas

ALONSO MARTÍNEZ, H. y SOLER ARCIA, D. *Desarrollo del Módulo de Reportes del Sistema Integrado de Gestión Bibliotecaria Koha para la Biblioteca Nacional de Cuba José Martí.* Universidad de las Ciencias Informáticas, 2010.

BAHIT, E. *El paradigma de la Programación Orientada a Objetos en PHP y el patrón de arquitectura de Software MVC.* 2011. 66 p.

BARRUECO, J. M. y KRICHEL, T. Acceso a prepublicaciones sobre economia: RePEc. *El profesional de la información.*, 1999, nº [Consultado el: 23 febrero 2012]. Disponible en: http://www.elprofesionaldelainformacion.com/contenidos/1999/diciembre/acceso_a_prepublicaciones_sobre_economia_repec.html. ISSN 1386-6710.

BARRUECO, J. M. y SUBIRATS COLL, I. OAI-PMH: Protocolo para la transmisión de contenidos en Internet. *El profesional de la información.*, 2003, vol. 12, nº p. 13. [Consultado el: 31 octubre 2011]. Disponible en: http://eprints.rclis.org/bitstream/10760/4093/1/cardedeu.pdf.

BRITO ACUÑA, K. *Selección de metodologías de desarrollo para aplicaciones Web en la facultad de informática de la Universidad de Cienfuegos.* Facultad de Informática, Carrera de Ingeniería Informática Universidad de Cienfuegos "Carlos Rafael Rodríguez" 2009.

BUENO-DE-LA-FUENTE, G. y RODRÍGUEZ-MATEOS, D. Herramientas de software para OAI-PMH. 2007, nº p. 49. [Consultado el: 11 de noviembre de 2011]. Disponible en: http://e-archivo.uc3m.es/handle/10016/9088. ISSN 978-958-9121-89-4.

CARPENTER, L. *Historia y desarrollo del OAI-PMH* [Página Web]. Universidad de Bath, Última actualización: 14 de octubre de 2003. [Consultado el: 31 de octubre de 2011]. Disponible en: http://travesia.mcu.es/portalnb/jspui/html/10421/1823/intro.htm.

CIBERAULA. *Introducción, definición y evolución de PHP* [Página Web]. Madrid: [Consultado el: 26 febrero de 2012]. Disponible en: http://php.ciberaula.com/articulo/introduccion_php.

COMITÉ DE METADATOS DE LA BIBLIOTECA NACIONAL DE CHILE. *Manual para la creación de metadatos usando Dublin Core.* Chile: 2006, 99 p.

CRUZ MUNDET, J. R. *Administración de documentos y archivos. Textos fundamentales.* Madrid: 2011. 468 p. ISBN 978-84-615-5150-7.

CHAVEZ GONZÁLEZ, M. *Ley de 9 de enero de 1984, Archivos de Andalucía.* Andalucía: Boletín Oficial de la Junta de Andalucía, 1999, [Consultado el: 03 abril 2012]. Disponible en: http://webs.um.es/isgil/Ley%20de%20Archivos%201984%20Andalucia%20Gil%20Leiva.pdf.

DAVIS, J.; FIELDING, D., *et al. Dienst Protocol Specification* [Página Web]. [Consultado el: 18 de marzo de 2012]. Disponible en: http://www.cs.cornell.edu/cdlrg/dienst/protocols/dienstprotocol.htm#2.%20Dienst%20architecture%20overview.

DORADO SANTANA, Y., MENA MUGICA, MAYRA M. . Evolución de la ciencia archivística. *ACIMED*, 2009, vol. 20, nº 1, [Consultado el: 30 abril 2012]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352009000700004&lng=es&nrm=iso. ISSN 1024-9435.

DUBLIN CORE METADATA INITIATIVE. *The Dublin Core Metadata Initiative* [Consultado el: 05 de noviembre de 2011]. Disponible en: http://dublincore.org/

ECURED. *Patrones de Casos de Uso* [Página Web]. La Habana: Última actualización: 17 mayo 2011. [Consultado el: 10 enero de 2012]. Disponible en: http://www.ecured.cu/index.php?title=Patrones_de_Casos_de_Uso&action=history.

F. ROMERO, D. *¿Qué Es Un Servidor Web?* [Página web]. Última actualización: 27 noviembre de 2007. [Consultado el: 26 de febrero de 2012]. Disponible en: http://www.editum.org/Que-Es-Un-Servidor-Web-p-401.html.

FUSTER RUIZ, F. *Archivo, Documento de Archivo... Necesidad de clarificar los conceptos* Disponible en: http://www.slideshare.net/veronicaguz22/archivo-documento-de-archivo-necesidad-de-clarificar-los-conceptos.

GLOSARIOIT. *Glosario Informático* [Diccionario en línea]. Buenos Aires: GlosarioIT, 2012, [Consultado el: 15 mayo 2012]. Disponible en: http://www.glosarioit.com/.

GÓMEZ DÍAZ, R. y BRINGAS GONZÁLEZ, R. Normalización y requisitos funcionales de la descripción archivística: una propuesta metodológica. *Scire: Representación y organización del conocimiento*, 2005, vol. 11, nº p. 103-112. [Consultado el: 17 de mayo de 2012]. Disponible en: http://eprints.rclis.org/handle/10760/13665. ISSN 1135-3761.

GÓMEZ LAUREANO, F. Interoperabilidad en los sistemas de información documental (SID): La información debe fluir. *Revista de la Facultad de Sistemas de Información y Documentación (Códice)*, 2007, vol. 3, nº p. 18. [Consultado el: 14 diciembre 2011]. Disponible en: http://redalyc.uaemex.mx/pdf/953/95330103.pdf. ISSN 1794-9815.

GUAJARDO SALINAS, A. *Z39.50 y OAI-PMH: Protocolos de Transferencia y Recuperación de Información.* Chile: 2010, vol. 2011, Disponible en: http://www.bibliotecarios.cl/descargas/2010/11/guajardo_z3950.pdf.

HEREDIA HERRERA, A. *Archivística General Teoría y Práctica.* Quinta ed. Diputación Provincial de Sevilla., 1991. ISBN 84 - 7798 - 056 - X.

HERRERA TEJADA, C. *ICA-AtoM, una buena herramienta para la difusión de los archivos en la web* Madrid: [Consultado el: 05 de mayo de 2012]. Disponible en: www.ateneodemadrid.com.

IBM CORPORATION. *Rational Unified Process.* 2006,

INSTITUTO NACIONAL DE ESTADÍSTICA E INFORMÁTICA. *Herramientas Case.* Perú: Instituto Nacional de Estadística e Informática, 1999, Disponible en: http://www.inei.gob.pe/biblioineipub/bancopub/Inf/Lib5103/Libro.pdf.

JACOBSON, I.; BOOCH, G.*, et al. El proceso unificado de desarrollo de software. .* Madrid: 2000. vol. 2012, ISBN 84-7829-036-2.

LAMARCA LAPUENTE, M. J. *Hipertexto, el nuevo concepto de documento en la cultura de la imagen.* Tesis doctoral, Universidad Complutense de Madrid, 2011.

LARMAN, C. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México: publicado el: 18 de mayo de 2012 de 1999, última actualización: 18 de mayo de 2012. Disponible en: http://eva.uci.cu/file.php/161/Documentos/Materiales_basicos/Materiales_basicos_de_la_Unidad_3/UML_y_Patrones/09_Presentacion.pdf. ISBN 970-17-0261-1.

LUNA HERNÁNDEZ, D. F. La importancia del aseguramiento de la calidad en los sistemas de información. *Ingeniería Primero*, octubre 2010, nº p. 45-48. Disponible en: http://www.tec.url.edu.gt/boletin. ISSN 2076-3166.

MARQUINA, E. y PARRA, J. D. *Guía de Patrones, Prácticas y Arquitectura .NET.* La Habana: publicado el: 7 de Febrero de 2008, última actualización: 7 de Febrero. Disponible en: http://sunshine.prod.uci.cu/gridfs/sunshine/books/PPArquitecturaNET.pdf.

MATO GARCÍA, R. M. *Diseño de Bases de datos.* 1999, [Consultado el: 11 de junio de 2012]. Disponible en: http://eva.uci.cu/mod/resource/view.php?id=11576&subdir=/Bibliografia_Basica.

MATOS PADILLA, R. *Programación Web y Tecnologías Informáticas* [Página web]. Creative Commons by-nc-sa 3.0, Última actualización: 28 de febrero de 2008 Disponible en: http://zenkius.blogspot.com/2008/02/tecnologias-del-lado-del-cliente.html.

MENA MUGICA, M. *Gestión documental y organización de los archivos.* La Habana: 2005. 96 p. ISBN 959-258-950-X.

MENÉNDEZ-BARZANALLANA ASENSIO, R. *Informática Aplicada a la Gestión Pública. Facultad Derecho UMU.* [Página web]. España: Universidad de Murcia, Última actualización: 2011/10/13. [Consultado el: 24 de abril de 2012]. Disponible en: http://www.um.es/docencia/barzana/IAGP/IAGP2-Metodologias-de-desarrollo.html.

MICROSOFT CORPORATION. *Diccionario de Informática e Internet de Microsoft* Editado por: González, C. S. España: McGraw-Hill/Interamericana de España, 2000, [Consultado el: 20 marzo 2012]. 844 p. ISBN 84-481-2893-1

MINISTERIO DE CULTURA. *Archivística.* Segunda ed. España: 1995, Disponible en: http://www.mcu.es/archivos/MC/DTA/Diccionario.html#archiv%C3%ADstica. ISBN 84-8181-066-5.

MYSQL. *Las principales características de MySQL* [Página Web]. [Consultado el: 18 de enero de 2012]. Disponible en: http://dev.mysql.com/doc/refman/5.0/es/features.html.

ODILOTID. *OdiloA3W* [Consultado el: 08 de noviembre de 2011]. Disponible en: http://www.odilotid.es/general/odiloa3w.htm.

OPEN ARCHIVES INITIATIVE. *Open Archives Initiative Protocol for Metadata Harvesting* [Consultado el: 06 de octubre de 2011]. Disponible en: http://www.openarchives.org.

PEIS, E. y RUIZ RODRÍGUEZ, A. A. *EAD (Encoded Archival Description): Desarrollo, estructura, uso y aplicaciones* [Página Web]. Barcelona: Universidad Pompeu Fabra, Última actualización: 06 mayo 2011. [Consultado el: 16 de febrero de 2012]. Disponible en: http://www.hipertext.net/web/pag223.htm.

QUERO CATALINAS, E.; GARCÍA ROMÁN, A.*, et al. Mantenimiento de Portales para la Información.* Editado por: Tompson Editores Spain. Madrid: Paraninfo S.A., 2007, 407 p. Disponible en: http://books.google.com.cu/books?id=tetmS1ORsHoC&printsec=frontcover&hl=es#v=onepage&q&f=false. ISBN 976-84-9732-504-2.

STANFORD DIGITAL LIBRARY TECHNOLOGIES. *The Simple Digital Library Interoperability Protocol* [Página Web]. California: [Consultado el: febrero de 2012]. Disponible en: http://dbpubs.stanford.edu:8091/~testbed/doc2/SDLIP/SDLIPProjDesc/sdlipProject.htm.

THE JQUERY FOUNDATION. *jQuery is a new kind of JavaScript Library.* [Página web]. [Consultado el: 15 de marzo de 2012]. Disponible en: http://jquery.com/.

THE SOCIETY OF AMERICAN ARCHIVISTS. *Encoded Archival Description Tag Library, Version 2002.* Chicago: 2002. ISBN 1-931666-00-8.

VISUAL PARADIGM. *Boost Productivity Whith Innovative and Intuitive Tecnologies* [Página web]. [Consultado el: 3 de marzo de 2012]. Disponible en: http://www.visual-paradigm.com/.

WORLD WIDE WEB CONSORTIUM. *Extensible Markup Language (XML)* [Página web]. Tim Bray Jean Paoli C. M. Sperberg-McQueen, [Consultado el: 23 de noviembre de 2011]. Disponible en: http://www.w3.org/TR/1998/REC-xml-19980210.

Bibliografía

BARROSO HORTA, L. y DELGADO ARCEO, M. M. *Los Archivos Históricos como fuente de conocimiento y cultura para los estudiantes universitarios* Villa Clara: Archivo Histórico Provincial de Villa Clara, [Consultado el: 17 febrero de 2012]. Disponible en: http://archivohistorico.villaclara.cu/estudios/los-archivos-historicos-y-las-universidades.

BUENO-DE-LA-FUENTE, G. *Análisis de la interoperabilidad entre los sistemas de apoyo a la formación de TecMinho.* Universidade do Minho. Departamento de Sistemas de Informaçao: 2008, Disponible en: http://e-archivo.uc3m.es/handle/10016/9089.

CABEZAS BOLAÑOS, E. La descripción archivística y su aplicación en documentos particulares: El caso del álbum de Figueroa. . *Diálogos Revista Electrónica de Historia*, 2000, vol. 1, nº 002, p. 9. [Consultado el: 23 enero 2012]. Disponible en: http://redalyc.uaemex.mx/pdf/439/43910207.pdf. ISSN 1409-469X.

CIBERAULA. *Patrones de Diseño en aplicaciones Web con Java J2EE* [Consultado el: 25 de abril de 2012]. Disponible en: http://java.ciberaula.com/articulo/diseno_patrones_j2ee.

CONFEDERATION OF OPEN ACCES REPOSITORIES. *El caso de Interoperabilidad para Repositorios de Acceso Abierto.* Publicado el: 25 de octubre del 2011, última actualización: 25 de octubre del 2011. Disponible en: http://www.coar-repositories.org/files/de_la_investigaci%C3%B3n-a-trav%C3%A9s-de-redes-globales-de-Repositorios-de-Acceso-Abierto-final-version.pdf.

DE VOLDER, C. *Los repositorios de acceso abierto en Argentina: situación actual* Centro de Documentación e Información, Instituto de Investigaciones Gino Germani, Facultad de Ciencias Sociales, Universidad de Buenos Aires., publicado el: 25 de enero de 2008, última actualización: 25 de enero de 2012. Disponible en: http://www.scielo.org.ar/scielo.php?script=sci_arttext&pid=S1851-17402008000200005. ISBN 1851-1740.

LUNA HERNÁNDEZ, D. F. La importancia del aseguramiento de la calidad en los sistemas de información. *Revista Ingeniería primero*, octubre 2010 2010, nº 19, p. 45-48. [Consultado el: 06 de diciembre 2012]. Disponible en: http://www.tec.url.edu.gt/boletin/URL_19_SIS09_GESTION.pdf. ISSN 2076-3166.

MARTÍNEZ, J. A. y LARA, P. Interoperabilidad de los contenidos en las plataformas de e-learning: normalización, bibliotecas digitales y gestión del conocimiento. *RU&SC. Revista de Universidad y Sociedad del Conocimiento.*, 2006, vol. 3, nº 002, Disponible en: http://redalyc.uaemex.mx/pdf/780/78030206.pdf. ISSN 1658-580X.

MELERO, R. Acceso abierto a las publicaciones científicas: definición, recursos, copyright e impacto. *El profesional de la información.*, 2005, vol. 14, nº 4, p. 255-266. [Consultado el: 15 noviembre 2011]. Disponible en: http://digital.csic.es/handle/10261/1486.

MELERO, R. y PÉREZ AGÜERA, J. R. Plataforma digital de revistas científicas electrónicas españolas. Relación con el Movimiento Open Access. *Scripta Nova. Revista electrónica de Geografía y Ciencias Sociales.*, 1 de agosto de 2004 2004, vol. VIII, nº 170 (74), [Consultado el:

25 de octubre de 2011]. Disponible en: http://www.ub.edu/geocrit/sn/sn-170-74.htm. ISSN 1138-9788.

OCHOA AGÜERO, A.; SÁCHEZ MANSOLO, A.*, et al. Repositorios de objetos de aprendizaje de Acceso Abierto para la educación de postgrado.* Venezuela: Misión Médica Cubana en la República Bolivariana de Venezuela, publicado el: 26 de marzo de 2010, última actualización: 26 de marzo de 2012. Disponible en: http://www.rcim.sld.cu/revista_21/articulo_pdf/repositorioaprendizaje.pdf.

REPOSITORIO INSTITUCIONAL DE LA UNIVERSIDAD DE LOS ANDES. *¿Qué es Dspace?* Mérida, Venezuela: [Consultado el: 06 de diciembre de 2011]. Disponible en: http://www.saber.ula.ve/que_dspace.jsp.

SERVICIO DE CAROTGRAFÍA DIGITAL E INFRAESTRUCTURA DE DATOS ESPACIALES. *Dspace* Última actualización: 15 de octubre de 2010. [Consultado el: 18 de febrero de 2012]. Disponible en: http://secad.unex.es/portal/index.php?option=com_content&view=article&id=83:dspace&catid=35:faqsecad&Itemid=59.

SIZA RAMÍREZ, J. P. *Estudio, análisis, evaluación e implementación de un protocolo de intercambio de contenidos sobre internet para la interoperabilidad de repositorios de la Biblioteca Digital en la Universidad Nacional de Colombia con un proveedor de servicios de contenidos.* Tutor: Ramos De Florez, Z. I. Tesis de maestría en Ingeniería - Telecomunicaciones., Facultad de Ingeniería. Universidad Nacional de Colombia, 2010.

UNIVERSIDAD CARLOS III DE MADRID. *Dublin Core. Metadatos para describir e identificar un documento en la red.* Madrid: Universidad Carlos III de Madrid, [Consultado el: 18 de noviembre de 2011]. Disponible en: http://www.metadatos-xmlrdf.com/metadatos/dublin-core.

UNIVERSITY OF SOUTHAMPTON. *E-Prints* [Consultado el: 25 de marzo de 2012]. Disponible en: http://www.eprints.org/.

Anexos

Anexo 1

Figura 1.1. Funcionamiento básico de OAI-PMH.

Anexo 2

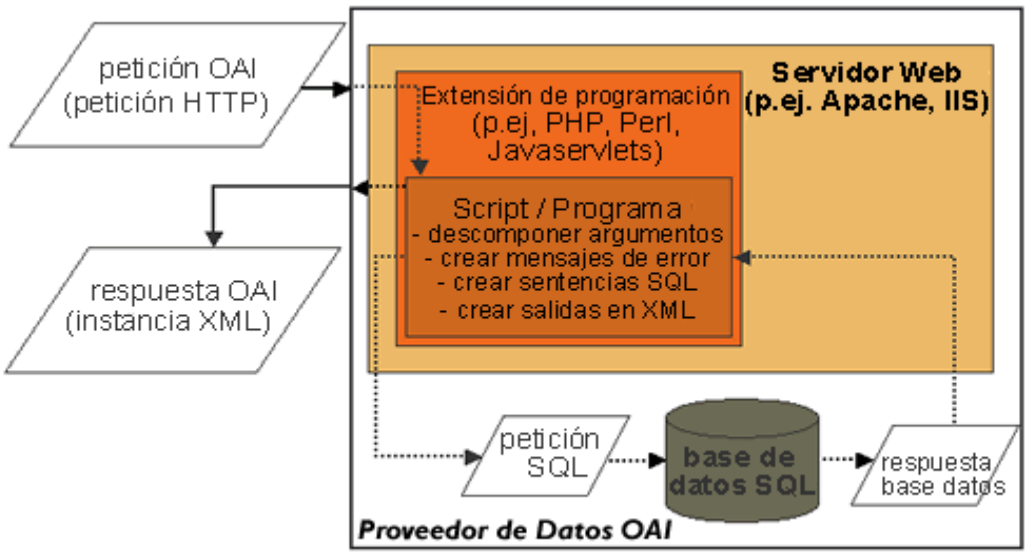


Figura 2.1. Arquitectura de un proveedor de datos.

Anexo 3

ISAD(G)	EAD
3.1.1 Código de referencia.	<unitid>
3.1.2 Título	<unittitle>
3.1.3 Fechas	<unitdate>
3.1.4 Nivel de descripción	<archdesc>, <c>
3.1.5 Volumen y soporte	<physdesc>, <extent>
3.2.1 Nombre de los productores	<origination>
3.2.2 Historia institucional/Reseña bibliográfica	<bioghist>
3.2.3 Historia archivística	<custodhist>
3.2.4 Forma de ingreso	<acqinfo>
3.3.1 Alcance y contenido	<scopecontent>
3.3.2 Valoración, Selección y Eliminación	<appraisal>
3.3.3 Nuevos ingresos	<accruals>
3.3.4 Organización	<arrangement>
3.4.1 Condiciones de acceso	<accessrestrict>
3.4.2 Condiciones de reproducción	<userestrict
3.4.3 Lengua/Escritura de los documentos	<archdesc> con atributo LANGMATERIAL
3.4.4 Características físicas y requisitos técnicos	<phystech>
3.4.5 Instrumentos de descripción	<otherfindaid>
3.5.1 Existencia y localización de los originales	<odd>
3.5.2 Existencia y localización de copias	<altformavail>
3.5.3 Unidades de descripción relacionadas	<relatedmaterial>, <separatedmaterial>
3.5.4 Nota de publicaciones	<bibliography>
3.6.1 Notas	<odd>

Tabla 3.1. Transformación ISAD(G) a EAD.

Anexo 4

ISAD(G)	DC
3.1.1 Código de referencia.	<identifier>
3.1.2 Título	<title>
3.1.3 Fechas	<date>
3.1.3 Fechas	<coverage> (temporal)
3.1.5 Volumen y soporte	<format>
3.2.1 Nombre de los productores	<creator>
3.3.1 Alcance y contenido	<description>
3.4.1 Condiciones de acceso	<rights>
3.4.3 Lengua/Escritura de los documentos	<language>
3.4.4 Características físicas y requisitos técnicos	<type>
3.5.1 Existencia y localización de los originales	<source>
3.5.3 Unidades de descripción relacionadas	<relation>

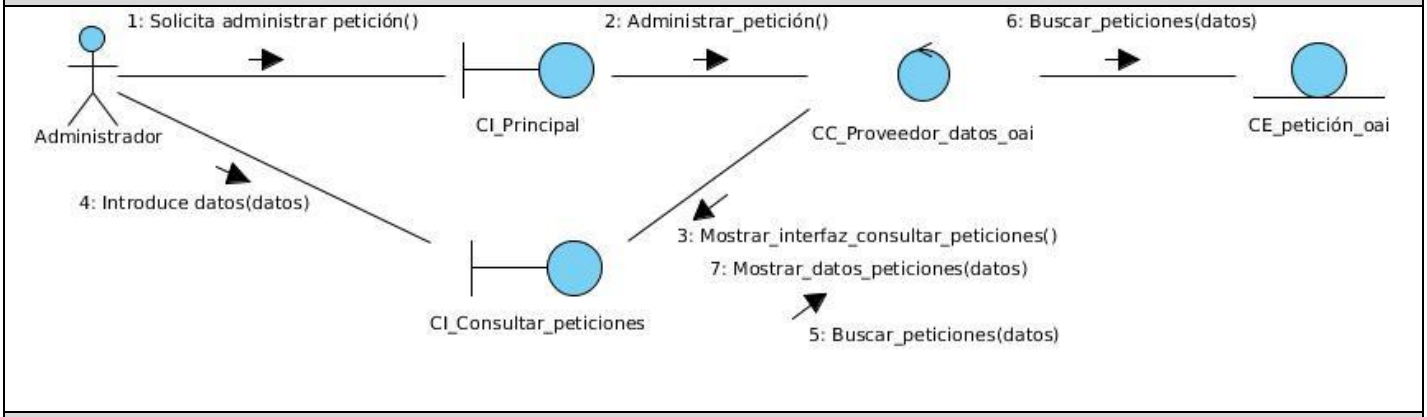
Tabla 4.1. Transformación ISAD(G) a Dublin Core.

Anexo 5

Diagramas de interacción

CU Gestionar petición

Sección Listar



Sección Eliminar

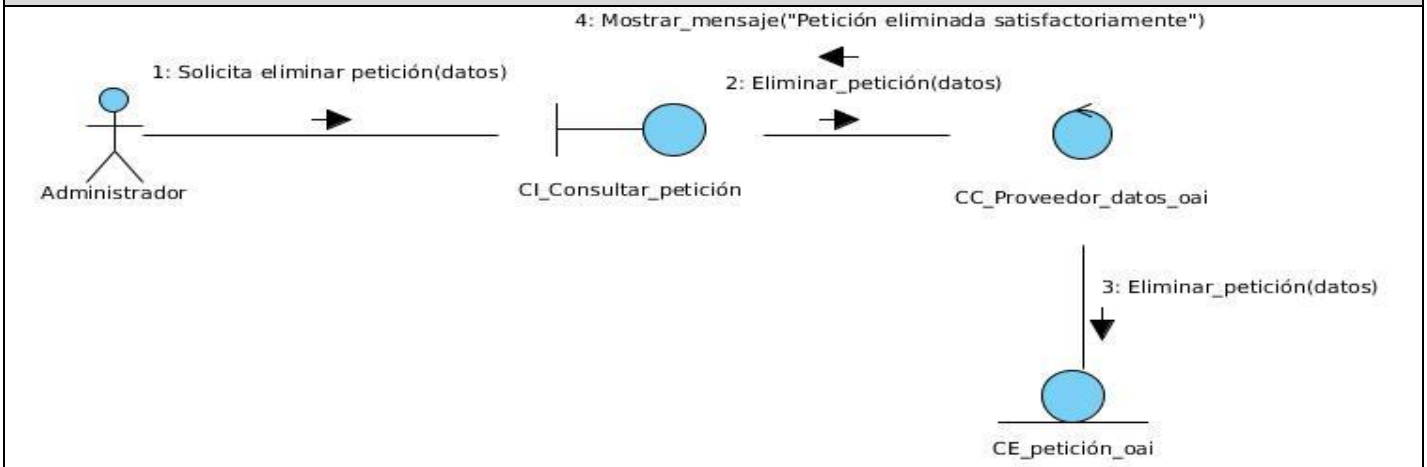


Tabla 5.1. Diagrama de interacción del CU Gestionar petición.

CU Consultar cantidad de peticiones por verbo

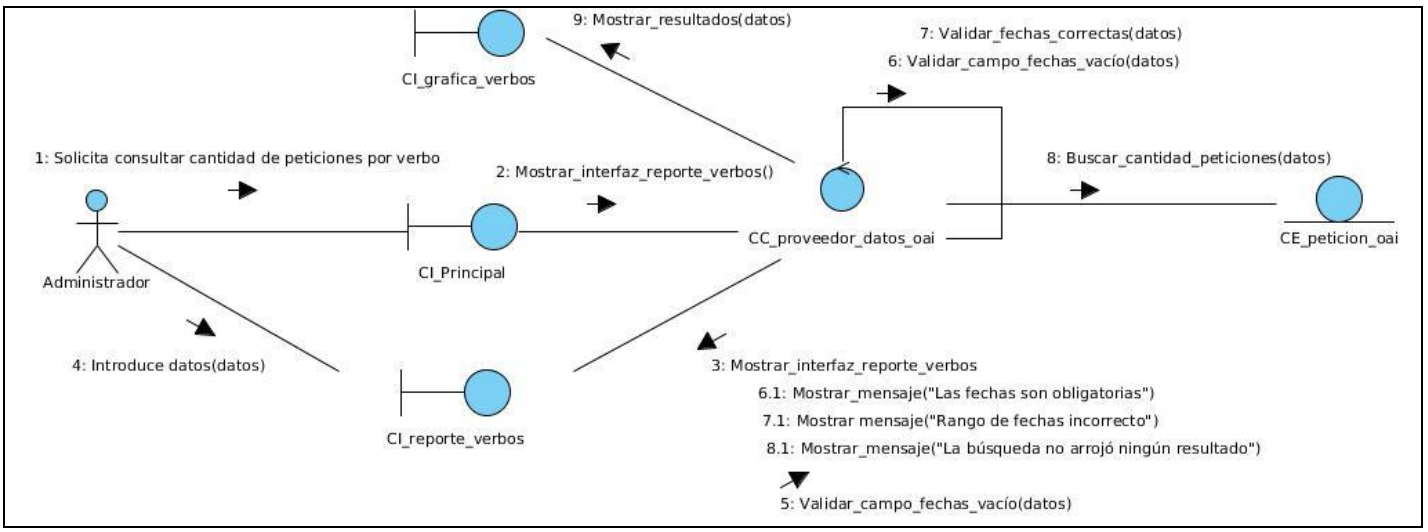


Tabla 5.2. Diagrama de interacción del CU Consultar cantidad de peticiones por verbo.

CU Consultar cantidad de peticiones por estado

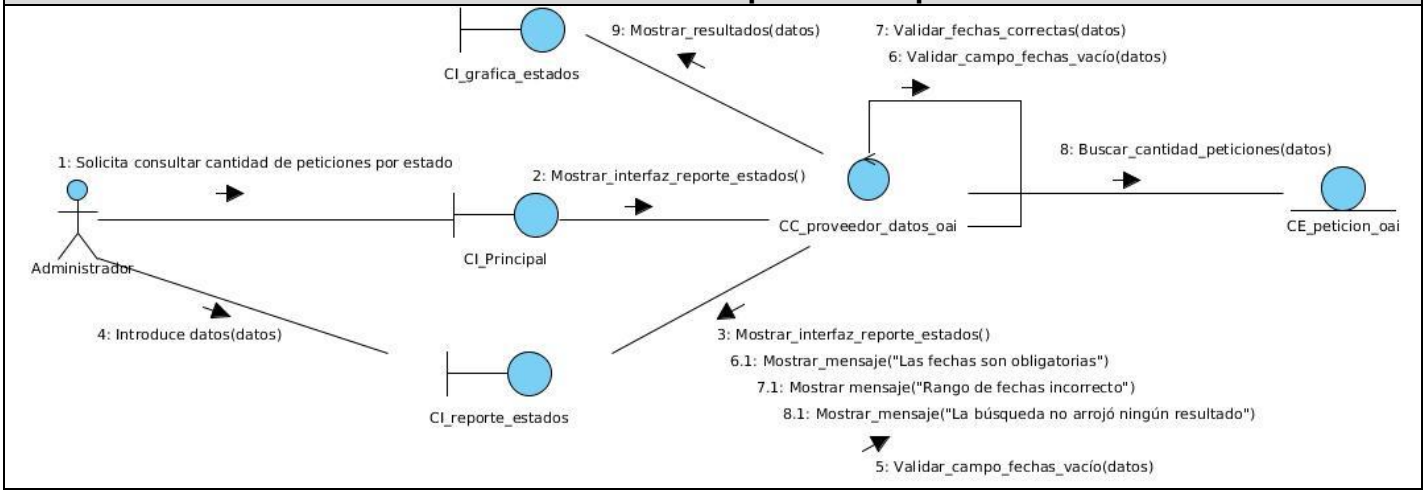


Tabla 5.3. Diagrama de interacción del CU Consultar cantidad de peticiones por estado.

CU Consultar cantidad de peticiones por tipo de respuesta

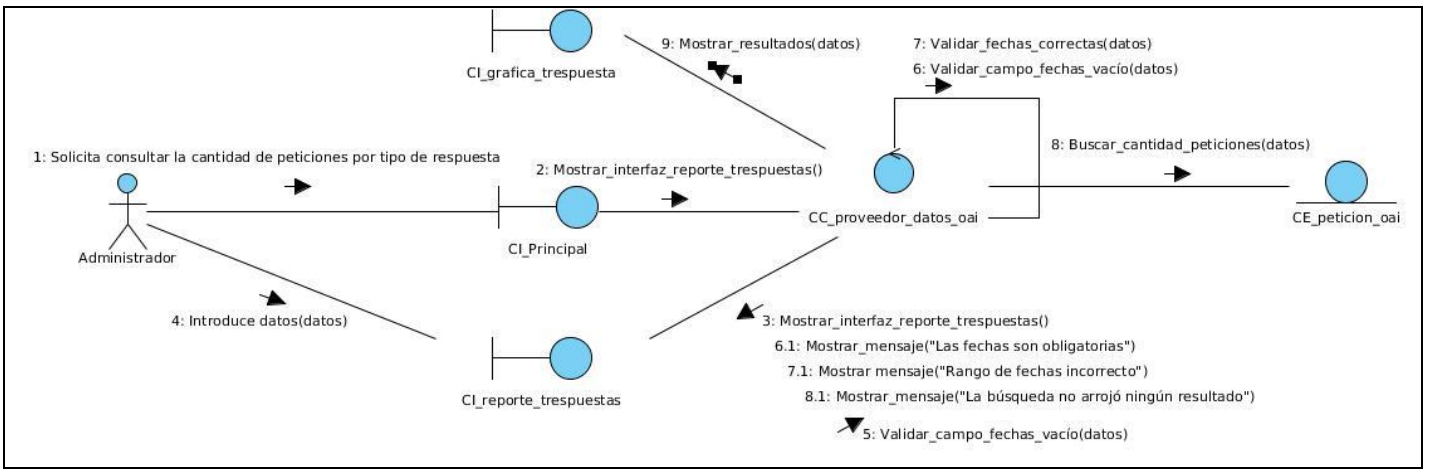


Tabla 5.4. Diagrama de interacción del CU Consultar cantidad de peticiones por tipo de respuesta.

CU Consultar cantidad de peticiones por recolector

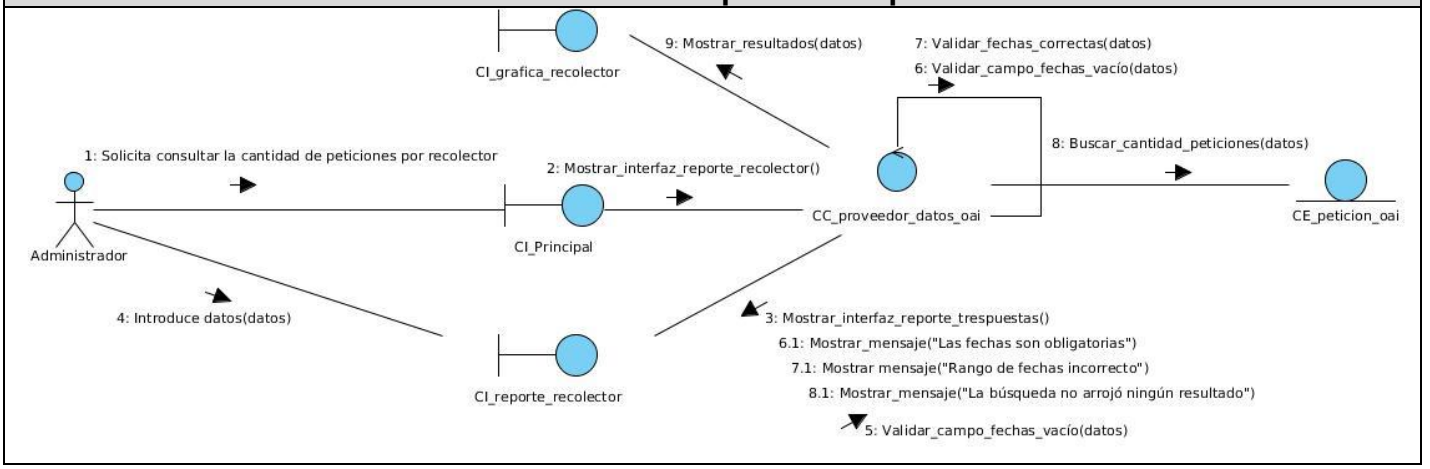


Tabla 5.5. Diagrama de interacción del CU Consultar cantidad de peticiones por recolector.

CU Consultar cantidad de descripciones por recolector

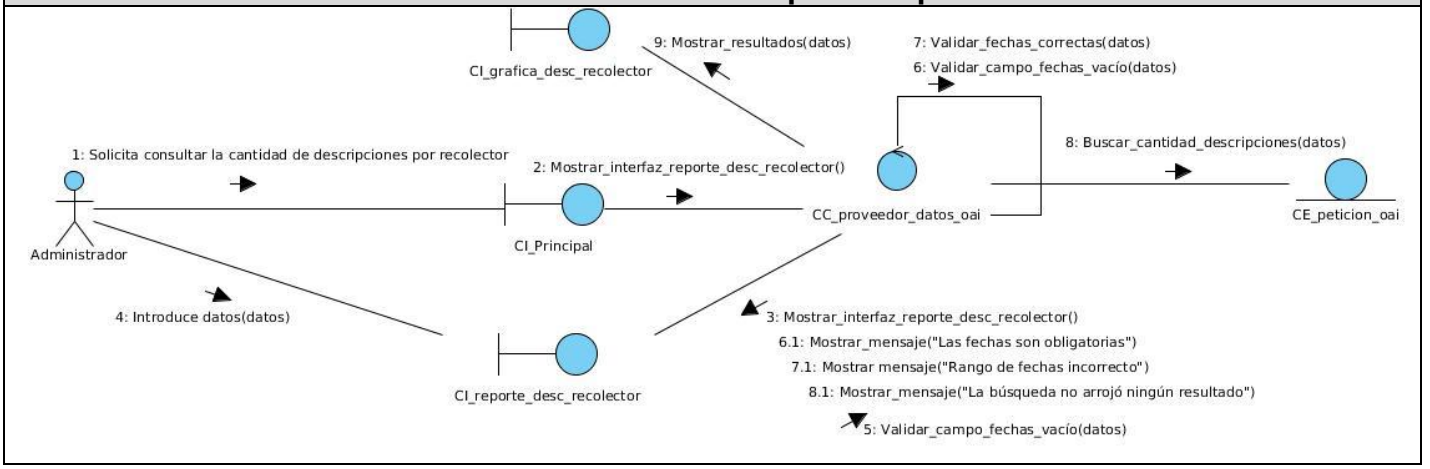


Tabla 5.6. Diagrama de interacción del CU Consultar cantidad de descripciones por recolector.

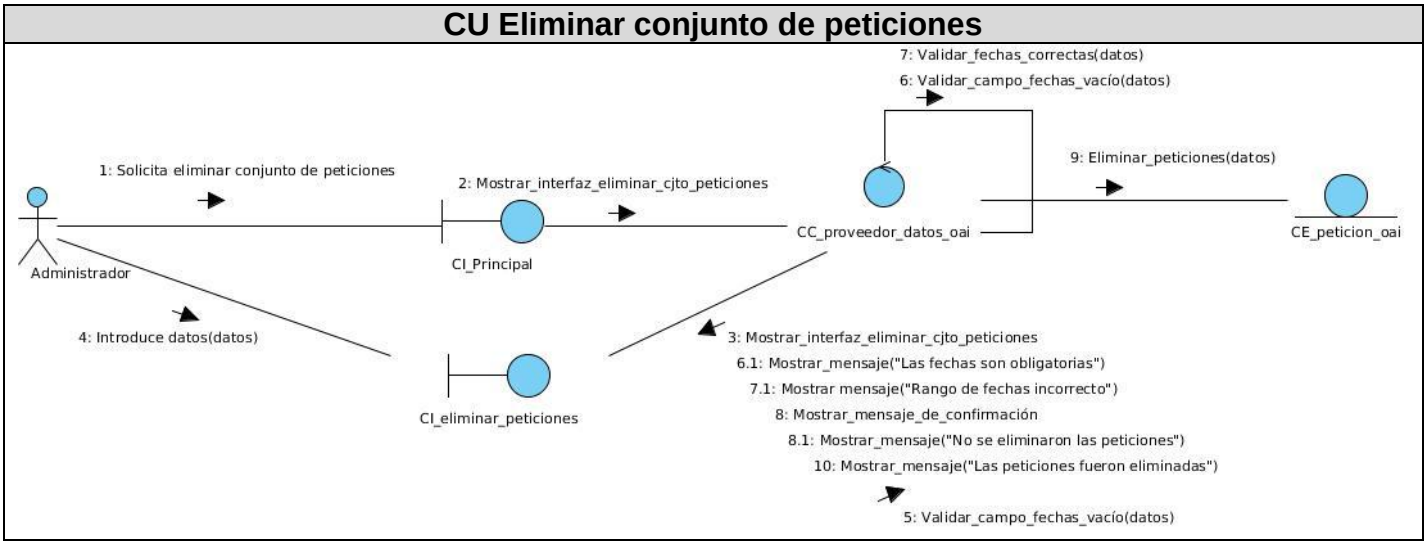


Tabla 5.7. Diagrama de interacción del CU Eliminar conjunto de peticiones.

Anexo 6

Diagramas de componentes

CU Gestionar petición

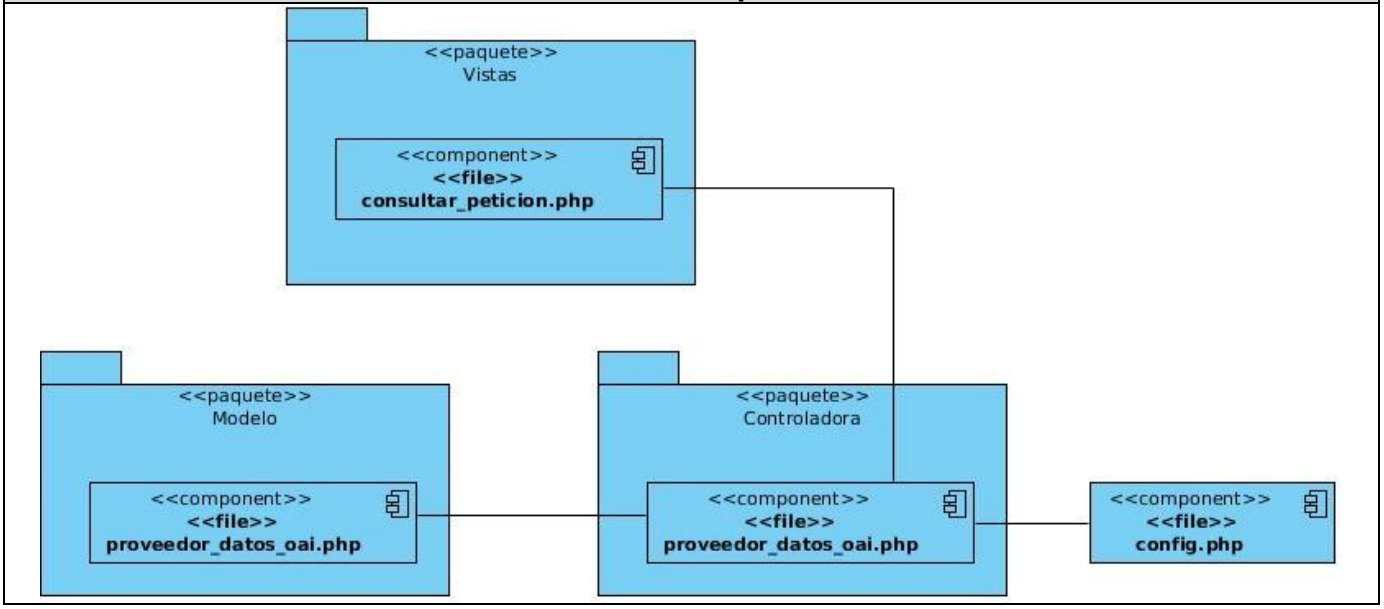


Tabla 6.1. Diagrama de componentes del CU Gestionar petición.

CU Consultar cantidad de peticiones por verbo

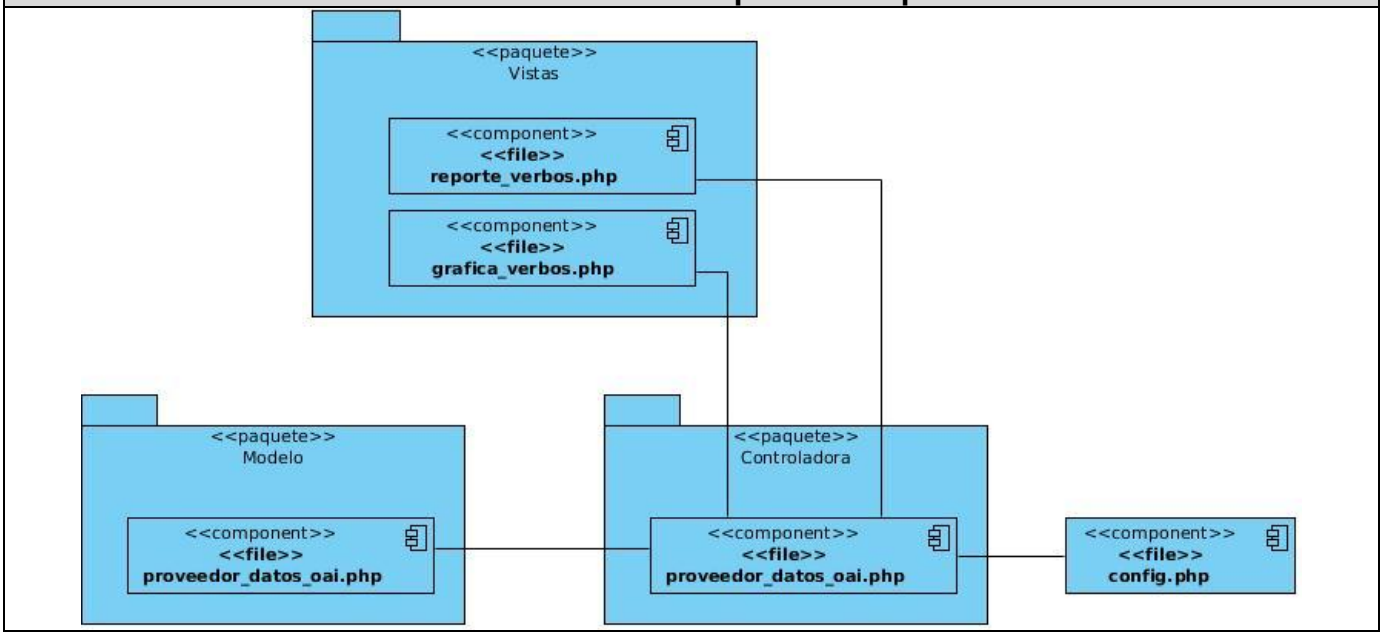


Tabla 6.2. Diagrama de componentes del CU Consultar cantidad de peticiones por verbo.

CU Consultar cantidad de peticiones por estado

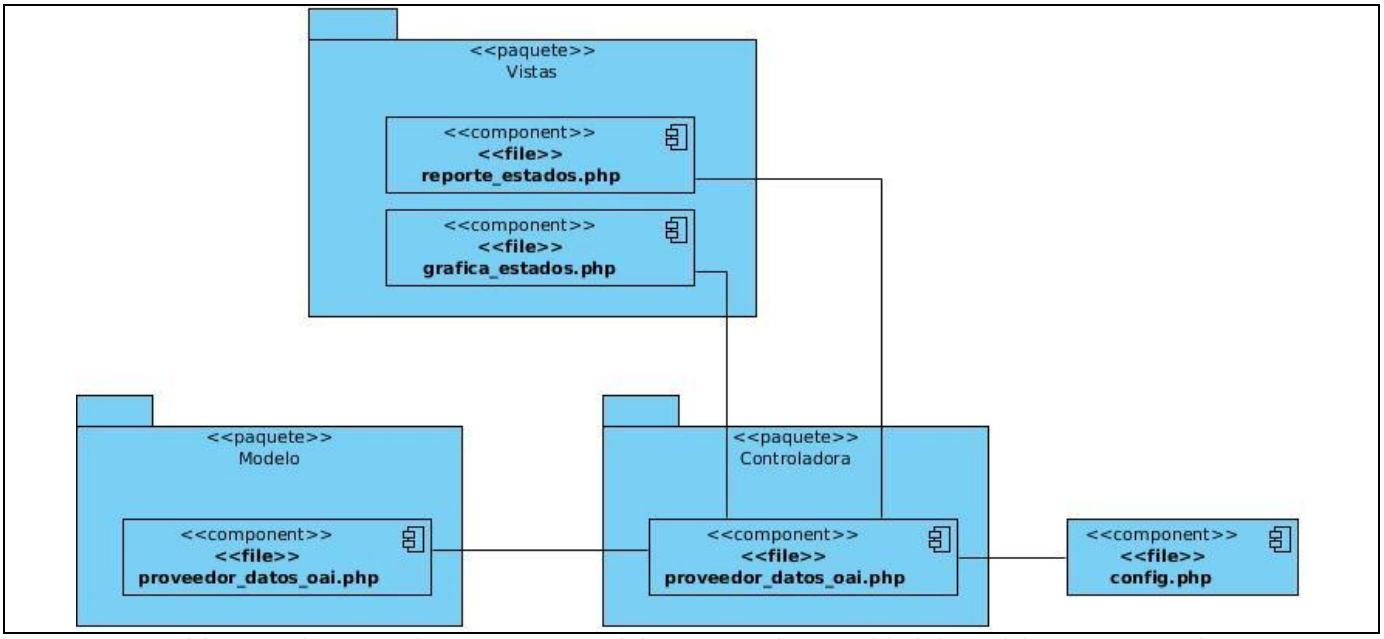


Tabla 6.3. Diagrama de componentes del CU Consultar cantidad de peticiones por estado.

CU Consultar cantidad de peticiones por tipo de respuesta

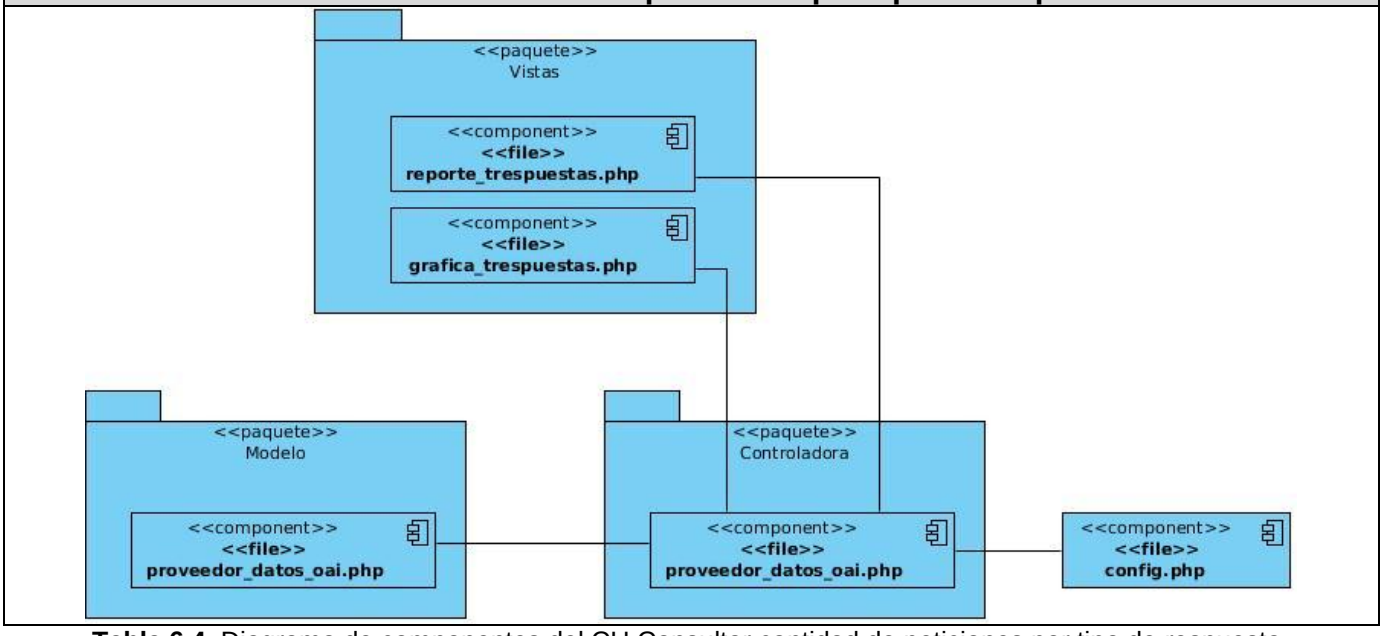


Tabla 6.4. Diagrama de componentes del CU Consultar cantidad de peticiones por tipo de respuesta.

CU Consultar cantidad de peticiones por recolector

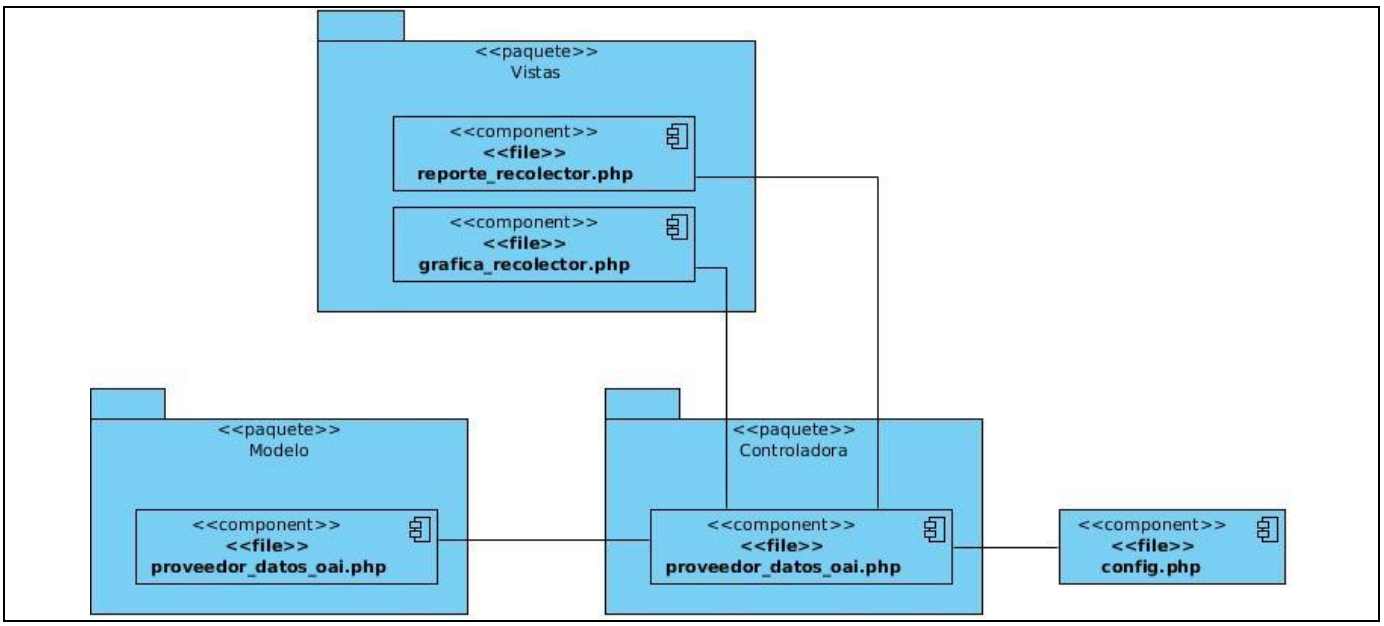


Tabla 6.5. Diagrama de componentes del CU Consultar cantidad de peticiones por recolector.

CU Consultar cantidad de descripciones por recolector

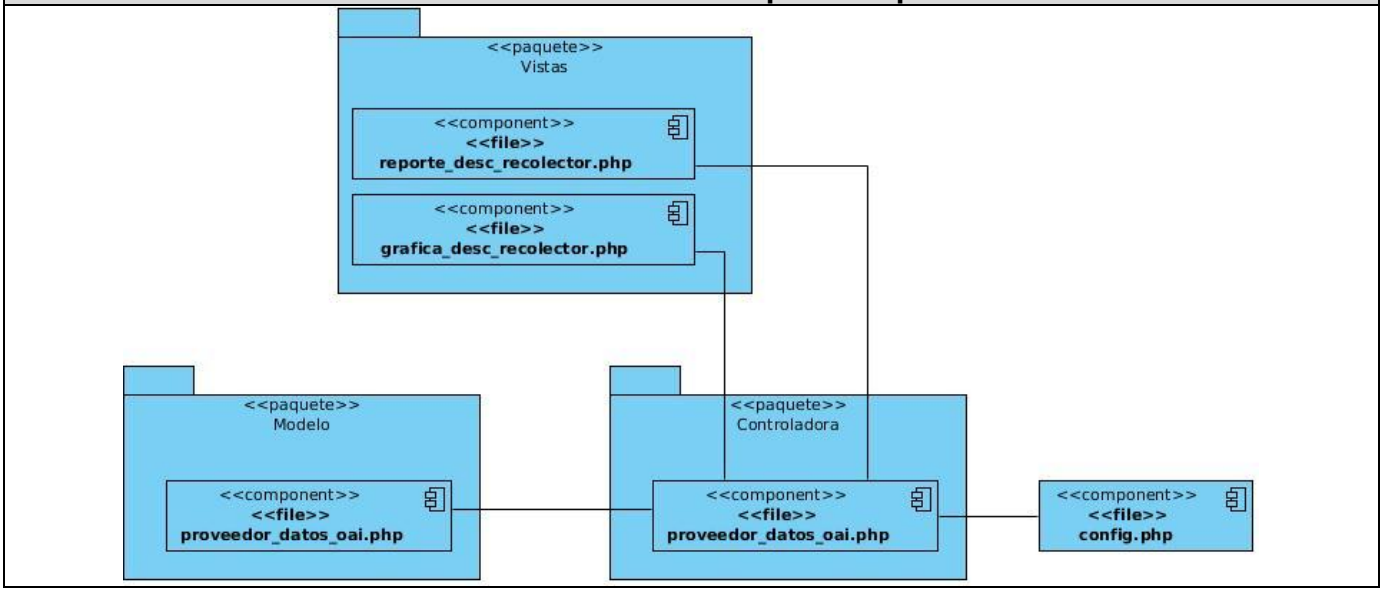


Tabla 6.6. Diagrama de componentes del CU Consultar cantidad de descripciones por recolector.

CU Eliminar conjunto de peticiones

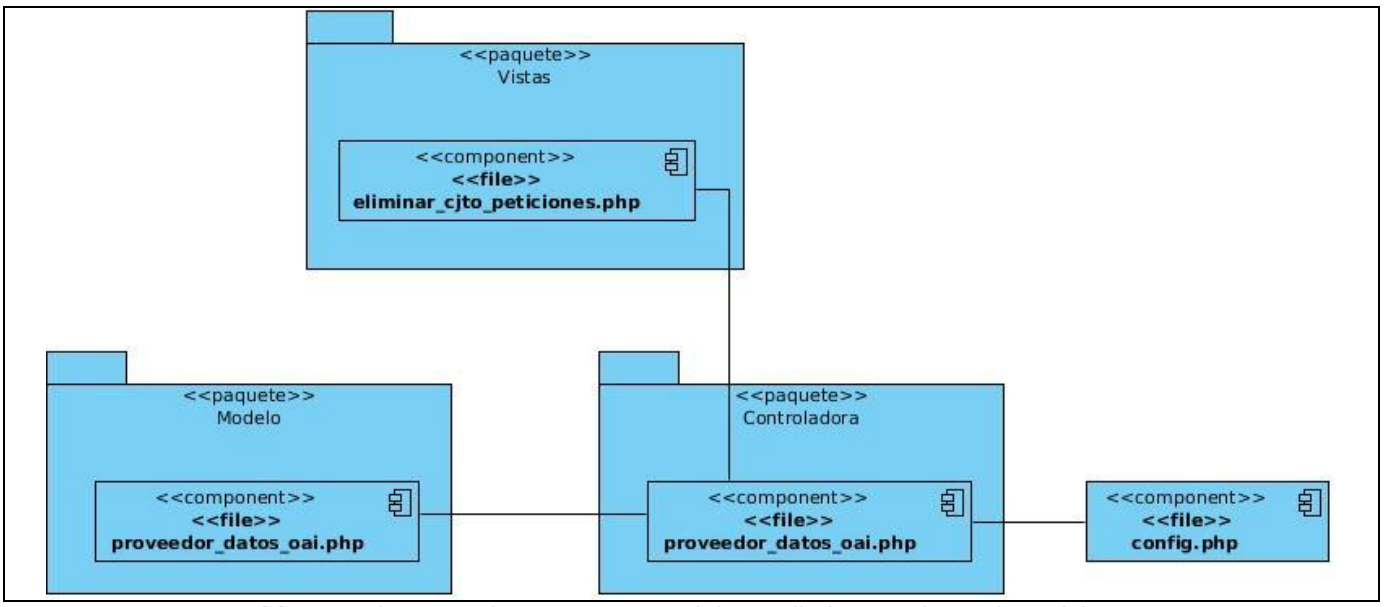


Tabla 6.7. Diagrama de componentes del CU Eliminar conjunto de peticiones.